# ЯЗЫК ОПИСАНИЯ И ПОСТРОЕНИЯ ЛЕКАЛ

***Принципиально новые возможности системы помогут Вам воплотить свои замыслы***

# ВВЕДЕНИЕ

В первой части описания приведены возможности системы ЛЕКО, последовательность установки системы на жесткий диск и изложен общий порядок работы с системой, даны ее характеристики и возможности.

В предлагаемой части описания системы ЛЕКО формально представлен язык описания и построения лекал, даны синтаксис и семантика всех языковых конструкций, предназначенных для описания построения лекал, а также приведены ссылки на связь этих построений с конструированием и моделированием лекал швейных изделий.

Как было сказано в первой части описания, в основе подхода к конструированию и моделированию одежды в системе ЛЕКО лежат расчетно-пропорциональные методы конструирования. С одной стороны, такой подход не является новым, и очень многие методики построения лекал основаны на расчетах. С другой стороны, существующие расчетные методы конструирования, ориентированные на ручное построение, не доведены до полностью формализованных схем, позволяющих на их основе пользователю с любой квалификацией построить требуемую модель. Ориентация этих методик на самые простые инструменты ручного построения (линейка и циркуль) и ручной расчет требуемых для построения расстояний и величин (даже без использования калькулятора) можно отнести к недостатку таких методик при использовании компьютера. Как правило, такие методики подразумевают наличие у пользователя определенного опыта по доводке лекал и посадке изделия на фигуру.

В системе ЛЕКО методики описания и построения лекал должны быть полностью формализованы, вся последовательность построения должна быть четко выражена в терминах системы. Это требует определенной квалификации конструктора, работающего с системой. Мощная вычислительная база системы ЛЕКО предоставляет новые возможности по созданию расчетных алгоритмов построения лекал, обеспечивающих точное соответствие фигуре, согласованность всех лекал изделия между собой, автоматическое размножение лекал на любой размеро-рост.

Для описания и построения лекал моделей швейных изделий в системе ЛЕКО используется простой специализированный язык записи методики описания и построения лекала (язык программирования), позволяющий записывать геометрические построения и выводить результат построения на печать. По аналогии с языками программирования, будем называть правила записи описания построения

моделей швейных изделий синтаксисом и семантикой языка, отдельные элементарные действия - операторами, а текстовое описание алгоритма аналогом программы.

В первых частях описания были приведены основные доводы, обосновывающие использование специализированного языка программирования, и приведены преимущества такого подхода. Но преимущества не являются чем-то абсолютным, независимым от пользователя. *Необходимо изучить все возможности системы и языкового представления описания и построения лекал, а затем на практике реализовать преимущества системы.*

Система проектирования одежды (СПО) ЛЕКО постоянно развивается и дополняется новыми возможностями конструирования, описания лекал, оформления документации. Начиная с версии 6.6, дополнительные возможности и расширения системы ориентированы в первую очередь на профессиональную работу. Это значит, что не только расширяются возможности системы, но и на изучение и освоение этих возможностей необходимо потратить некоторое время. Расширения системы следует рассматривать как «возможности», которые можно реализовать и использовать, а можно оставить как «нереализованные возможности».

Начальные версии были ориентированы на конструктора, незнакомого с вычислительной техникой. Это накладывало ограничения на язык описания и построения лекал, предполагало использование упрощенной терминологии и, естественно, несколько сдерживало развитие системы. В настоящее время основной упор делается на упрощение и повышение эффективности профессиональной работы, что может несколько осложнить начальный этап работы с системой.

Использование системы «в лоб» (например, ввод Единого метода конструирования без изменений и модификации под возможности компьютера) не позволяет использовать весь потенциал системы.

Система в большей степени ориентирована на конструктора, знакомого с основами программирования, или технического специалиста, знакомого с основами конструирования. Возможна совместная работа конструктора и программиста для детальной настройки системы на особенности работы конкретного предприятия.

# 1. ОСНОВНЫЕ ЭЛЕМЕНТЫ ЯЗЫКА

Возможности языка описания и построения лекал выбирались исходя из необходимых при описании и построении моделей швейных изделий действий, геометрических построений и используемых примитивов. Если взять любую методику построения лекала, то первое действие конструктора - установка точки в некоторой системе координат. Затем строятся линии, отрезки, дуги, по заданным условиям проводятся прямые и кривые линии. Любая методика подразумевает набор основополагающих элементов: система координат, единицы измерения, доступный набор инструментов и операций, исходные данные (размерные признаки, задание на конструирование, эскиз).

Основа системы ЛЕКО - вычисление координат опорных точек лекала при помощи арифметических формул, в которых можно использовать значения размерных признаков из базы данных. В формулах можно также использовать: числа, задаваемые константами; переменные, рассчитываемые по формулам; длины, расстояния, величины углов, определяемые между различными элементами. При записи методики построения лекала в качестве геометрических объектов можно использовать:

- *точки, заданные своими координатами;*
- *отрезки, задаваемые двумя точками;*
- *дуги;*
- *сплайны;*
- *ломанные.*

Язык описания построения лекал моделей похож на несложный язык программирования, поэтому позволит любому пользователю быстро освоить написание методик построения лекал. Ниже следует неформальное описание основных элементов и конструкций языка.

## 1.1. КЛЮЧЕВЫЕ СЛОВА

Как и в любом языке программирования, для написания программы на языке описания и построения лекал используются ключевые слова, имеющие в системе свое фиксированное, неизменное назначение. Не разрешается использовать эти слова для обозначения геометрических объектов и переменных (если Вы попробуете это сделать, то система при обработке укажет Вам на такое использование ключевых слов как на ошибку). Приведем несколько примеров ключевых слов:

*отрезок*
*точка*
*сплайн_к*
*симметрия_л*
*перенос*

Названия и назначение ключевых слов будут даны по ходу описания; полный список ключевых слов указан в приложении.

# 1.2. ИДЕНТИФИКАТОР

Все геометрические объекты (величины, точки, отрезки, линии) в программе имеют свой идентификатор (название), однозначно определяющий этот объект. Идентификатор (название) представляет собой последовательность букв, цифр и символа "_", начинающуюся с буквы. Максимальная длина идентификатора 30 символов. Могут использоваться буквы русского и латинского алфавита. Например:

*точка_плеча_1*
*точка_плеча_2*
*нижняя_дуга_проймы*

При записи формул расчета координат в программе могут использоваться величины размерных признаков, выбираемые из базы данных, созданной при помощи системы ЛЕКО (см. описание работы с базой данных). Величины размерных признаков могут использоваться в формулах как переменные-числа и обозначаются в соответствии с порядковыми номерами (например: рз_23, рз_1, рз_56, рз_114) или с присвоенными Вами обозначениями (например: обх1, обх2, талия).

Далее все объекты программы, по аналогии с языками программирования, мы будем называть переменными. Названия переменных в программе, как принято в языках программирования, будем называть идентификаторами. В программе для записи идентификаторов и ключевых слов могут использоваться прописные и строчные буквы, между которыми не делается различий. Могут использоваться буквы русского и латинского алфавитов. При этом буквы "о а с т н у" и др. на русской и латинской клавиатуре считаются различными, поэтому при редактировании текста программы необходимо следить за тем, в каком режиме находится клавиатура: русском или латинском. Если после редактирования текста возникает ошибка "Такого идентификатора нет", то следует проверить режим ввода. Самый простой способ избежать таких ошибок - активно использовать режим копирования идентификаторов и фрагментов текста (Ctrl+Ins, Shift+Ins, см. порядок работы с редактором), словарей и переноса названий точек и линий из графической части экрана (клавиши Alt+Z или Alt+Shift+Z).

# 1.3. ОПРЕДЕЛЕНИЕ ПЕРЕМЕННЫХ

В процессе выполнения программы происходит последовательное определение и построение геометрических объектов и скалярных величин-переменных (обычных чисел). Для определения геометрических объектов и переменных используется знак присваивания ":=". Далее за знаком присваивания следует выражение или ключевое слово, которые определяют тип переменной. Например:

*высота_3:=t20*0.35;*
*т:=точка(1,10);*
*плечо:=отрезок(точка_плеча_2,точка_плеча_1);*

После своего определения (создания) переменная имеет тип и значение, которое можно использовать далее в программе для определения других переменных. Например, запись

*т:=точка(1,10);*

означает, что определена переменная-точка с названием (идентификатором) "т" и имеющая значение - горизонтальную и вертикальную координаты (1,10). А запись

*т:=1.1;*

означает, что определена переменная-число со значением 1.1.

При просмотре изображения Вы можете определить последовательность построения и взаимное расположение определяемых переменных – геометрических объектов.

Прямое присваивание одного объекта другому невозможно (и не имеет смысла давать два названия одному объекту).

Использование переменных очень упрощает написание гибких, настраиваемых алгоритмов. Наглядным примером является использование при конструировании верхней одежды коэффициента прилегания по линии талии.

*прил_тал:=0.2*; { коэффициент прилегания по линии талии }

Этот коэффициент затем учитывается при расчете распределения вытачек и их построении. Величина этого коэффициента несет и смысловое значение. Он определяется максимально возможным раствором вытачек по линии талии ("обхват бедер - обхват талии"). Если коэффициент равен 0, то это прямой силуэт, если же значение коэффициента - 1, то это говорит о том, что при построении вытачек использовался весь раствор вытачек. Промежуточное значение (0 < прил_тал < 1) определяет ширину изделия по линии талии. Для изменения силуэта изделия Вам не нужно теперь возвращаться к построению. Добиться желаемого результата можно, изменяя лишь введенный Вами коэффициент прилегания по линии талии.

Однако здесь хочется отметить, что не стоит воспринимать использование этого коэффициента как аксиому. Это лишь демонстрация возможностей и порядка использования системы.

## 1.4. ЧИСЛА

Самый простой объект программы - числа. Числа могут обозначать расстояния, коэффициенты, угловые величины и т.д. Для определения переменной-числа необходимо записать его идентификатор (название), символ присваивания ":=" и далее формулу или константу, например:

*длина_плеча:=10;*
*высота_3:=t20*0.35;*

В конце формулы ставится разделитель ";" (точка с запятой), завершающий оператор. Разделитель ";" ставится после каждого оператора.

Переменные-числа могут использоваться в формулах и в качестве параметров при обращении к встроенным функциям. Например:

*высота_4:=0.65*длина_плеча;*

*линия_пл:=сплайн_д(о1,о2,д1,д2,к1,длина_плеча);*

Примером использования числовых характеристик может служить и рассмотренный нами случай с коэффициентом приталенности, где величина этого коэффициента несет конструктивное значение.

В формулах можно использовать арифметические функции:

| Обозначение | Функция |
|---|---|
| ABS | абсолютное значение числа |
| ATAN | Арктангенс |
| COS | косинус числа |
| EXP | Экспонента |
| LN | натуральный логарифм |
| ROUND | округление до ближайшего целого |
| SIN | Синус |
| SQRT | квадратный корень |
| SQR | квадрат числа |
| TRUNC | выделение целой части |

# 1.5. КООРДИНАТНЫЕ ОСИ

Для задания координат геометрических объектов в системе определена прямоугольная система координат: точка (0,0) - начало координат, ось X - горизонтальная ось, направленная слева направо (первая координата), ось Y - вертикальная ось, направленная сверху вниз (вторая координата). Такое направление осей выбрано в связи с тем, что конструирование, как правило, ведется сверху вниз, и выбранное направление оси Y обеспечивает работу с положительными приращениями координат. Единица измерения по осям - 1 сантиметр.

Положительное направление отсчета углов ведется традиционно - от оси X к оси Y, и за счет изменения направления оси У положительное направление изменения углов направлено по часовой стрелке. Это необходимо учитывать при использовании угловых величин в построении.

# 1.6. ТОЧКИ

Основной элемент любых геометрических построений - точка. Переменная-точка задается двумя координатами: по оси X (горизонтальная) и оси Y (вертикальная). Для определения переменной-точки используется ключевое слово "точка" и две координаты, записанные через запятую в круглых скобках:

*идентификатор точки:=точка(X,Y);*

где *X* и *Y* - любые арифметические выражения или квалификаторы.

Координаты *X* и *Y* могут задаваться константами и формулами, в которых могут использоваться различные, определенные ранее, переменные. Например:

*точка_плеча_2:=точка(10,20);*
*точка_плеча_1:=точка(Ширина_гор+0.5,-Ширина_гор/3);*

Переменные-точки могут определяться не только путем задания координат, но и как пересечение линий, откладываемые расстояния, построение нормали (перпендикуляра) к прямой и т.д. Но задание координат точек - самый универсальный способ установки точки, при помощи которого можно провести практически любое построение.

Приведем пример построения основных опорных точек спинки.

*а0:=точка( 0, 0);*
*л:=точка( а0.х, а0.у+19+0.25*(рост-164)); {уровень лопаток}*
*т:=точка( а0.х, а0.у+Рз_40+Пдтс ); {уровень талии}*
*б:=точка( т.х, т.у+19+0.25*(рост-164)); {уровень бедер}*
*г:=точка( т.х, т.у-(Рз_39-Пспр)); {уровень груди}*
*а2:=точка( а0.х+0.33*Сш+Пшр, а0.у); {определение ширины ростка}*
*а:=точка( а0.х, а0.у+0.35*(0.33*Сш+Пшр)); {линия основания ростка}*
*а11:=точка(а0.х+0.3*(0.33*Сш+Пшр), 0.у+0.35*(0.33*Сш+Пшр));*
*{промежуточная точка при оформлении линии ростка}*
*н:=точка( а.х, а.у+Ди+Пди); {уровень низа}*
*г1:=точка( г.х+г_г1, г.у); {ширина спинки}*
*б1:=точка( г1.х, б.у); {уровень бедер}*

На этом небольшом фрагменте программы хорошо видно использование элемента построения лекал ТОЧКА и размерных признаков. Добавление к идентификатору (названию) точки квалификаторов ".х" ".у" дает соответственно значение горизонтальной и вертикальной координаты точки. К примеру, запись

*б1:=точка( г1.х, б.у); {уровень бедер}*

означает, что горизонтальная координата точки б1 такая же, как у точки г1, а вертикальная координата – как у точки б.

Отметим еще одну особенность приведенного выше фрагмента. При построении практически не используются числовые значения, связанные с измерениями. Эти значения (Рз_39, Рз_40, рост и т.д.) берутся из базы данных размерных признаков и подставляются в формулы. При записи формулы "т:=точка( а0.х, а0.у+Рз_40+Пдтс ); {уровень талии}" можно не знать конкретного значения РЗ_40, но можно быть точно уверенным, что построенная точка будет находиться на линии талии при построении на любую типовую или индивидуальную фигуру.

Сделаем небольшое отступление по поводу построения точки "б":

*б:=точка( т.х, т.у+19+0.25*(рост-164)); {уровень бедер}*

Т.к. в программе измерений типовых фигур не производится замер положения линии бедер, то в каждой методике используются свои зависимости для ее определения. В данном случае выбрана следующая зависимость: для роста 164 точка "б" находится на 19 сантиметров ниже линии талии; при уменьшении или увеличении роста на 1 сантиметр точка "б" приближается или удаляется на 0.25 см к линии талии.

Как видно из приведенного примера, переменные-точки можно устанавливать относительно начала координат (а0:=точка( 0, 0);) или относительно других точек:

*л:=точка( а0.х, а0.у+19+0.25*(рост-164)); {уровень лопаток}*

(откладывание расстояния 19+0.25*(рост-164) по вертикали вниз от точки "а0")

*г1:=точка( г.х+г_г1, г.у); {ширина спинки}*

(откладывание расстояния "г_г1" по горизонтали вправо от точки "г")

*б1:=точка( г1.х, б.у); {уровень бедер}*

(пересечение вертикальной линии из точки "г1" и горизонтальной линии из точки "б").

При необходимости возможны расчетные формулы любой степени сложности с использованием тригонометрических соотношений.

## 1.7. ОТРЕЗКИ

Переменные-отрезки используются для построения других геометрических объектов, для отображения контуров лекала и внутренних деталей. Переменная-отрезок определяется при помощи ключевого слова "отрезок" и идентификаторов или определений двух точек:

*идентификатор отрезка:=отрезок(Т1,Т2);*

где *Т1* и *Т2* - идентификаторы точек, определяющих концы отрезка. Например:

*плечо:=отрезок(точка_плеча_2,точка_плеча_1);*
*плечо1:=отрезок(точка(10,20),точка_плеча_1);*
*плечо2:=отрезок(точка(10,20),точка(3,Шп+1));*

Отрезок имеет направление: от первой точки ко второй. Это необходимо учитывать при описании контура лекала, ломанных или внутренних линий.

Чаще всего отрезки используются для промежуточных построений и оформления контуров лекал. Например, определив основные точки плеча спинки (точ3 - точка ширины ростка, точ4, точ6 - точки вытачки на плече, точ5 - точка начала плечевой вытачки, точ7 - наивысшая плечевая точка), можно построить линию плеча:

*плечо1:=отрезок(точ3, точ4);*
*плечо2:=отрезок(точ4, точ5);*
*плечо3:=отрезок(точ5, точ6);*
*плечо4:=отрезок(точ6, точ7);*

Отрезки *плечо1* и *плечо4* - отрезки плечевого среза, а *плечо2* и *плечо3* - оформление вытачки спинки.

Иногда нужно чтобы отрезок состоял из нескольких участков (например, при разведении). Для этого можно указать количество точек используемых для построения линии:

*плечо1:=отрезок[10](точ3, точ4);*

Вместо конкретного числа 10 можно использовать числовую переменную.

В современных построениях отрезки, как отдельные переменные, используются достаточно редко. Для записи лекал достаточно использовать точки, для расчета и задания направлений или расчета расстояний можно использовать неявные отрезки, о чем будет сказано ниже.

# 1.8. ДУГИ

Переменные-дуги, как и отрезки, предназначены, как правило, для промежуточных построений, отображения контуров лекала и внутренних деталей. Дуга определяется точкой центра окружности, радиусом, начальным и конечным углами. Для определения переменной-дуги используется ключевое слово "дуга":

*идентификатор дуги:=дуга(центр, радиус, начальный угол, конечный угол);*

где *центр* - идентификатор точки, являющейся центром окружности,

*радиус* - арифметическое выражение или число, определяющее радиус,

*начальный угол, конеч6ный угол* - арифметические выражения или числа, определяющие начальный и конечный углы дуги, град.

*плечо:=дуга(точка_плеча_2,радиус,нач_угол,кон_угол);*

Точка - центр окружности может быть задана идентификатором точки или определением точки с заданием координат. В качестве радиуса, начального и конечного угла могут использоваться константы или формулы.

*дуга1:=дуга(точка(10,10), 10, 0, -30);*

Дуга имеет направление: от первой точки, соответствующей первому углу, ко второй точке, соответствующей второму углу.

Использовать дуги для построения кривых линий лекал следует только тогда, когда требуется не просто плавное сопряжение, а именно дуга. Например, это необходимо для деталей, получаемых методом высечки, формы для которых изгибаются на круглых трубах (кожгалантерея). В тех случаях, когда требуется только плавное сопряжение линий, рекомендуется использовать для построения кривых линий сплайны.

# 1.9. СПЛАЙНЫ

Очень часто при построении лекала требуется соединить две точки плавной кривой. Это могут быть случаи оформления линии проймы, ростка, линии низа или каких-либо декоративных деталей. Использовать для этих целей дугу часто достаточно сложно, да и не требуется: в системе предусмотрен оператор построения сплайна.

Сплайн - это плавная кривая линия, проходящая через заданные точки и удовлетворяющая дополнительным условиям. При большом разнообразии различных видов сплайнов в системе выбраны сплайны, задаваемые начальной и конечной

точками и заданными углами касательных к сплайну в его начальной и конечной точках, для согласования сплайна с другими объектами. Дополнительно при построении сплайна можно удовлетворить некоторым условиям, например, построить сплайн заданной длины.

Основной способ построения сплайна - путем задания коэффициента, характеризующего выпуклость сплайна. Для такого определения переменной-сплайна используются: начальная точка, конечная точка, угол касательной в начальной точке, угол касательной в конечной точке и коэффициент. Кривая строится таким образом, чтобы она проходила через две точки и касательные в начальной и конечной точках к ней совпадали с требуемыми направлениями. При этом дополнительно задается коэффициент, определяющий выпуклость кривой и приближенность сплайна к своим касательным (заданным направлениям) на концах. Коэффициент может меняться от 0 до любого большого положительного числа:

*идентификатор сплайна:=сплайн_к(Т1, Т2, кас1, кас2, К);*

где *Т1* и *Т2* - идентификаторы двух крайних точек сплайна,

*кас1, кас2* - арифметические выражения или числа, определяющие углы наклона касательных на концах сплайна,

*К* - коэффициент выпуклости.

Например:

```
к1:=1.5;
линия_пл:=сплайн_к(о1,о2,д1,д2,к1);
```

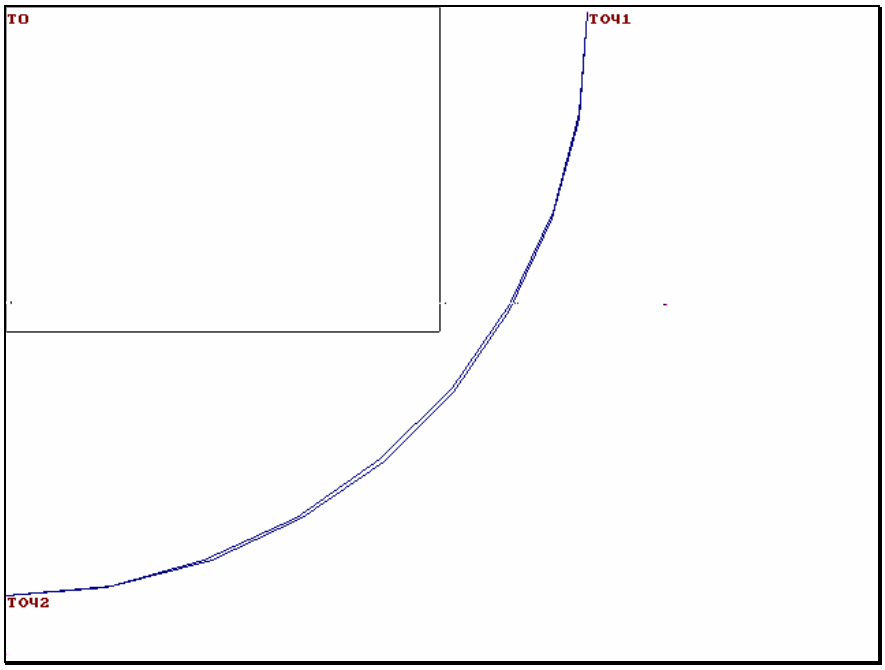
или

```
сплайн_к(о1,о2,д1,д2,к1);
```

Сплайну можно и не присваивать идентификатор. В таком случае система сама присвоит идентификатор по двум точкам сплайна – с_о1_о2.

Значение коэффициента 1.2 обеспечивает построение такой кривой, которая при аппроксимации дуги в четверть круга на взгляд неотличима от этой дуги. Например:

```
т0:=точка( 0, 0);
т1:=точка( 10, 0);
т2:=точка( 0, 10);
д1:=дуга( т0, 10, 0, 90);
с1:=сплайн_к( т1, т2, 90, 180, 1);
```




Значение коэффициента 0 приводит к тому, что кривая линия вырождается в отрезок. При увеличении значения коэффициента от нуля до единицы получающиеся сплайны будут располагаться между отрезком и "единичным" сплайном. При значениях коэффициента больше единицы сплайн становится все более выпуклым, пока не превратится в линию с самопересечениями (коэфф. больше 3-4). Для конструирования имеет смысл использовать сплайны с коэффициентами от 0.5 до 1.5, которые дают достаточно выпуклые, гладкие и красивые линии. Например, Вы можете построить "веера" сплайнов при помощи следующей программы:

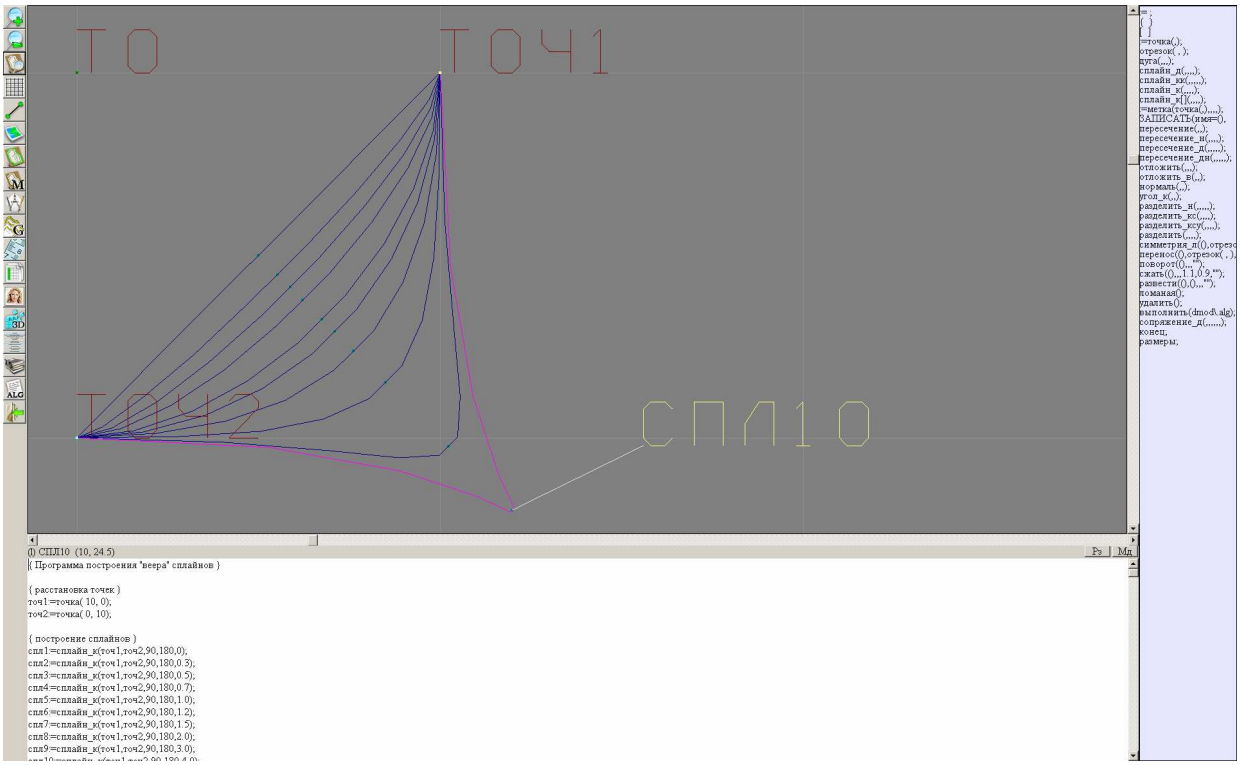
```
{ Программа построения "веера" сплайнов }

{ расстановка точек }
точ1:=точка( 10, 0);
точ2:=точка( 0, 10);

{ построение сплайнов }
спл1:=сплайн_к(точ1,точ2,90,180,0);
спл2:=сплайн_к(точ1,точ2,90,180,0.3);
спл3:=сплайн_к(точ1,точ2,90,180,0.5);
спл4:=сплайн_к(точ1,точ2,90,180,0.7);
спл5:=сплайн_к(точ1,точ2,90,180,1.0);
спл6:=сплайн_к(точ1,точ2,90,180,1.2);
спл7:=сплайн_к(точ1,точ2,90,180,1.5);
спл8:=сплайн_к(точ1,точ2,90,180,2.0);
спл9:=сплайн_к(точ1,точ2,90,180,3.0);
спл10:=сплайн_к(точ1,точ2,90,180,4.0);

конец
```

После построения при просмотре система покажет набор кривых. Заметим, что касательные в начальной и конечной точке у всех кривых совпадают.




Для решения некоторых задач может потребоваться более гибкое задание формы сплайна, т.е. помимо управления "выпуклостью" сплайна необходимо влиять и на "симметричность" линии сплайна. Для этого может использоваться оператор "сплайн_кк", в котором помимо коэффициента выпуклости сплайна (к1) задается коэффициент асимметрии сплайна (к2). При коэффициенте асимметрии равном 1 "сплайн_кк" совпадает со "сплайн_к". При изменении коэффициента асимметрии "сплайн_кк" будет более сильно прилегать к линии касательной в первой или во второй точке, в зависимости от величины коэффициента.

*идентификатор сплайна:=сплайн_кк(Т1, Т2, кас1, кас2, К1, К2);*

где *Т1* и *Т2* - идентификаторы двух крайних точек сплайна,
*кас1, кас2* - арифметические выражения или числа, определяющие углы наклона касательных на концах сплайна,
*К1* - коэффициент выпуклости сплайна,
*К2* - коэффициент асимметрии сплайна.

*линия_пл:=сплайн_кк(о1,о2,д1,д2,к1,к2);*

Сплайн - достаточно универсальный инструмент для конструирования лекал. С его помощью можно изобразить и другие элементы построения: отрезки, дуги. Но, в отличие от дуг, сплайны обладают гораздо большей гибкостью и простотой использования. Сплайн в системе ЛЕКО имеет дополнительные степени свободы, позволяющие удовлетворять дополнительным условиям. Например, это обеспечивает возможность построения "веера" сплайнов для нижней части проймы, построенных с

различными коэффициентами. Все они имеют вертикальную касательную в первой точке и горизонтальную во второй. При построении проймы конструктор сам выбирает коэффициенты, дающие, с его точки зрения, наиболее приемлемый вид проймы. В других случаях свобода выбора коэффициента сплайна уже недопустима - например, при построении части оката соответствующей определенной части проймы.

## 1.10. ГЛАДКОСТЬ КРИВЫХ ПРИ ПОСТРОЕНИИ

Все криволинейные участки в компьютере задаются отрезками. В зависимости от выполняемых работ может потребоваться различное количество отрезков, описывающих кривую.

При построении дуг и сплайнов после определяющего их ключевого слова может стоять число, заключенное в скобки [ ]. Это дополнительная информация, которая указывает число точек, по которым строится кривая. Записывается это в следующем виде:

*сплайн1:=сплайн_к[20](о1,о2,10,20,1);*
*дуга1:=дуга[20](о3, 10, 0, -30);*

Т.е. в построении данного сплайна и дуги использовалось по 20 промежуточных точек. Если же эта информация не указана, то по умолчанию берется 10 точек. При отладке программ можно не указывать число точек. Это делается для ускорения обработки алгоритмов конструкций. Если же Вы убеждены в правильности построения и Вам необходимо получить по возможности наиболее гладкую кривую, то можно указать эту информацию.

## 1.11. ЛОМАНЫЕ

В любой вычислительной машине все объекты представлены числами. Геометрические объекты сложной формы, как правило, описываются набором отрезков. Сплайны и дуги в системе ЛЕКО также состоят из набора отрезков. Для работы со сложными кривыми, состоящими из нескольких участков, в системе используется понятие ломаной. В ломаную (набор отрезков) можно объединить несколько объектов, образующих вместе одну непрерывную кривую линию:

*идентификатор ломаной:=ломаная(A1, A2, A3, A4,.....);*

где *A1, A2, A3,....* - список объектов, объединяемых в ломаную - идентификаторы точек, отрезков, сплайнов, дуг.

Например:

*с1:=сплайн_к( m7, m8, [m6:m7].ф1, 90, 1);*
*с2:=сплайн_к( m8, m9, 90, 0, 1);*
*пройма_спинки:=ломаная(с1,с2);*

После такой записи можно забыть, что линия проймы состоит из двух участков, и пользоваться единой непрерывной линией "пройма_спинки". При объединении объектов в ломаную необходимо помнить, что все переменные, кроме переменных-точек, имеют направление. Если требуется включить переменную с обратным направлением обхода точек, то нужно поставить знак минуса перед переменной.

*пройма_спинки:=ломаная(с1,-с2);*

Во многих случаях использование ломаной позволяет упростить построение и сократить запись.

# 2. ОПЕРАТОРЫ ПРОГРАММЫ

Программа на языке описания и построения моделей состоит из операторов, последовательно определяющих переменные или выполняющих определенные действия (например, направление на печать). Все операторы отделяются друг от друга разделителем ";" (точка с запятой). Один оператор может быть записан на нескольких строках, на одной строке могут быть записаны несколько операторов. Рекомендуем не экономить строки и записывать каждый оператор на одной или нескольких строках, разбивать пустыми строками и комментариями группы логически взаимосвязанных операторов.

Все операторы выполняются последовательно, если нет явного указания об изменении последовательности выполнения операторов.

При работе программы происходит последовательное определение координат и параметров переменных.

Прямое определение переменных производится при помощи следующих ключевых слов:

*ТОЧКА*
*ОТРЕЗОК*
*ДУГА*
*СПЛАЙН*
*ЛОМАНАЯ*

Возможно не прямое определение переменных, а построение их при помощи встроенных функций (определение точек через пересечение объектов, перенос или поворот объектов).

Для каждого типа переменных определен набор допустимых действий: пересечение, параллельный перенос, центральная и осевая симметрия, поворот, вывод на печать. Практически любое из этих преобразований можно реализовать также при помощи формул расчета координат, но использование встроенных функций позволяет упростить запись и повысить ее наглядность.

Программа состоит из последовательного определения новых геометрических объектов - переменных, координаты которых рассчитываются по формулам или определяются при помощи преобразований.

При выполнении программы система последовательно просматривает операторы и выполняет определяемые ими действия. После просмотра всей

программы система формирует и записывает изображения лекал для просмотра на экране и вывода на печать, а так же координаты геометрических объектов.

В целом операции предлагаемого языка соответствуют операциям, выполняемым при построении чертежей вручную. Это соответствие и возможность многократного быстрого "перепостроения" чертежа делают систему удобной в использовании, а формализм построения и возможность использования базы данных с хранимыми значениями делают возможным использование системы как действительно серьезного автоматизированного рабочего места.

# 3. КВАЛИФИКАТОРЫ

В формулах Вы можете использовать, помимо констант и переменных - размерных признаков, параметры других геометрических объектов. Самое простое - это использование координат точек. Например,

*точка_плеча_1:= точка(m2.x+0.5,m2.y+5);*

Это значит, что *точка_плеча_1* имеет горизонтальную координату, смещенную на 0.5 см вправо относительно точки *m2,* а вертикальная координата смещена на 5 см вниз относительно точки *m2*.

Т.е. при ссылке на переменную-точку Вы можете ссылаться на нее как на целый геометрический объект (при построении отрезков, линий) и можете ссылаться на ее координаты, поставив за идентификатором точки символ "." и дописав название координаты X или Y. Это позволяет откладывать точки относительно заданной переменной - точки по горизонтали, вертикали и вдоль направления, заданного другой точкой. Например, если задана точка тчк_1. то точка

*тчк_1:=точка( 1, 1);*
*тчк_2:=точка(тчк_1.x+10, тчк_1.y);*

сдвинута на 10 см по горизонтали, точка

*тчк_3:=точка(тчк_1.x, тчк_1.y+10);*

сдвинута на 10 см по вертикали, точка

*тчк_3:=точка(тчк_1.x+тчк_4.x*2, тчк_1.y+тчк_4.y*2);*

сдвинута относительно тчк_1 в направлении (0,тчк_4) на удвоенное расстояние |т0:тчк_4|.

Перечислим квалификаторы, которые можно использовать при записи формул для различных переменных:

| Переменные | Квалификатор | Значение |
|---|---|---|
| Точки | | |
| | X | координата точки по оси X |

| | | |
|---|---|---|
| | Y | координата точки по оси Y |
| Отрезки, Дуги | | |
| | Т1 | первая точка |
| | Т2 | вторая точка |
| | X1 | координата первой точки по оси X |
| | Y1 | координата первой точки по оси Y |
| | X2 | координата второй точки по оси X |
| | Y2 | координата второй точки по оси Y |
| | dx | разница между координатами начальной и конечной точек по оси X |
| | dy | разница между координатами начальной и конечной точек по оси Y |
| | Ф1 | угол наклона направленного отрезка |
| | Ф2 | Угол наклона направленного отрезка |
| | Л | Длина |
| Сплайны, Дуги, Ломаные | | |
| | т1 | первая точка |
| | т2 | вторая точка |
| | X1 | координата первой точки по оси X |
| | Y1 | координата первой точки по оси Y |
| | X2 | координата второй точки по оси X |
| | y2 | координата второй точки по оси Y |
| | Ф1 | угол наклона касательной в первой точке |
| | Ф2 | угол наклона касательной во второй точке |
| | К | коэффициент сплайна |
| | Л | Длина |
| Сплайны, Дуги | | |
| | К | коэффициент сплайна |

Очень часто при построении в расчетах используется такая величина, как длина сплайна. Конструктору не требуется самому рассчитывать это значение, более того, ему не требуется вообще знать это значение, необходимо лишь при построении после названия сплайна поставить квалификатор *л*.

*спл1:=сплайн_к(о1,о2,д1,д2,1);*
*спл2:=сплайн_к(о3,о4,д3,д4,0.8);*
*отложить_в(спл2,спл1.л,т10);*

В данном случае система отложит на сплайне *спл2* длину сплайна *спл1*.

# 4. УГЛЫ

Величины углов - это просто переменные-числа, однако они рассматриваются отдельно из-за того, что имеют конкретную геометрическую интерпретацию и свою размерность. В системе ЛЕКО величины углов измеряются в градусах.

При построении лекал на многие участки накладывается требование по согласованию углов. Эти требования могут формироваться или для получения прямой

линии при соединении лекала, или из геометрических построений (перпендикуляры, биссектрисы), или из требования пространственного расположения прямых. Ввиду важности угловой информации в системе предусмотрена гибкая возможность получения угловой информации при помощи квалификаторов, о чем сказано ниже.

Как уже отмечалось, при построении лекал широко используется угловая информация. Может быть, это кажется не таким очевидным, но конструктор почти на каждом шагу построения неявно учитывает эти величины. Например, при проведении перпендикуляра, касательной, параллельной прямой часто возникает необходимость проверки сопряжения углов. В системе ЛЕКО, как правило, не требуется вычислять эти углы, Вы можете и не знать их абсолютного значения. Для использования угловых величин при построении можно воспользоваться квалификатором.

Рассмотрим пример построения нижнего среза изделия. Вы уже предварительно построили боковые срезы на полочке и спинке (бок_пол, бок_сп), а также средние линии на полочке и спинке (сер_бок, сер_сп). Здесь следует помнить, что бок_пол, бок_сп, сер_пол и сер_сп - это не вся кривая, определяющая боковые срезы или середину переда или спинки, а лишь конечные звенья, которые подходят к нижнему срезу изделия. Нижний срез строится исходя из фасона изделия, но наиболее часто используется строго горизонтальный, перпендикулярный к боковым швам, для того, чтобы не было излома линии при соединении деталей. Опишем это.

Пусть бок_пол, бок_сп, сер_пол и сер_сп - отрезки, а конечные точки этих звеньев (точ1, точ2, точ3, точ4) участвуют в построении низа. Тогда нижний срез полочки, проходящий через точки точ1 и точ2, можно будет представить одной строкой:

*низ_пол:=сплайн_к(точ1,точ2, сер_пол.ф1+90,бок_пол.ф1+90,1);*

Записи *сер_пол.ф1+90* и *бок_пол.ф1+90* означают, что линия низа перпендикулярна середине полочки и перпендикулярна боковому срезу полочки.

Если же боковой срез представляет собой кривую (бок_пол - сплайн), то разница в записи будет лишь в том, что квалификатор укажет начальную или конечную точку сплайна, в зависимости от того, какая точка участвует в построении низа:

*низ_пол:=сплайн_к(точ1,точ2, сер_пол.ф1+90,бок_пол.ф2+90,1);*

Аналогично строится и нижний срез на спинке. Однако Вы можете заложить в построение и другое условие, а именно: линия низа в области бокового шва должна быть гладкой, т.е. должно выполняться сопряжение углов. Тогда это требование можно записать в виде:

*угол_бок_пол:=бок_пол.ф1-низ_пол.ф2;*
*з_сп:=сплайн_к(точ3,точ4, сер_сп.ф+90,бок_сп.ф2+(180-угол_бок_пол),1);*

где *угол_бок_пол* - угол, который составляет нижний срез с боковым на полочке.

Такое условие удобно при внесении изменений в фасон. К примеру, Вы захотели сделать линию низа на полочке короче, чем на спинке - так называемый эффект шлейфа. Тогда Вам потребуется изменить лишь угол, который составляет нижний срез с боковым на полочке:

*низ_пол:=сплайн_к(точ1,точ2, сер_пол.ф1+90,бок_пол.ф1+45,1);*

Если же Вы заложили условие сопряжения углов, то система сама перестроит нижний срез на спинке.

# 5. НЕЯВНОЕ ЗАДАНИЕ ОТРЕЗКОВ

Построение переменных-отрезков используется для промежуточных операций и для описания контура лекала. Когда отображение отрезка на экране не обязательно, можно использовать неявное задание отрезков (без их построения и задания идентификатора) прямо в формулах. Для этого через символ ":" записываются названия точек в квадратных скобках, и затем - требуемый квалификатор, например:

*[точка1:точка2].л*
*[точка1:точка2].ф1*

Неявное задание отрезка в некоторых случаях может быть удобнее, чем использование идентификатора отрезка, т.к. отрезок с идентификатором имеет и заданное направление от первой точки ко второй, и в формулах, использующих направление отрезка, необходимо помнить, как определялся отрезок, а также его идентификатор.

При использовании в формуле длины отрезка или расстояния между точками можно воспользоваться записью:

|*точка1:точка2*|

Эта запись идентична записи [точка1:точка2].л. Оставляем Вам право выбора наиболее удобной и привычной.

Можно проиллюстрировать использование этих записей.

*отложить(а2, [а2:п].ф1, 0.3*Шп, и);*

Здесь используется неявное задание отрезка для определения направления. Производится построение плечевого среза. В команде "Отложить" можно использовать неявное определение длины отрезка:

*отложить(а20, [а20:п10].ф1, [а2:и].л, и10);*

В данном случае строится ответный плечевой срез на смежной детали. Одновременно закладывается соответствие длин плечевых срезов на двух деталях.

# 6. КОММЕНТАРИЙ

Текст, заключенный в фигурные скобки, воспринимается системой при построении как комментарий и никак не влияет на построение лекала. Комментарий может содержать любой текст, поясняющий выполняемые действия, дающий ссылки на литературу или вообще не относящийся к построению. Например:

*{ Это комментарий }*

*{ Это тоже комментарий ч1:=10; }*

Комментарии - необходимая часть любой программы. Если Вы хотите написать действительно хороший алгоритм построения лекала, то в описании обязательно должны быть комментарии, поясняющие действия, производимые в программе и дополняющие и поясняющие то, что осталось за рамкой программы, - ограничения, допущения. Правила написания программы позволяют записывать многие действия достаточно компактно, и для того, чтобы при просмотре программы сразу был ясен смысл выполняемых действий, чтобы можно было сразу отыскать в программе место, где необходимо сделать коррекцию или исправления, комментарии должны разбивать программу на части. В комментарий Вы можете записывать ссылки на источник информации, свои авторские реквизиты.

Пример использования комментариев в программе:

```
{ -------построение проймы спинки------- }
п2:=точка( а_.х, п1.у);
п3:=точка( г1.х, г1.у-|п2:г1|/3);
{ Возможно смещение п3 вверх: п3:=точка( г1.х, г1.у-(|п2:г1|/3+2)); }
г2:=точка( г1.х+|г1:г4|/2, г1.у);
пр_г01:=0.7*осанка-0.35; { зависимость припуска при построении точки г01 от осанки }
```

Легко заметить, что при таком написании алгоритмов можно без труда отследить методику построения, внести при необходимости какие-либо изменения.

# 7. ПОДКЛЮЧЕНИЕ ТАБЛИЦЫ РАЗМЕРНЫХ ПРИЗНАКОВ

Для подключения таблицы размерных признаков используется оператор с ключевым словом "РАЗМЕРЫ". Встретив этот оператор, система считывает таблицу текущих, ранее выбранных пользователем, размерных признаков для типовой или индивидуальной фигуры. Система заносит значения размерных признаков в список текущих переменных под обозначениями, заданными в подсистеме работы с базой данных размерных признаков.

Если Вы не используете таблицу размерных признаков типовых фигур, например, при построении лекал по индивидуальным размерным признакам, то можно определить размерные признаки непосредственно в программе, используя оператор присваивания. Например:

```
Рз_1:=164;
Рз_18:=73;
Рз_19:=100;
Ди:=60;
Псб:=1.5;
Пди:=1;
{-------------------------------------------------}
{    построение заднего полотнища    }
```

```
{-------------------------------------------------}
.................
```

При задании индивидуальных размерных признаков непосредственно в программе советуем использовать те же обозначения, что используются в базе данных размерных признаков. Это обеспечит Вам возможность быстрого перехода от размерных признаков, определенных в программе к размерным признакам типовых фигур и наоборот. Для приведенного примера такой переход может быть сделан следующим образом:

```
{ рз_1:=164; рз_18:=73; рз_19:=100; }
РАЗМЕРЫ;
Ди:=60;
Псб:=1.5;
Пди:=1;
{-------------------------------------}
{    построение заднего полотнища    }
{-------------------------------------}
```

Возможности обозначения размерных признаков в различных версиях системы ЛЕКО различаются. В настоящий момент в WINDOWS версиях осталось только обозначение через порядковый номер. Но при чтении размеров из текстовых файлов могут передаваться любые переменные числового или строкового формата.

# 8. ВСТРОЕННЫЕ ФУНКЦИИ ЯЗЫКА

По аналогии с языками программирования будем называть встроенными функциями дополнительные действия и операции над геометрическими объектами-переменными, которые можно использовать при написании программы.

Описанные выше основные элементы языка позволяют посредством расчета координат выполнить практически любое геометрическое преобразование, однако в некоторых случаях формулы такого преобразования будут выглядеть слишком громоздко и непонятно. Для улучшения и повышения наглядности записи некоторых преобразований, упрощения написания программ в язык системы введены встроенные функции. Встроенные функции позволяют выполнять наиболее часто встречающиеся преобразования над списками переменных (фрагментами контуров деталей).

## 8.1. СИММЕТРИЯ

Рассмотрим сначала одно из основных преобразований, используемых при построении лекал, - симметрию. Используется обычно два вида симметрии: центральная и осевая. Для записи оператора преобразования используются ключевые слова «*СИММЕТРИЯ_О» или «СИММЕТРИЯ_Л» з*а ними в круглых скобках записываются параметры. В качестве первого параметра в скобках записывается список переменных, подлежащих преобразованию. Далее следует идентификатор

точки, линии или отрезка, относительно которых проводится симметрия. Затем записывается список названий преобразованных переменных.

*СИММЕТРИЯ_Л((об1,об2,об3,...,об$N$),отрезок, (об1_п,об2_п,об3_п,...,об$N$_п));*

где *об1,об2,об3,...,об$N$* - список имен графических примитивов, которые будут симметрично отражаться,

*отрезок* - отрезок, который определяет ось симметрии, может быть задан неявно,

*об1_п,об2_п,об3_п,...,об$N$_п* - новые идентификаторы объектов после отражения.

Например:

*СИММЕТРИЯ_Л((т11,т2,о34_56),л_спины, (т11_п,т2_п,о34_56_п));*

Для повышения наглядности Вы можете записать этот оператор в две строчки, записывая идентификаторы преобразуемых переменных над их образами:

*СИММЕТРИЯ_Л((т11, т23, о34_56 ),л_спины, (т11_п, т23_п, о34_56_п));*

Если Вы используете для преобразованных переменных одинаковые изменения их идентификаторов (как в приведенном примере: дописывание строки "_п" в конце идентификатора), то можете упростить запись, указав вместо второго списка подстроку, которую необходимо добавить в конце идентификатора переменной. Например:

*СИММЕТРИЯ_Л((т11, т23, о34_56 ),л_спины,"_п");*

Добавляемую к названию подстроку необходимо заключить в кавычки - одинарные или двойные. Пользуясь возможностью добавления строки, следует помнить, что максимальная длина идентификатора 20 символов, и если при добавлении строки длина будет больше 20 символов, то возможно усечение названия новой переменной или появления ошибки "Такой идентификатор уже есть". Т.е. если Вы выполняете преобразование переменных с длинными идентификаторами, следует явно указывать идентификаторы новых переменных после преобразования

Симметрией удобно пользоваться при моделировании воротника, когда он конструируется в "отложенном" виде, а затем отображается относительно линии сгиба.

Часто при моделировании требуется полное симметричное лекало. Вы можете получить развернутый чертеж, к примеру, полочки, отобразив ее контур относительно середины полочки:

*симметрия_л((пл1л,пл2л,пл3л,пл4л,пл5л,пл6л,пл7л,пл8л,т4л,н4л),пл8л, (пл1п,пл2п,пл3п,пл4п,пл5п,пл6п,пл7п,пл8п,т4п,н4п));*

где

*(пл1л,пл2л,пл3л,пл4л,пл5л,пл6л,пл7л,пл8л,т4л,н4л)* - контур полочки,

*пл8л* - линия середины полочки,

*(пл1п,пл2п,пл3п,пл4п,пл5п,пл6п,пл7п,пл8п,т4п,н4п)* - отображенный контур.

Если необходимо симметрично отобразить деталь так, чтобы не менялись названия точек, то

*симметрия_л((об1,об2,об3,...,об$N$),отрезок, "");* При этом старая деталь стирается.

Центральная симметрия:

*СИММЕТРИЯ_О((об1,об2,об3,...,об$N$),точка, (об1_п,об2_п,об3_п,...,об$N$_п));*

где *об1,об2,об3,...,об$N$* - список имен графических примитивов, которые будут симметрично отражаться,

*точка* - идентификатор точки, являющейся центром симметрии,

*об1_п,об2_п,об3_п,...,об$N$_п* - новые идентификаторы объектов после отражения.

Иногда моделирование удобнее проводить не на основном лекале, а на перенесенной в новое место копии лекала. Для этого используется функция "перенос".

## 8.2. ПЕРЕНОС

Следующее преобразование - параллельный перенос. Для записи преобразования используется ключевое слово «*перенос*» и далее записываются параметры, как и для симметрии. Для задания вектора параллельного переноса могут использоваться отрезок или точка.

*ПЕРЕНОС((об1,об2,об3,...,об$N$),отрезок, (об1_п,об2_п,об3_п,...,об$N$_п));*

где *об1,об2,об3,...,об$N$* - список имен графических примитивов, которые нужно параллельно перенести,

*отрезок* - идентификатор точки или отрезка, определяющих вектор переноса. Если в качестве параметра переноса указывается точка, то перенос осуществляется на вектор, задаваемый началом координат и этой точкой; если указан отрезок - то вектор, задаваемый отрезком.

*об1_п,об2_п,об3_п,...,об$N$_п* - новые идентификаторы объектов после переноса.

Например:

*ПЕРЕНОС((плечо,пройма),о_переноса,(плечо_к,пройма_к));*

В этом случае переменные "плечо" и "пройма" (например, отрезок и сплайн) будут перенесены в направлении вектора, задаваемого отрезком "о_переноса", и их копии будут названы "плечо_к" и "пройма_к" соответственно.

Если же Вы предпочитаете производить моделирование (предыдущий пример) имея перед глазами основу, то можно перенести ее на другое место:

*перенос((пл1,пл2,пл3,пл4,пл5,пл6,пл7,пл8,н5,н4,г0,т4),а3с,"л");*
*симметрия_л((пл1л,пл2л,пл3л,пл4л,пл5л,пл6л,пл7л,пл8л,т4л,н4л),пл8л,*
*(пл1п,пл2п,пл3п,пл4п,пл5п,пл6п,пл7п,пл8п,т4п,н4п));*

Здесь перенесенные звенья отличаются от исходного лекала буквой "л" в имени.

# 8.3. ПОВОРОТ

Следующее важное преобразование - поворот вокруг точки. Для записи преобразования используется ключевое слово «*поворот*» и далее записываются параметры. Для задания поворота необходимы центр и угол поворота.

*ПОВОРОТ((об1,об2,об3,...,об*N*), точка, угол, (об1_п,об2_п,об3_п,...,об*N*_п));*

где *об1,об2,об3,...,об*N - список имен графических примитивов, которые нужно повернуть,
*точка* - идентификатор точки центра поворота,
*угол* - угол поворота,
*об1_п,об2_п,об3_п,...,об*N*_п* - новые идентификаторы объектов после поворота.

Например:

*ПОВОРОТ((плечо,пройма),точка_поворота,угол_поворота,(плечо_к,пройма_к))
;*

Поворот можно использовать при построении и переносе вытачек:

*поворот((точ100, точ11, точ16, точ15, с_точ100точ12, с_точ10_точ11, с_точ15_точ16, о_точ16_точ11,о_точ14_точ15 ), точ12, уг_н_выт, "");*

Поворот удобно использовать и при моделировании драпировок. Нанеся основные линии драпировки, производим повороты:

```
{ первый поворот }
у_п_точ12:=10;
поворот((о1, о2, точ68, т10, пл22л, пл21л, пл23л, пл13л, пл20л, пл19л,
пл52, пл41, пл17л, пл18л, пл42, пл3л, пл2л, пл25, пл9л, пл9, пл10 ),
точ12, -у_п_точ12,"");
{ второй поворот }
у_п_точ68:=15;
поворот((о1, т10, пл23л, пл13л, пл20л, пл19л, пл52, пл41, пл17л,
пл18л, пл42, пл3л, пл2л, пл25, пл9л, пл9, пл10 ),точ68, -у_п_точ68,"");
```

В приведенных примерах поворот производится без дублирования, т.е. первоначально расположенной поворачиваемой детали не остается. На это указывает параметр "".

# 8.4. СЖАТИЕ

При построении лекал для материалов с повышенной группой растяжимости (трикотажные полотна, особенно с использованием лайкры) необходимо сжать отдельные участки лекала для учета их последующего растяжения. В некоторых

случаях, наоборот, требуется растянуть лекало для учета его последующего сжатия из-за усадки ткани или использования какой-либо технологии обработки. Для такой модификации лекала в системе имеется функция сжатия по двум направлениям. Для задания функции сжатия необходимы центр сжатия, угол основного направления сжатия, коэффициенты сжатия вдоль основного направления и вдоль перпендикулярного направления.

*СЖАТЬ((об1,об2,об3,...,обN), точка, угол,*
*К1, К2, (об1_п,об2_п,об3_п,...,обN_п));*

где *об1,об2,об3,...,обN* - список имен графических примитивов, которые нужно сжать,
*точка* - идентификатор точки центра сжатия,
*угол* - угол основного направления сжатия,
*К1* - коэффициент сжатия вдоль основного направления,
*К2* - коэффициент сжатия вдоль перпендикулярного направления,
*об1_п,об2_п,об3_п,...,обN_п* - новые идентификаторы объектов после сжатия.

Например:

*СЖАТЬ((о1,о2,точ68,т10,пл21п), точ12, -у_п_точ12, 1.1, 0.99, '');*

Сжатие целесообразно проводить, по всей видимости, перед записью контура лекала, когда все линии лекала сформированы, хотя, возможно, для некоторых моделей может потребоваться многократное сжатие/растяжение промежуточных контуров. В качестве основного направления рекомендуется использовать направление долевой линии лекала, и, соответственно, первый коэффициент будет определять сжатие/растяжение лекала по долевой линии.

При создании оператора "СЖАТЬ" предполагалось использовать его в первую очередь для создания лекал для трикотажа. Однако оказалось, что этот оператор можно использовать и для конструирования: с его помощью удобно сопрягать по длине различные участки (пройма-рукав, боковые швы), так как небольшое сжатие/растяжение практически не меняет форму лекала, изменяя длины срезов. Более подробно об этом можно прочитать в книге по конструированию.

## 8.5. УСТАНОВКА ТОЧКИ

При построении лекала наиболее часто используемое действие - установка точки относительно известной точки в заданном направлении, на заданном расстоянии. Если направление совпадает с одной из координатных осей, то построение проводится через задание координат новой точки, например:

*новая_точка:=точка(известная_точка.х + смещ_по_Х, известная_точка.у);*

или

*новая_точка:=точка(известная_точка.х , известная_точка.у + смещ_по_У);*

Проиллюстрировать это можно примером определения опорных точек лекала:

л:=точка( а0.х, а0.у+19+0.25*(рост-164)); {линия лопаток}
т:=точка( а0.х, а0.у+Рз_40+Пдтс ); {линия талии}
б:=точка( т.х, т.у+19+0.25*(рост-164)); {линия бедер}
г:=точка( т.х, т.у-(Рз_39-Пспр)); {линия груди}
н:=точка( а.х, а.у+Ди+Пди); {линия низа}

Если требуется отложить расстояние по направлению, задаваемому отрезком или линией, то можно использовать встроенную функцию "ОТЛОЖИТЬ". Для этой функции в качестве параметров Вы задаете: начальную точку, угол, расстояние и идентификатор новой точки. В качестве угла Вы можете использовать угол, задаваемый отрезком, написав идентификатор отрезка с квалификатором .ф1:

*ОТЛОЖИТЬ(т1, угол, смещение, т2));*

где *т1* - идентификатор базовой точки,
*угол* - угол относительно оси Х, по направлению которого откладывается новая точка,
*смещение* - расстояние-число или выражение,
*т2* - идентификатор новой точки,

Например:

*отложить(известная_точка, отрезок.ф1, смещение, новая_точка);*

Если Вы хотите, чтобы направление, в котором будет откладываться точка, составляло определенный угол с базовым направлением, то эту поправку можно просто добавить в величину параметра-угла:

*отложить(известная_точка, отрезок.ф1+поправка, смещение, новая_точка);*

В частности, если Вы хотите откладывать точки в ортогональном направлении по отношении к базовому направлению, задаваемому отрезком или линией, то для этого достаточно добавить или вычесть 90 градусов в параметре-угле в зависимости от стороны, в которую Вы откладываете перпендикуляр:

*отложить(известная_точка, отрезок.ф1+90, смещение, новая_точка);*

*или*

*отложить(известная_точка, отрезок.ф1-90, смещение, новая_точка);*

Часто в промежуточных построениях требуется найти точку на биссектрисе угла. Пример построения проймы спинки:

*отложить(г1, [г1:г2].ф1+([г1:г2].ф1-[г1:г3].ф1), 0.2*|г1:г4|+пр_г01,г01);*

Функцию "ОТЛОЖИТЬ" можно использовать и для деления отрезка на заданные доли, продолжения отрезка в любую сторону. Для деления отрезка точкой на одну и две трети можно записать функцию с параметрами:

*отложить(начало_отрезка, отрезок.ф1, (отрезок.л)/3.0 , новая_точка);*

Для того, чтобы отложить 0.75 длины отрезка от первой точки в противоположном направлении, можно использовать изменение или угла, или знака откладываемого расстояния:

*отложить(начало_отрезка, отрезок.ф1, -отрезок.л*0.75 , новая_точка);*

или

*отложить(начало_отрезка,отрезок.ф1+180,отрезок.л*0.75 , новая_точка);*

Заметим, что все эти преобразования можно было сделать и через расчет координат, однако такая запись была бы громоздкой и непонятной. Исключение составляет деление отрезка пополам, которое записывается как:

*новая_точка:= точка((начало_отрезка.х + конец_отрезка.х)/2, (начало_отрезка.у + конец_отрезка.у)/2 );*

или через функцию "ОТЛОЖИТЬ"*:*

*отложить(п6, [п6:п5].ф1, |п6:п5|/2, г03);*

В новых версиях системы ЛЕКО оператор "ОТЛОЖИТЬ" можно записать в виде

*Имя_точки:=отложить(имя_точки,угол,длина);*

Можно откладывать не по одному направлению, а по нескольким:

*Имя_точки:=отложить(имя_точки,((угол1,длина1), (угол2,длина2)…));*

Рекомендуется для повышения наглядности методики использовать не расчет координат в операторе "ТОЧКА", а использование оператора "ОТЛОЖИТЬ".

Вы можете наиболее удобное для Вас представление.

Иногда требуется отложить расстояние от заданной точки не по прямой, а вдоль дуги или сплайна. Это может потребоваться при установке контрольных точек на пройме и окате рукава или на других криволинейных участках сплайна. Для выполнения такой операции в систему включена встроенная функция "ОТЛОЖИТЬ_В" (вдоль дуги, сплайна или ломаной). Для этой функции в качестве параметров Вы задаете: идентификатор дуги или сплайна, расстояние, идентификатор новой точки.

*ОТЛОЖИТЬ_В(сп, расстояние, т);*

где *сп* - идентификатор дуги или сплайна,
*расстояние* - число или выражение,
*т* - идентификатор новой точки,

Например:

*отложить_в(сплайн_1, расстояние, новая_точка);*

Использование функции при расстановке меток в лекалах:

*отложить_в(спл1,спл1.л*0.5,р1);*
*угол_к(спл1,р1,у_р1);*
*мс1:=метка(р1,тип_метки,у_р1-90,0.5,0);*

Вернемся к команде "ОТЛОЖИТЬ_В". Расстояние откладывается от начала дуги или сплайна. Откладываемое расстояние должно быть меньше длины используемых дуги или сплайна, в противном случае выдается сообщение об ошибке. Если Вы хотите получить перпендикуляр к дуге или сплайну в отложенной точке, Вы можете использовать направление касательной, прибавив или отняв 90 градусов. Например:

*отложить(новая_точка,касательная+90,10,точка_нормали);*

Точка на сплайне может откладываться для того, чтобы затем этот сплайн "разрезать" на две части и развести их - например, при построении нагрудной вытачки, выходящей из горловины. Для этого нужно воспользоваться командой "РАЗДЕЛИТЬ". Для этой функции в качестве параметров Вы задаете: идентификатор дуги или сплайна, расстояние, идентификатор новой точки и идентификаторы новых сплайнов.

*РАЗДЕЛИТЬ(сп, расстояние, т, сп1, сп2);*

где *сп* - идентификатор разделяемой дуги или сплайна,
*расстояние* - число или выражение,
*т* - идентификатор новой точки,
*сп1* - идентификатор первой части разделенной дуги или сплайна,
*сп2* - идентификатор второй части разделенной дуги или сплайна,

Например:

*РАЗДЕЛИТЬ(сплайн_1, 2, новая_точка, сплайн_1_1, сплайн_1_2);*

И теперь наш пример с делением горловины полочки выглядит следующим образом:

*разделить( с_а4а5, 2*с_а4а5.л/3, п7, с_а4п7, с_п7а5);*

При делении сплайна или ломаной не всегда можно задать расстояние до искомой точки вдоль кривой. Требуемая точка может определяться исходя из дополнительных условий. Для их реализации в системе имеется оператор "РАЗДЕЛИТЬ_Н" (направлением). В этом случае в качестве параметров задается точка и направление линии, вдоль которой необходимо разделить сплайн или ломаную:

*РАЗДЕЛИТЬ_Н(сп, т1, угол, т2, сп1, сп2);*

где *сп* - идентификатор разделяемой дуги или сплайна,

*т1* - идентификатор точки, определяющей направление,

*угол* - угол направления линии, вдоль которого необходимо разделить сплайн или ломаную,

*т2* - идентификатор новой точки,

*сп1* - идентификатор первой части разделенной дуги или сплайна,

*сп2* - идентификатор второй части разделенной дуги или сплайна,

Например:

*разделить_н( с_а4_а5, т1, 90, п7, с_а4_п7, с_п7_а5);*

# 8.6. ПЕРЕСЕЧЕНИЕ

Почти все построение лекал основано на пересечении. Это может быть пересечение прямых, дуг, отрезков в любой их комбинации. Часто пересечение используется при моделировании - нанесение фасонных линий и определение их положения на контуре лекал, перенос вытачек и т.д. Для реализации таких действий в языке предусмотрена функция пересечения.

Для записи этой функции используется ключевое слово "ПЕРЕСЕЧЕНИЕ". В качестве параметров записываются пересекаемые переменные и идентификатор точки пересечения.

*ПЕРЕСЕЧЕНИЕ(об1, об2, т);*

или

*ПЕРЕСЕЧЕНИЕ(об1, об2, т,об11,об12,об21,об22);*

где *об1, об2* - отрезок, дуга или сплайн, реально пересекающиеся,

*т* - идентификатор точки пересечения,

об11,об12 – идентификаторы новой дуги или сплайна, полученные в результате деления первой дуги или сплайна,

об21,об22 – идентификаторы новой дуги или сплайна, полученные в результате деления второй дуги или сплайна.

Например:

*пересечение(сплайн_1, сплайн_2, пересечение_сплайнов_1_и_2 );*

определение на боковом шве уровней талии и бедер:

*пересечение( л_т, о_бок, т4);*
*пересечение( л_б, о_бок, б4);*

Если при построении система не найдет пересечения переменных, то дальнейшая обработка программы прекращается и выдается диагностика "Отсутствует пересечение". В этом случае необходимо поправить данные таким образом, чтобы переменные обязательно пересекались при варьировании данных по всему диапазону используемых данных (для всех используемых для этой модели размеро-ростов).

При ручном построении часто используют пересечения прямой и дуги для того, чтобы отложить точку на прямой, однако эту операцию удобней выполнять с использованием функции "ОТЛОЖИТЬ".

# 8.7. ПЕРЕСЕЧЕНИЕ ДУГ

Одна из часто встречающихся операций - поиск пересечения двух дуг. Можно найти пересечение, определив дуги, т.е. задав центры, радиусы, начальные и конечные углы и использовав оператор поиска пересечения переменных. Однако эти дуги используются только в качестве промежуточных построений и в дальнейшем, как правило, не нужны. Для того, чтобы сократить количество промежуточных построений в программе, в язык введен встроенный оператор поиска пересечения двух дуг - "ПЕРЕСЕЧЕНИЕ_Д". В качестве параметров задаются центр и радиус первой и второй дуг, и признак, определяющий, какую из двух точек пересечения окружностей использовать. Принцип такой: если конструктор мысленно "встанет" в первой точке "лицом" ко второй точке, то одна точка пересечения окажется в левой полуплоскости, вторая - в правой. Задавая число больше или меньше нуля, Вы можете указать, какую точку следует использовать. Если после построения окажется, что был указан неверный знак, то можно просто задать поворот на 180°. За признаком Вы записываете идентификатор новой точки.

*ПЕРЕСЕЧЕНИЕ_Д(ц1, р1, ц2, р2, П, т);*

где *ц1* - центр первой окружности,
*р1* - радиус первой окружности,
*ц2* - центр второй окружности,
*р2* - радиус второй окружности,
*П* - параметр, определяющий выбор одной из точек пересечения, *П=1; -1,*
*т* - идентификатор точки пересечения.

Например:

*Пересечение_д( т23, 10, т34, о_14_56.л, п, т45);*

Как было сказано, оператор "ПЕРЕСЕЧЕНИЕ_Д" введен в язык только для сокращения записи программы и количества промежуточных построений. Вы можете использовать его в своих алгоритмах, но можете спокойно обойтись и без него.

Как и при поиске пересечения, если при построении лекала система не найдет пересечения окружностей, то дальнейшая обработка программы прекращается и выдается сообщение "Отсутствует пересечение".

Наиболее ярким примером использования этой функции является определение точки плеча через Впк и Шп (измерения "высота плеча косая" и "ширина плеча"):

*пересечение_д( т0, Впк+Пр, а2, Шп, -1, п);*

# 8.8. ПЕРЕСЕЧЕНИЕ НАПРАВЛЕНИЙ

Функция поиска пересечения переменных достаточно универсальна и позволяет найти точку пересечения или указать, что пересечение отсутствует. Однако для поиска пересечения двух линий, как и пересечения дуг, не всегда удобно проводить полное

построение линий, особенно если эти линии нужны только для поиска указанного пересечения.

Для поиска пересечения двух линий, задаваемых точкой и направлением, в языке предусмотрена функция поиска пересечения направлений. Для записи этой функции используется ключевое слово "ПЕРЕСЕЧЕНИЕ_Н". В качестве параметров записываются точки и направление, задающие первую и вторую линию, идентификатор точки пересечения.

*ПЕРЕСЕЧЕНИЕ_Н(т1, угол1, т2, угол2, т3);*

где *т1* - идентификатор точки, определяющей первое направление,
*угол1* - угол направления первой линии,
*т2* - идентификатор второй точки,
*угол2* - угол направления второй линии,
*т3* - идентификатор точки пересечения.

Например:

*пересечение_н(т1, ф1, т2, ф2, пересечение_линий_1_и_2 );*

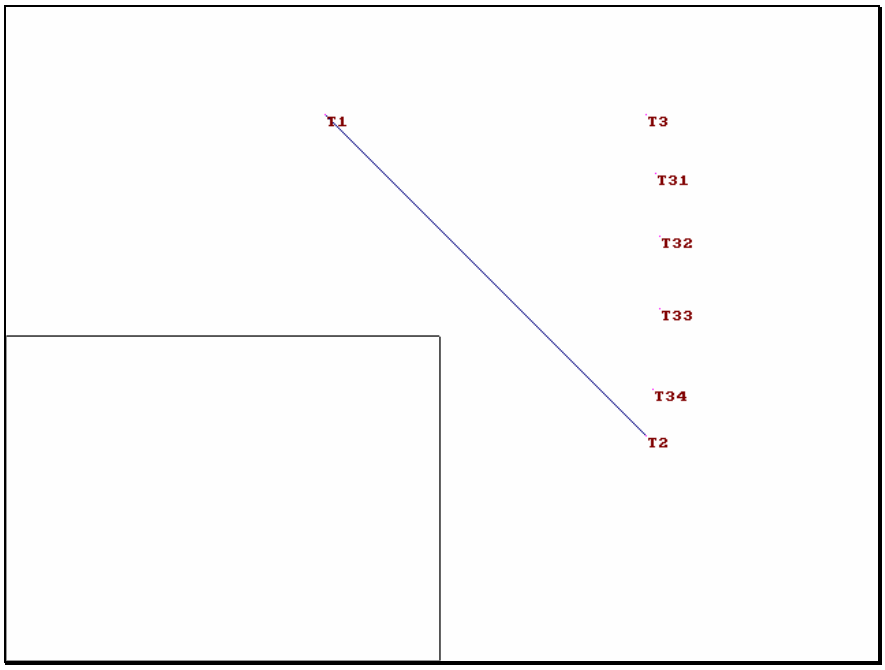
Результатом построения следующей программы:

```
{ пересечение направлений }
т1:=точка( 1, 1);
т2:=точка( 2, 2);

пересечение_н( т1, 0, т2, 90, т3);
пересечение_н( т1, 10, т2, 92, т31);
пересечение_н( т1, 20, т2, 94, т32);
пересечение_н( т1, 30, т2, 96, т33);
пересечение_н( т1, 40, т2, 98, т34);

отрезок(т1, т2);
конец;
```

будет следующее расположение точек:




Заметим, что, как и при поиске пересечения дуг, описываемая функция позволяет сократить количество промежуточных построений. Однако не следует применять эти функции во всех случаях подряд, т.к. в иногда промежуточные построения могут использоваться для дополнительного визуального контроля правильности построения. Основная цель встроенных функций заключается не в том, чтобы убрать все промежуточные построения, а в том, чтобы убрать ненужные промежуточные построения, мешающие дальнейшей работе.

## 8.9. ПЕРЕСЕЧЕНИЕ ДУГИ И НАПРАВЛЕНИЯ

Следующая встроенная функция позволяет найти пересечение окружности и направления. По аналогии с уже описанными процедурами, для задания параметров процедуры необходимо задать: центр окружности, радиус, точку и направление, число-признак, определяющий, какую точку пересечения использовать.

*ПЕРЕСЕЧЕНИЕ_ДН(ц, р, т1, угол, П, т2);*

где *ц* - центр окружности,
*р* - радиус окружности,
*т1* - идентификатор точки, определяющей направление,
*угол* - угол направления линии,
*П* - параметр, определяющий выбор одной из точек пересечения, *П=1; -1,*
*т2* - идентификатор точки пересечения.

Например, результат работы следующей программы:

```
{ пересечение дуги и направления }
т1:=точка( 10, 10);
т2:=точка( 20, 20);
```

*пересечение_дн( m1, 10, m2, 30, 1, m21);*
*пересечение_дн( m1, 10, m2, 30, -1, m22);*

*пересечение_дн( m1, 10, m2, 40, 1, m23);*
*пересечение_дн( m1, 10, m2, 40, -1, m24);*

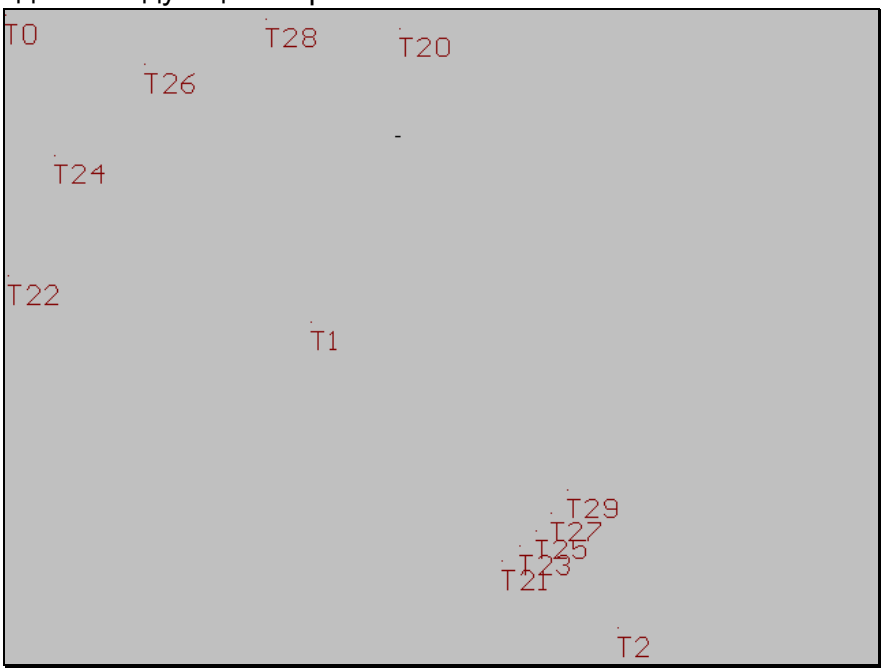
*пересечение_дн( m1, 10, m2, 50, 1, m25);*
*пересечение_дн( m1, 10, m2, 50, -1, m26);*

*пересечение_дн( m1, 10, m2, 60, 1, m27);*
*пересечение_дн( m1, 10, m2, 60, -1, m28);*

*пересечение_дн( m1, 10, m2, 70, 1, m29);*
*пересечение_дн( m1, 10, m2, 70, -1, m20);*

*конец;*

будет выглядеть следующим образом:




## 8.10. СОПРЯЖЕНИЕ ЛИНИЙ ДУГАМИ

Некоторые задачи конструирования требуют сопряжения линий не сплайнами, а дугами заданного радиуса. Для построения таких дуг в системе используется оператор "СОПРЯЖЕНИЕ_Д". В качестве параметров построения дуги задаются: точка на первой линии и направление первой линии, точка на второй линии и направление второй линии, радиус дуги сопряжения, количество точек на дуге, название получившейся дуги.

*СОПРЯЖЕНИЕ_Д(т1, угол1, т2, угол2, р, к, дуга);*

где *т1* - идентификатор точки на первой линии,
*угол1* - направление первой линии,
*т2* - идентификатор точки на второй линии,
*угол2* - направление второй линии,
*р* - радиус дуги сопряжения,
*к* - количество точек на дуге,
*дуга* - идентификатор получившейся дуги.

Например:

*сопряжение_д( т1, н1, т2, н2, 5.5, 20, д1);*

# 8.11. ПОДКЛЮЧЕНИЕ ТАБЛИЦЫ РАЗМЕРНЫХ ПРИЗНАКОВ

При построении лекала Вы можете использовать в расчетных формулах значения размерных признаков, выбираемые из базы данных. Это позволяет вести разработку лекал не в абсолютных величинах, а в пропорциях, пользуясь в основном пропорциональными коэффициентами. Преимущества такого подхода описаны в следующей части описания системы.

Значения размерных признаков из текущей таблицы размерных признаков для выбранного типа фигуры могут использоваться в программе после того, как система встретит ключевое слово "размеры".

# 8.12. ИСПОЛЬЗОВАНИЕ ПАРАМЕТРОВ ПОСТРОЕНИЯ

Разработанным алгоритмом может пользоваться человек далекий от конструирования, или алгоритм может распространяться в зашифрованном виде. В этом случае может потребоваться дать более удобный способ изменения порядка построения, чем редактирование алгоритма.

Оператор «выбрать» позволяет выдать на экран пользователю пункты меню и в зависимости от выбранного пункта сформировать переменную–ответ

```
ВЫБРАТЬ("ц", воп_сп, "Выберите тип спинки",
  ("спинка со сборкой", 0),
  ("спинка без сборки",1));
ВЫБРАТЬ("ц", воп_пол, "Выберите тип полочки",
  ("полочка со сборкой", 0),
  ("полочка без сборки",1));
```

В зависимость от переменной-ответа пользователя в конструкцию вносятся изменения

```
если равно (воп_сп,0) то
  если равно (воп_пол,0) то
```

```
    пнс:=2;          { понижение линии низа спинки }
    пнп:=2;          { понижение линии низа полочки }
    иначе
    пнс:=2;
    пнп:=0;
  конец_если;
иначе
  если равно (воп_пол,0) то
    пнс:=0;
    пнп:=2;
    иначе
    пнс:=0;
    пнп:=0;
  конец_если;
конец_если;
```

При отработке лекал и выборе значений параметров построения не всегда удобно для изменения значений пользоваться текстовым редактором. Для оперативного ввода значений параметров можно использовать встроенную функцию "ВВОД_П" (ввод параметров). При записи функции за ключевым словом "ввод_п" записывают список комментариев и вводимых параметров, например:

```
ввод_п(("угол наклона плеча",угол_п),
     ("угол вытачки",угол_в));
```

При выполнении функции "ВВОД_П" система выдает на экран окно для ввода значений параметров. В окне выводятся до 10 названий параметров, дополнительные комментарии и значения параметров, которые Вы можете исправить. В качестве начальных значений берутся текущие значения, которые имеют параметры в программе.

Значения параметров, вводимые во время построения, записываются на диск (в файл с расширением .prm) и могут затем быть загружены, используя при последующем построении в программу ключевое слово "параметры". Например, если имеется фрагмент построения:

```
параметры;
ввод_п(("высота подплечника",выс_п),
  ("приталенность изделия",к_прит_и),
  ("высота проймы",к_проймы));
```

при первом построении значения параметров в окне ввода будут нулевыми. Если Вы введете значения:1.2, 0.3, 0.1, то при последующем обращении на экран будут выданы введенные значения:

| *Введите параметры* | | |
|---|---|---|
| **высота подплечника** | **выс_п** | **1.2** |
| **приталенность изделия** | **к_прит_и** | **0.3** |
| **высота проймы** | **к_проймы** | **0.1** |

После того, как значения некоторых параметров будут выбраны, эти значения могут быть перенесены непосредственно в текст программы.

# 8.13. ВЫТАЧКА

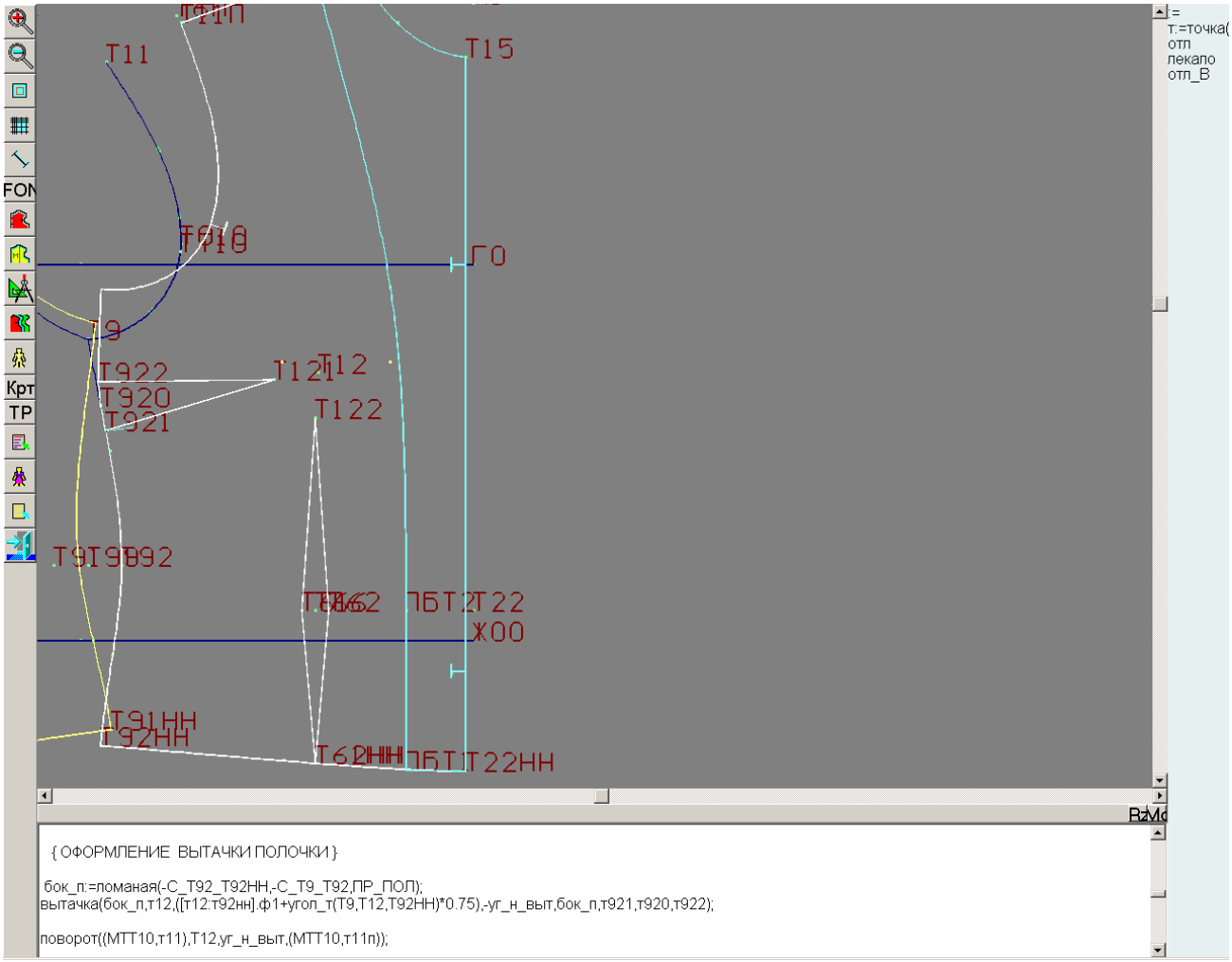
Новый оператор «Вытачка» предназначен для перевода вытачки и оформления концов вытачки в зависимости от способа обработки. Рассмотрим пример перевода нагрудной вытачки из плечевого шва в боковой.

Для начала необходимо сформировать ломаную, контур которой подлежит модификации. Направление ломаной задается в соответствие с направлением закрытия вытачки.

Оператор «Вытачка» записывается следующим образом:

*бок_п:=ломаная(-С_Т92_Т92НН,-С_Т9_Т92,ПР_ПОЛ);*
*Вытачка(ломаная1,т_цетра,уг1,уг2,ломаная1_1,т1,т2,т3);*

где
ломаная1 – идентификатор ломаной, подлежащей модификации;
т_центра – точка центра вытачки;
уг1 – угол разреза ломаной для перевода вытачки;
уг2 – угол раствора вытачки (берется со знаком «+» или «-» в зависимости от способа обработки вытачки: заутюживания вверх или вниз);
ломаная1_1 – идентификатор новой ломаной;
т1,т3 – точки концов вытачки;
т2 – точка середины вытачки.




В нашем примере этот оператор будет выглядеть следующим образом:

*Вытачка(бок_п,т12,([т12:т92нн].ф1+угол_т(T9,T12,T92НН)*0.75),-уг_н_выт,бок_п,т921,т920,т922);*

Следует обратить внимание на угол разреза вытачки. В нашем примере этот угол берется в пропорции от угла между точкой глубины проймы, точкой центра вытачки и нижней точкой бокового шва, что обеспечивает контроль над переводом вытачки при градации.

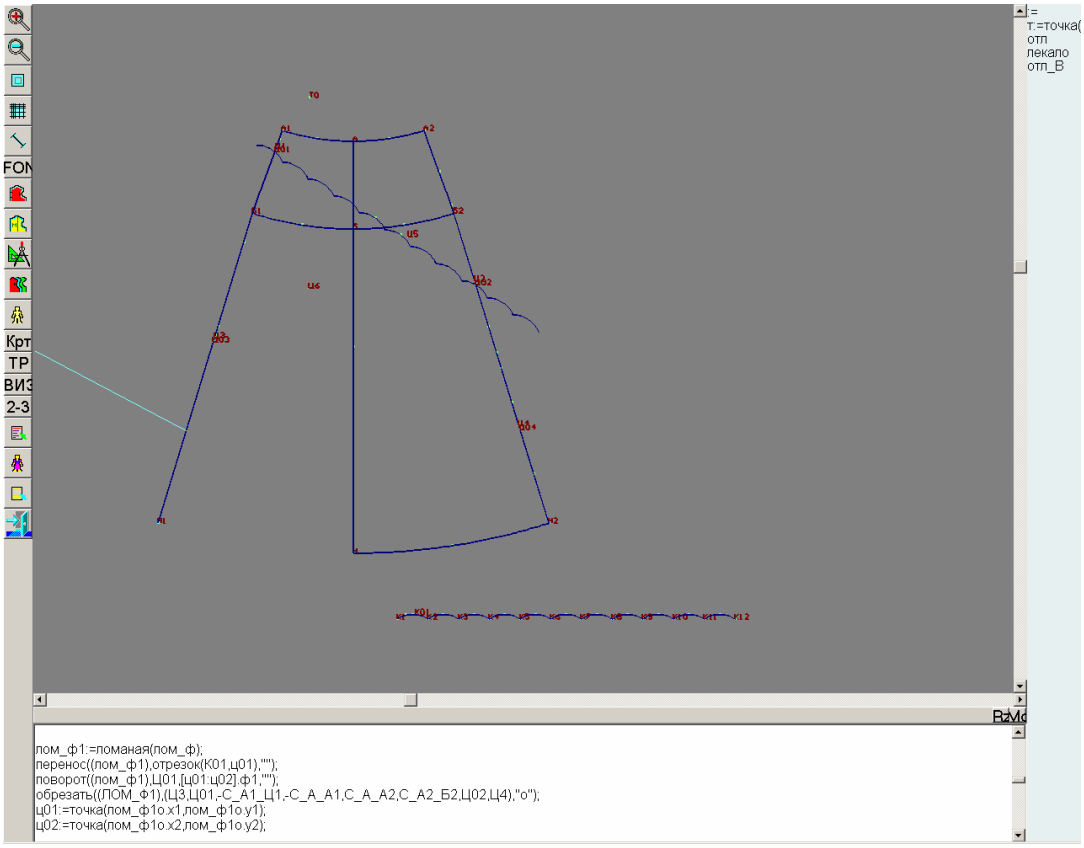
# 8.14.ОБРЕЗАТЬ

Этот оператор предназначен для оформления контуров лекала фигурными линиями, а также для нанесения этих линий на лекало. К примеру, низ кокетки необходимо оформить фестонами или нанести линию притачивания декоративной тесьмы. Для начала необходимо изобразить эту линию отдельным элементом, причем желательно, чтобы длина этой линии была немного больше, чем размер этой линии на лекале. Фигурную линию можно составить из сплайнов, отрезков и т. д., а потом записать их в ломаную. Далее необходимо перенести эту линию на то место, где она будет расположена на лекале, с помощью функций «Перенос» и «Поворот». Затем эта

линия как бы обрезается контуром конструкции, и тем самым находятся точки пересечения фигурной линии и линий конструкции.

*обрезать((ид_линии),(контур лекала),"");*

где ид_линии – идентификатор фигурной линии,
контур лекала – последовательность графических элементов, записанных в виде контура лекала, которым обрезается искомая линия;
"" – идентификатор новой ломаной, полученной в результате применения этой функции.




*лом_ф:=ломаная(С_К1_К2,С_К2_К3,С_К3_К4,С_К4_К5,С_К5_К6,С_К6_К7,С_К7_К8,С_К8_К9,С_К9_К10,С_К10_К11,С_К11_К12);*

*лом_ф1:=ломаная(лом_ф);*
*перенос((лом_ф1),отрезок(К01,ц01),"");*
*поворот((лом_ф1),Ц01,[ц01:ц02].ф1,"");*
*обрезать((ЛОМ_Ф1),(Ц3,Ц01,-С_А1_Ц1,-С_А_А1,С_А_А2,С_А2_Б2,Ц02,Ц4),"о");*
*ц01:=точка(лом_ф1о.х1,лом_ф1о.у1);*
*ц02:=точка(лом_ф1о.х2,лом_ф1о.у2);*

Этот оператор применяется, когда размеры и форма фигурной линии должны оставаться неизменными. Если же размеры фигурной линии могут меняться, а форма должна оставаться постоянной, то для этого используется оператор «Совместить».

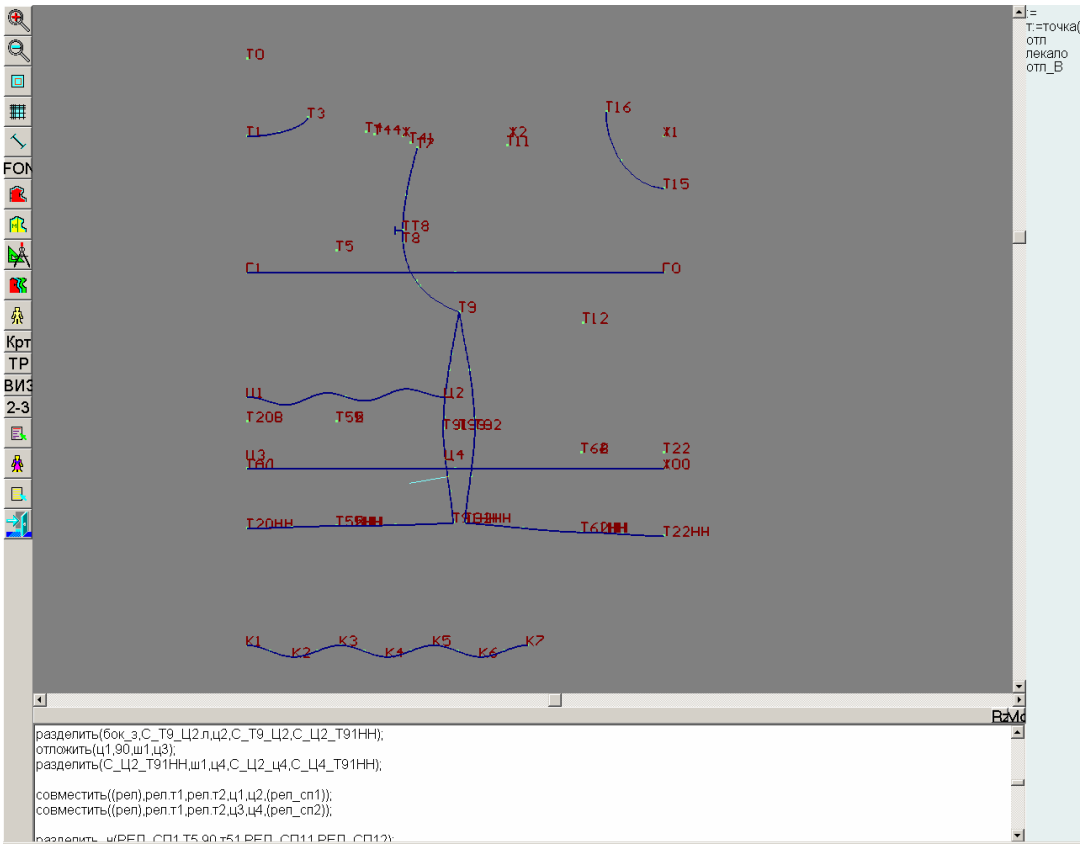
# 8.15. СОВМЕСТИТЬ

Для использования этого оператора сначала следует создать ломаную в виде отдельного элемента (аналогично созданию ломаной в операторе «Обрезать»). Затем на чертеже конструкции необходимо нанести точки, между которыми будет проходить фигурная линия.

*совместить((ид_линии),ид_линии.т1,ид_линии.т2,т11,т12,(ид_линии_нов);*

где ид_линии – идентификатор фигурной линии;
ид_линии.т1 – начальная точка фигурной линии, которая должна быть совмещена с точкой на чертеже;
ид_линии.т2 – конечная точка фигурной линии, которая должна быть совмещена с точкой на чертеже;
т11 – точка на чертеже, с которой совмещается начальная точка фигурной линии;
т12 – точка на чертеже, с которой совмещается конечная точка фигурной линии;
ид_линии_нов – идентификатор новой ломаной.

При выполнении оператора «Совместить» происходит перенос, поворот и масштабирование искомой фигурной линии.




*рел:=ломаная(С_К1_К2,С_К2_К3,С_К3_К4,С_К4_К5,С_К5_К6);*
*отложить(т20нн,-90,ш1*2.1,ц1);*
*разделить_н(бок_з,ц1,0,ц2,с_т9_ц2,с_ц2_т91нн);*

*совместить((рел),рел.т1,рел.т2,ц1,ц2,(рел_сп1));*

# 8.16. РАЗВЕСТИ

Этот оператор предназначен для конического разведения деталей. Поначалу может показаться, что он очень громоздкий и содержит большое количество различных параметров. Однако, при ручном выполнении коническое разведение выполняется в несколько приемов, и чтобы добиться желаемого результата, необходимо многократно проверять и согласовывать размеры моделируемой детали. В данном случае все действия, с помощью которых выполняется коническое разведение, заключены в один оператор.

*РАЗВЕСТИ((контур1),линия_м,уг1,уг2,уг_разведения,коэфф1,*
*(контур2));*

где контур1 – идентификаторы графических элементов, которые подлежат разведению;

линия_м – линия, которая подлежит модификации и относительно которой разрезается деталь;

уг1, уг2 – начальный и конечный углы, вдоль которых разрезается деталь для дальнейшей трансформации;

угол_разведения – суммарный угол разведения детали;

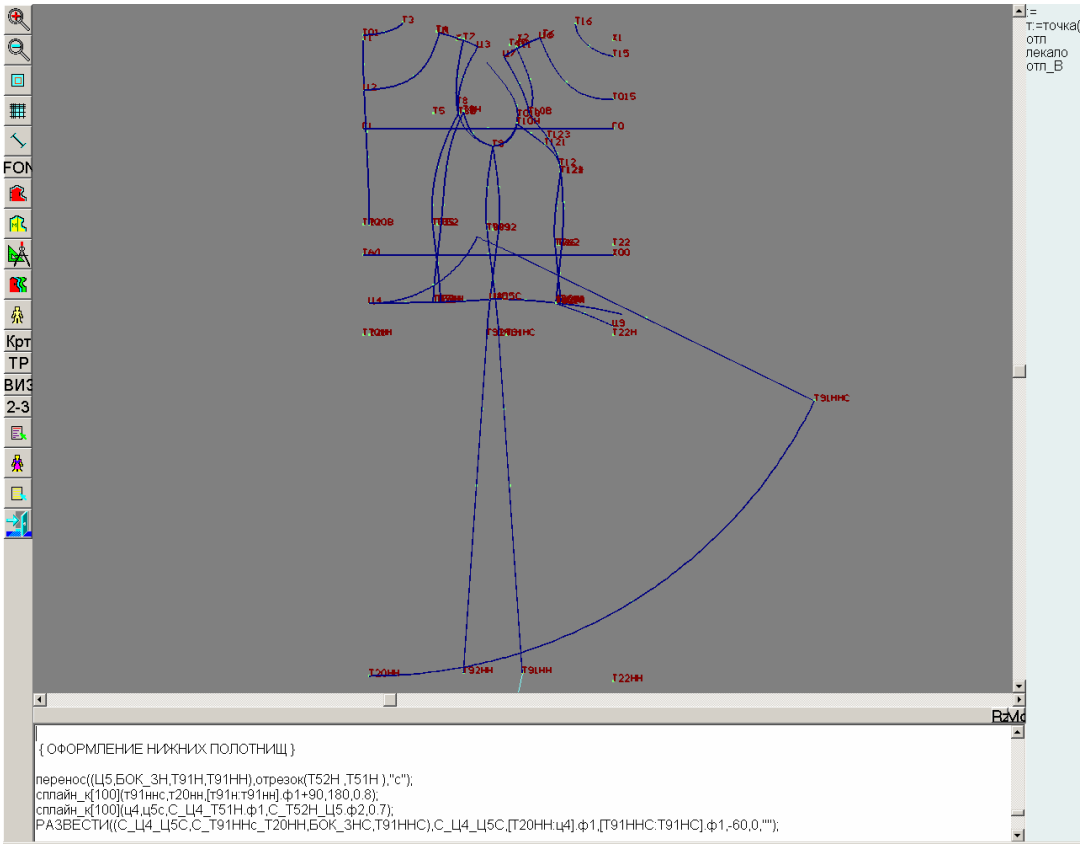
коэфф1 – коэффициент нелинейности; используется, если деталь разводится неравномерно. При равномерном разведении коэфф1=0;

контур2 – идентификатор новых графических элементов, полученных в результате модификации детали.

Для корректной работы этого оператора, необходимо правильно задать его параметры.

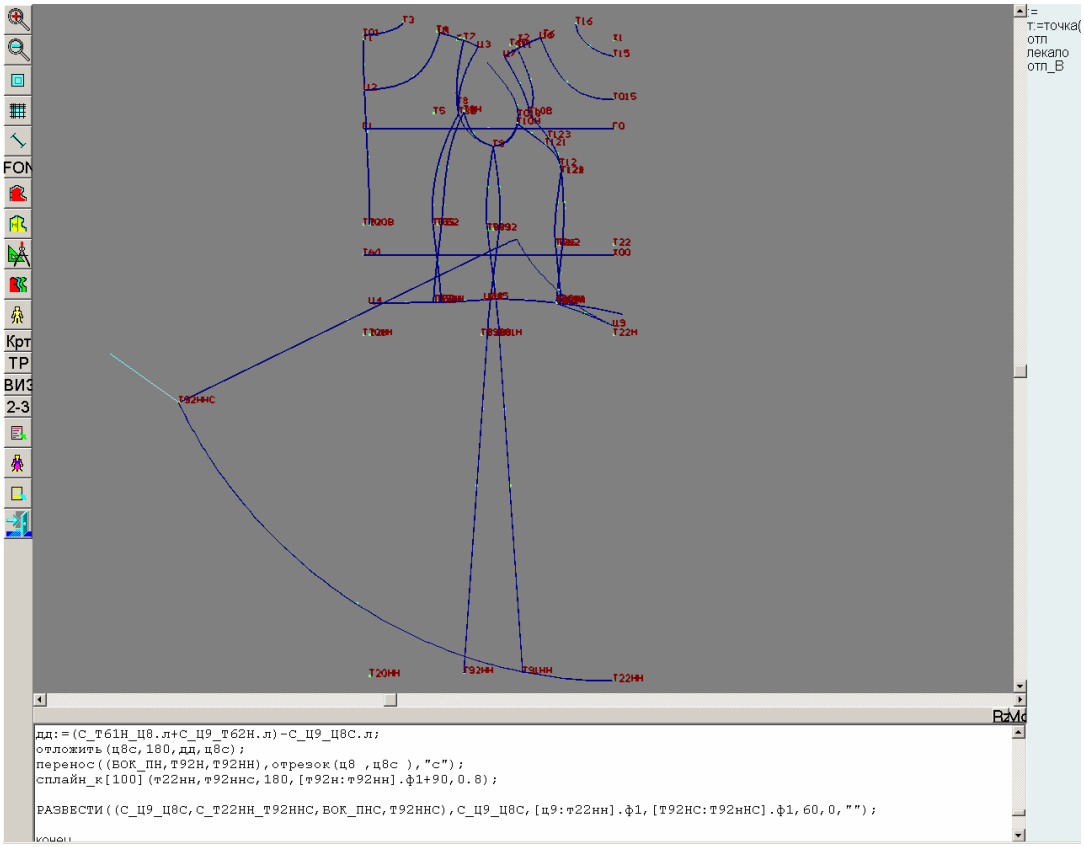
Вначале нужно определить, какой контур разводится - внутренний или внешний, т.к. направление углов разреза детали всегда задается на внешнюю сторону. Суммарный угол разведения детали может быть как отрицательным, так и положительным, в зависимости от того, какая сторона разводится – внутренняя или внешняя. Линия, относительно которой разрезается деталь, тоже задается с учетом направления. Если направление линии не совпадает с направлением перемещения детали при разведении, то используется знак «–», при совпадении направлений – знак «+» (можно опустить).

Пример разведения внутреннего контура:




*РАЗВЕСТИ((С_Ц4_Ц5С,С_Т91ННс_Т20НН,БОК_3НС,Т91ННС),С_Ц4_Ц5С,[Т20НН:ц4].ф1,[Т91ННС:Т91НС].ф1,-60,0,"");*

Пример разведения внешнего контура:




*РАЗВЕСТИ((С_Ц9_Ц8С,С_Т22НН_Т92ННС,БОК_ПНС,Т92ННС),С_Ц9_Ц8С,[ц9:т 22нн].ф1,[Т92НС:Т92нНС].ф1,60,0,"");*

## 8.16. ВОЛАН

Добавлен новый оператор ВОЛАН.

*волан (тц, р, ш1, ш2, д, п1,п2, в1, в2 );*

где тц - центр волана, р - начальный радиус, ш1 - начальная ширина волана, ш2 - конечная ширина волана, д - требуемая длина, два параметра п1, п2, которые пока не используются и лучше их задавать равными 0, в1 - внешний контур волана, в2 - внутренний контур. Для оформления лекала можно записать:

влн1:=лекало(имя=волан_1,контур=в1,внтр=(в2),цвет=15);

Реальная длина волана получается не меньше чем д. При формировании волана по внешнему контуру накручивается полоса по 180 градусов до тех пор, пока реальная длина не превысит заданную.

Оператор ВОЛАН неофициально был доступен и в прежних версиях, однако ширины ш1 и ш2 откладывались приблизительно в два раза больше. Если вы использовали его раньше, то обратите на это внимание.

## 8.17. НАПЕЧАТАТЬ

На рабочей панели есть функциональная кнопка "Просмотр текста". Для формирования просматриваемого текста служит оператор "напечатать". Это используется прежде всего для отладки алгоритма - будь это поиск необходимых переменных по всему тексту алгоритма или подбор различных коэффициентов и параметров.

Рассмотрим, как работает эта функция. Допустим, что конструкция разработана, лекала проверены в макете и теперь предстоит градация, которую нужно проверить. Многие пользователи возразят: градацию можно проверить и на сетке лекал. Да, именно на сетке в первую очередь и проверяется размножение лекал: не выпадает ли какой-либо размер из общего диапазона размероростов, на всех ли размерах соответствует конфигурация линий задумке конструктора и т.д. Но эта проверка только визуальная, а опытный конструктор всегда захочет проверить еще и численные значения коэффициентов и конструктивных отрезков. Все это можно теперь реализовать с помощью оператора "напечатать", который используется в самом алгоритме. В нем Вы указываете все те переменные, численные значения которых хотели бы проверить.

Например:

```
напечатать("Размер"+" "+рз_1+"-"+рз_16+"-"+рз_18);
напечатать("раствор_вытачки"+" "+уг_н_выт);
напечатать("ширина_кармана"+" "+ш_карм);
напечатать("ширина_плеча"+" "+шир_плеча);
напечатать("высота_груди"+" "+выс_гр_в);
напечатать("высота_ростка"+" "+гл_ростка);
напечатать("длина_изделия"+" "+дл_изд);
```

В этом операторе в скобках указаны строковые выражения, т.е. переменные и текст в двойных кавычках, между которыми ставится знак "+" (подробнее о строковых выражениях см. рассылку 12).

Затем, при нажатии кнопки "Просмотр текста" откроется новое окно, в котором будут представлены числовые значения заданных переменных.

```
Размер 170-92-100
Раствор вытачки 13.1479
Ширина кармана 11
Ширина плеча 8.456
Высота груди 27.3
Высота ростка 2.4
Длина изделия 52.3
```

Причем, при построении сетки лекал, в окне просмотра будут отображены числовые значения переменных градируемых размеров. Это своего рода табель мер. Весь текст можно распечатать, а затем проанализировать, как изменяются различные величины при градации и при необходимости внести в алгоритм соответствующие поправки.

Еще один из вариантов использования оператора "напечатать" - отработка циклов. При циклических построениях мы видим только конечный результат, а проследить пошагово, как меняется подбираемый параметр и точность его вычисления до недавнего времени не представлялось возможным. Теперь с помощью оператора "напечатать" можно проследить всю пошаговую работу цикла, определить начальное

значение подбираемого параметра, найти оптимальное приращение, чтобы сократить число итераций. Например, при построении реглана, в цикле подбирается угол наклона задней и передней части рукава. При отладке цикла можно проследить, как меняется угол наклона, определить, какое приращение угла необходимо задавать, чтобы с заданной точностью найти этот угол и тем самым сократить количество итераций.

Как видно, в программе появляются новые функции, которые не только автоматизируют методы конструирования, но и формируют профессиональный подход к самому процессу конструирования.

## 8.18. ЦВЕТ ЛИНИЙ

По умолчанию все линии промежуточных построений имеют темно-синий цвет. В системе есть возможность выбора цвета линий для промежуточных построений. То есть вы можете визуально разделить этап конструирования и моделирования, выделить этапы моделирования и уйти от "темно-синего" цвета, используемого по умолчания. Задание цвета особенно полезно при 3D построениях, так как там гораздо больше линий накладывающихся друг на друга. Операторы

Цвет_л2(число);
Цвет_л3(число);

Задают цвет линий промежуточных построений для 2D и 3D экранов просмотра.

Цвет можно задавать или как раньше в виде числа от 0 до 15, или как задание компонент RGB, например #FF0010.

# 9. АВТОМАТИЧЕСКАЯ ГЕНЕРАЦИЯ ИДЕНТИФИКАТОРОВ

При записи алгоритма всегда возникает желание не выполнять рутинную работу, связанную с набором текста программы. Частично автоматизировать этот процесс можно, используя возможности текстового редактора (см. использование словарей, возможности копирования фрагментов текста). В синтаксис языка также включены правила, позволяющие упростить запись текста программы. Так, при определении отрезков, сплайнов и дуг может быть опущено название переменной-идентификатора. В этом случае, система сама сформирует идентификатор из названий точек, определяющих элемент. Например, по записи оператора

*отрезок(т1,т2);*

система создаст идентификатор отрезка о_т1_т2, который можно использовать при дальнейшей записи программы. Для следующих операторов:

*сплайн_к(о1,о2,д1,д2,к1);*
*дуга(т_пл_2,10,180,270);*

система создаст соответственно следующие идентификаторы:

*с_о1_о2*
*д_т_пл_2*

Как видно из порядка формирования идентификаторов, из одной точки по умолчанию можно сформировать идентификатор только для одной дуги. При повторном определении дуги без идентификатора система выдаст сообщение об ошибке. Такой идентификатор уже есть. Описанная ситуация может возникнуть при определении двух сплайнов, проведенных из одних и тех же точек, в этом случае необходимо будет дать этим сплайнам разные имена:

*сп1:=сплайн_к( точ60, точ61, [точ60:точ58].ф-2, [точ58:точ61].ф+2, 1);*
*сп2:=сплайн_к( точ60, точ61, [точ60:точ59].ф+2, [точ59:точ61].ф-2, 1);*

# 10. ФОРМИРОВАНИЕ КОНТУРА ЛЕКАЛА

Описанные выше действия позволяли рассчитывать и строить точки и линии, пока еще не связанные между собой в отдельные детали проектируемого швейного изделия. Для описания контура лекала из набора кривых, отрезков и точек используется команда:

*записать*

Оператор "ЗАПИСАТЬ" позволяет определить название лекала, его контур и описать дополнительные признаки лекала, как последующей детали швейного изделия. Оператор "ЗАПИСАТЬ" позволяет задать последовательность точек, описывающих контур лекала. Описываемый контур лекала должен быть замкнут; система сама замыкает его по первой и последней точке. Точки контура лекала задаются последовательно по часовой стрелке. Контур лекала можно описать отдельными точками или отрезками, дугами, сплайнами. При использовании отрезков, дуг или сплайнов в контур включаются последовательно все точки переменной от первой до последней, и они также должны описывать контур по часовой стрелке. Если линия определяет часть контура против часовой стрелки, то при включении ее в контур необходимо поставить перед названием линии знак "-" или сформировать новую линию, описывающую контур в нужном направлении.

Оператор "ЗАПИСАТЬ" позволяет описать внутренние точки и линии лекала, долевую линию, дополнительную долевую.

Оператор "ЗАПИСАТЬ" содержит параметры, определяющие имя лекала, код, контур и цвет вывода контура на экран:

*ЗАПИСАТЬ(имя = (имя лекала),*
*код = (код лекала),*
*контур = (последовательность точек контура лекала,*
*записанная по часовой стрелке),*
*внтр = (последовательность точек, описывающая внутренние линии),*
*долевая = (точка, угол долевой линии),*

*надсечки = (последовательность надсечек для печати),*
*цвет = (номер цвета));*

*Имя лекала* - последовательность букв, цифр и символа "_", начинающуюся с буквы.

*Код лекала* - последовательность из 20 символов, предназначенная для внутрипроизводственной кодировки лекал.

*Контур* - последовательность точек, сплайнов, дуг; перечисление ведется по часовой стрелке из любой точки контура.

*Внтр* - внутренние линии лекал. Это может быть разметка петель, карманы, различные рельефные линии, складки, вытачки. Для задания внутренних линий необходимо записать ключевое слово "внтр", за которым в скобках следуют наборы внутренних линий, например:

*внтр=((пк1,-пк3,точ10п),(т2,т4),(т7,с4,с6)),*

*Долевая* - линия, определяющая направление долевой линии лекал. По умолчанию долевая на детали будет направлена вертикально.

*надсечки* - последовательность точек, используемых в качестве меток и предназначенных для контроля соединения лекал. Надсечки ставятся только в том случае, если сформированы припуски на швы. Если припуски на швы по какой-либо причине отсутствуют, то в качестве надсечек можно использовать метки.

*Цвет* - номер цвета вывода лекала на экран. Эти контуры "видны" для печати. Следует помнить, что на печать выводятся линии с номером цвета от 9 до 15.

Приведем пример записи лекал. Перед записью контура определим все звенья:

*пк1:=сплайн_к(точ10,точ11, -90, -90-уг_плеча, 1.10);*
*пк2:=отрезок(точ11,точ16);*
*пк3:=сплайн_к(точ15,точ16, -180, -90, 1.10);*
*пк4:=отрезок(точ15,точ10п);*
*пк5:=отрезок(точ10,точ10п);*

А теперь следует непосредственно сама запись лекала:

*записать(имя=(кокетка_полочки),*
*контур=(пк1,-пк3,точ10п),*
*цвет=11);*

Обратите внимание: второе звено опущено совсем. Это можно сделать, т.к. при последовательной записи сплайнов *пк1* и *-пк3* система автоматически соединит конец сплайна *пк1* (точку ТОЧ11) и начало сплайна *-пк3* (точку ТОЧ16). Таким образом, Вам не нужно насильно соединять ТОЧ11 и ТОЧ16 отрезком. Здесь же Вы могли заметить и использование отрицательного знака перед сплайном. При определении сплайна *пк3* построение велось от точки ТОЧ15 к точке ТОЧ16. Знак же "-" "переопределит" порядок построения - от точки ТОЧ16 до точки ТОЧ11. И последние два звена можно заменить одной точкой ТОЧ10п. Система автоматически соединит отрезком конец сплайна *-пк3* и точку ТОЧ10п и замкнет контур, соединив отрезком точку ТОЧ10п и точку, с которой начиналось описание контура - ТОЧ10.

При формировании маркировки система пытается вписать стандартный текст внутрь контура лекала. Иногда это нежелательно и нужно сформировать текст самостоятельно. Для этого при описании лекала используется оператор ОТМ_МАРК , позволяющий отключить автоматическое формирование маркировки на конкретном лекале. Это может понадобиться на деталях сложной формы, где автоматическая маркировка дает слишком мелкий или неудобно расположенный текст. Оператор может находиться в любом месте внутри оператора ЗАПИСАТЬ.

# 10.1. МЕТКИ

Внутренние линии лекала предназначены для разметки петель, точек расположения карманов, складок и т.д. Прорисовывать маленькие отрезки и прямоугольники по точкам достаточно утомительно. Для упрощения записи некоторых внутренних линий используются метки. Для записи используется ключевое слово "метка". В качестве параметров задаются центр метки - определение или идентификатор точки, тип метки - число от 1 до 8, угол наклона, два размера (при нулевом угле - горизонтальный и вертикальный размер метки). Например:

*м1:=метка(т1,1,0,2,0); { метка - отрезок }*
*м2:=метка(точка(120,40),2,0,2,0); { метка - прямоугольник }*

Для записи метки в качестве внутренней линии необходимо указать идентификатор метки в качестве отдельного контура, например:

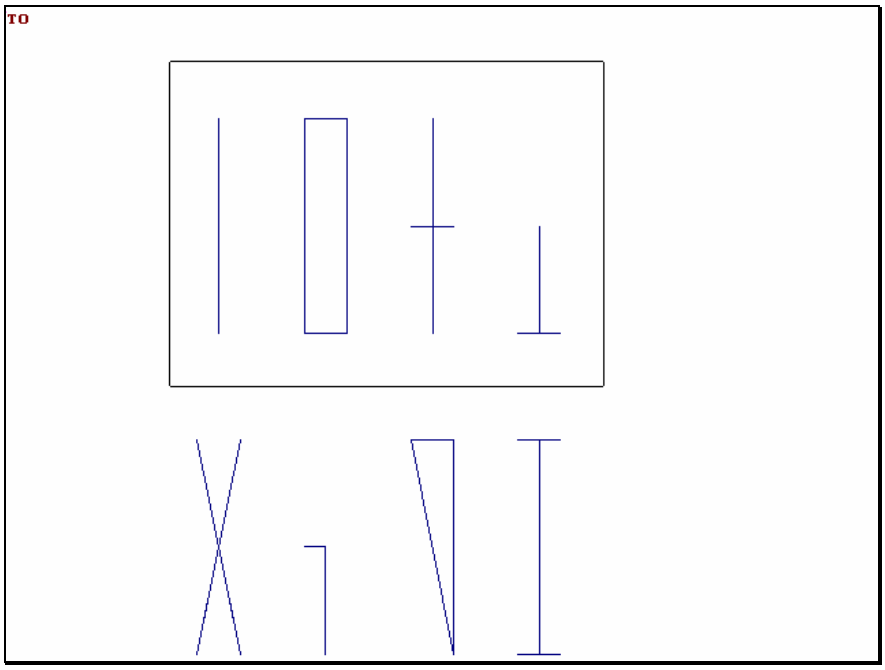
*внтр=(...,(м1),(м2),....),*

Некоторые метки образуют замкнутый контур или контур с самопересечениями, поэтому их не следует использовать для записи основного контура лекала.

Вы можете использовать 8 типов меток:

| номер | цвет |
|---|---|
| 1 | отрезок |
| 2 | прямоугольник |
| 3 | крестик |
| 4 | Т-образная |
| 5 | крестовина |
| 6 | уголок |
| 7 | треугольник |
| 8 | Н-образная |

Контур метки задается относительно прямоугольника с центром в центре метки, повернутом на заданный угол наклона метки и имеющим соответственно длины сторон, равные двум размерам метки.




Как видите, все метки можно описать другими операторами языка, но использование меток позволяет это сделать проще.

Можно использовать оператор <метки>. При помощи этого оператора метки можно описывать непосредственно в операторе <записать> без создания отдельной дополнительной переменной. Например:

```
ЗАПИСАТЬ(имя=(полочка),
контур=(т1,т2,т3,т4),
метки = (
(3,0,2,1,т5),
(14,90,30,2,точка(т2.x-5.5,т2.y+21.5)),
(11,0,2,2,(т1,45,30)),
(11,0,2,2,(т1,((45,30),(0,10))))
),
```

Как видно из примера, метка может устанавливаться в переменную-точку, в определение точки через координаты или через форму записи оператора <отложить>. т.е.

```
(номер_метки, угол, длина, ширина, ид_точки)
(номер_метки, угол, длина, ширина, точка(коорд_x, коорд_y))
(номер_метки, угол, длина, ширина, (ид_точки, угол, длина))
(номер_метки, угол, длина, ширина, (ид_точки,((уг1,дл1),(уг2,дл2)))
```

Расширено количество меток. Метки строятся на базе прямоугольника, с заданной длиной и шириной и повернутого на угол метки. Часть меток строиться на основе полного прямоугольника, с точкой в центре, другая часть на основе правой половины этого прямоугольника.

Посмотреть полный список меток и определить их назначение (или придумать свой способ использования) можно, запустив в новой версии следующий пример:

```
{ пример записи меток }
т1:=точка(10,10);
т2:=точка(70,10);
т3:=точка(70,70);
т4:=точка(10,70);
т5:=точка(24,14);
раст_пет:=2;
ЗАПИСАТЬ(имя=(полочка),
контур=(т1,т2,т3,т4),
метки = (
(3,0,2,1,т5),
(14,90,30,2,точка(т2.х-5.5,т2.у+21.5)),
(13,0,30,3,точка(т4.х+30.0,т4.у-3.5)),
(1,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*1)),
(2,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*2)),
(3,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*3)),
(4,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*4)),
(5,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*5)),
(6,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*6)),
(7,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*7)),
(8,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*8)),
(9,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*9)),
(10,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*10)),
(11,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*11)),
(12,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*12)),
(13,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*13)),
(14,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*14)),
(15,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*15)),
(16,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*16)),
(17,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*17)),
(18,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*18)),
(19,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*19)),
(20,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*20)),
(21,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*21)),
(22,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*22)),
(23,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*23)),
(24,0,2,1,точка(т1.х+5.5,т1.у+1.5+раст_пет*24)),
(11,0,2,2,(т1,45,30)),
(11,0,2,2,(т1,((45,30),(0,10))))
),
цвет=15);
конец;
```

## 10.2. ИСПОЛЬЗОВАНИЕ ЦВЕТА

При просмотре построения в режиме "ВСЕ ЛЕКАЛА" на экран выводятся все точки, линии, отрезки сплайны и дуги, которые были определены в программе. Чтобы

при просмотре на экране разобраться, какие линии основные, а какие вспомогательные, Вы можете использовать выделение контуров лекал различными цветами. По умолчанию, все элементы выводятся синего цвета. Используя оператор

*записать( ....., цвет=11);*

Вы можете изменить синий цвет (он имеет номер 1) на любой другой, с номером от 2 до 15, в соответствии с таблицей цветов:

| Номер | Цвет |
|---|---|
| 1 | синий |
| 2 | зеленый |
| 3 | голубой |
| 4 | красный |
| 5 | фиолетовый |
| 6 | коричневый |
| 7 | ярко-серый |
| 8 | темно-серый |
| 9 | ярко-синий |
| 10 | ярко-зеленый |
| 11 | ярко-голубой |
| 12 | ярко-красный |
| 13 | ярко-фиолетовый |
| 14 | ярко-желтый |
| 15 | белый |

## 10.3. ФОРМИРОВАНИЕ ПРИБАВОК НА ШВЫ

Для задания величины прибавки на швы при описании контура лекала используется ключевое слово "прибавка", например:

*записать(имя=(кокетка_полочки),*
*контур=(пк1,-пк3,точ10п),*
*прибавка=1.5 );*

При формировании прибавки на швы система обводит лекало новым контуром, передвигая каждую точку так, чтобы новый контур отстоял от исходного на расстояние не менее заданного. Все точки передвигаются одинаково, за исключением очень острых углов (менее 5 градусов), где система "обрезает" угол.

Такое задание общего припуска упрощает получение рабочих лекал, и во многих случаях не требует больше никакой доработки. Но иногда для получения окончательного вида лекала требуется задавать припуски на швы дифференцированно по участкам и дополнительно указывать порядок оформления "уголков", в зависимости от технологии обработки шва.

Для задания порядка оформления "уголков" лекал при формировании прибавки на швы используется ключевое слово "прибавка_т", за которым следует список точек и признаков, определяющих внешний вид и свойства "уголка":

*прибавка_т=(( точ1, 6), ( точ40, 1)....),*

Признак внешнего вида "уголка" может принимать различное значение. Он должен задаваться непосредственно числом, а не выражением. Если значение признака выходит за диапазон, то он принимается за ноль. Значения признака имеют следующий смысл:

| КОД | Оформление |
|---|---|
| 1 | усеченный уголок по точкам пересечения швов и линии края |
| 2 | усеченный уголок, линия усечения симметрична второму краю относительно первого шва |
| 3 | усеченный уголок, линия усечения симметрична первому краю относительно второго шва |
| 4 | усеченный уголок, прямой угол для первого шва |
| 5 | усеченный уголок, прямой угол для второго шва |
| 6 | усеченный уголок, линия усечения определяется по расстоянию относительно угловой точки. |

Другие уголки можно выбрать из списка, нажав Alt+I.

На отдельных участках лекала иногда требуется изменить величину прибавки на шов, например, прибавку на подгиб. Для этого используется ключевое слово "прибавка_у", за которым следует список, состоящий из начальной точки, величина прибавки слева от точки, величина прибавки справа от конечной точки, конечная точка. Например:

*прибавка_у=(( точ1, 2.5, 2, точ40)....),*

В качестве величин прибавок на швы могут быть выбраны числа или арифметические выражения, параметры. Величины прибавки справа от начальной точки и прибавки слева от конечной точки могут быть не равны между собой. Дифференцированное задание припуска на швы используется, например, вдоль линии сидения в классических мужских брюках. В этом случае результирующая величина прибавки будет равномерно изменяться от первого значения в начальной точке до второго в конечной. Например:

*записать(имя=(спинка),*
*контур=(точ1,сп2,сп4,сп7,сп8,-сп9,-сп13,р3,точ40),*
*прибавка = 0.5;*
*прибавка_т=((точ1,6),(точ40,1),(р1,5)),*
*прибавка_у=(( точ40, 2.5, 1.5, точ1)),*
*цвет=15);*

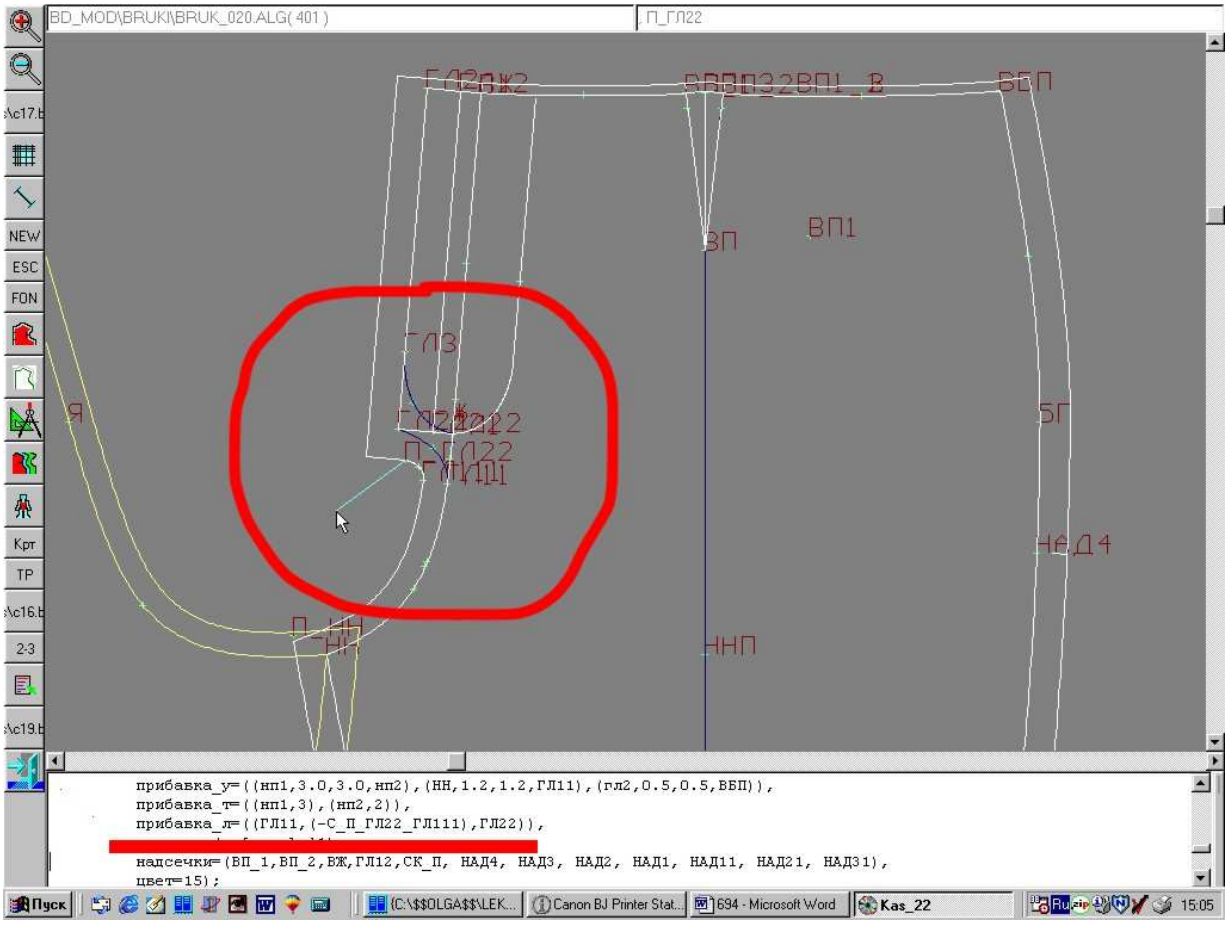
Формируемые автоматически прибавки на швы и порядок оформления уголков удовлетворяет многим конкретным приложениям. Если используется оригинальная технологическая обработка шва и автоматический вариант обработки контура не подходит (использование специфической фурнитуры, ориентация на возможности

оборудования, автоматов), всегда можно описать прибавку на швы для определенного участка на уровне конструкции и при описании лекала установить ручное описание прибавки на данном участке:

*прибавка_л=((m1,(m11,m12,m13,m14,m15),m2)),* где

где т1, т2 – точки, между которыми прибавка на швы описывается вручную;
т11,т12,т13,т14,т15 – контур для ручного построения прибавки на шов на заданном участке.

Пример:




Такой способ может понадобиться в тех случаях, когда автоматическое формирование прибавок не дает нужного результата (скругление краев, поворот линии на небольшой угол, оформление острых углов и т.п.).

## 10.4. СИММЕТРИЯ

При описании контура симметричного лекала можно описать только половину лекала, задав до описания контура признак симметричности, используя ключевое слово «симметрия». Например:

*записать(имя=(спинка),*
*симметрия,*
*контур=(точ1,сп2,сп4,сп7,сп8,-сп9,-сп13,р3,точ40),*

Для формирования симметричной части в качестве оси симметрии берется первая и последняя точка, заданные в контуре лекала (в примере – точ1 – точ40). Если в начале или конце описания контура стоит сплайн или ломанная, то берется первая или последняя точка сплайна или ломанной.

Иногда при построении конструкции получается так, что на чертеже представлена правая деталь, а лекало должно быть левым. Например, при проектировании блузки на чертеже получаются правые детали, а деталь полочки надо записать как левую (разница в деталях может быть в оформлении застежки, разметке карманов, асимметричных модельных особенностях и т.д.). Для того чтобы не выполнять отдельно симметрию графических элементов и затем записывать их в виде нового контура, можно использовать оператор «симметрия_л» непосредственно в записи лекала. Контур записывается в том виде, в каком он есть, а перед описанием контура необходимо добавить оператор «симметрия_л»:

*симметрия_л=отрезок( т1,т2 ),*

где отрезок ( т1, т2) – отрезок, относительно которого отобразится лекало.

# 10.5. ОБЪЕДИНЕНИЕ ЛЕКАЛ

Оператор "ОБЪЕДИНИТЬ_2" предназначен для объединения двух лекал в единое целое лекало. В каком случае это нужно, если можно сразу описать целиковое лекало? Иногда, нужно раскрыть сладки только на части лекала, оставляя остальную часть, так же попадающую на линию складки неизмененной. Для этого можно использовать представленный оператор: описываем два лекала, раскрываем на одном из них складки и соединяем два лекала в одно. Так же можно поступить если допустим нам нужна симметричная верхняя часть лекала и несимметричная нижняя часть: описываем верхнюю часть с признаком "симметричный" и соединяем верхнюю и нижнюю часть. По мере того, как будут появляться новые операторы для работы с целыми лекалами оператор объединения лекал будет более востребован.

Формат записи

Лек_о:= ОБЪЕДИНИТЬ_2(лек_1,(т1,т2), лек_2,(т3,т4));

Где

Лек_о - итоговое лекало, лек_1 лек_2 - соединяемые лекала, т1 т2 - точки отрезка соединения на первом лекале заданные по часовой стрелке, т3 т4 - на втором лекале. Для соединения требуется равенство длин отрезков на первом и втором лекале. Возможный нахлест лекал на линии соединения игнорируется. После объединения соединяемые лекала удаляются.

Если итоговое лекало Лек_о не задано, то есть оператор выглядит

ОБЪЕДИНИТЬ_2(лек_1,(т1,т2), лек_2,(т3,т4));

То к первому лекалу присоединяется второе.

Сделано расширение оператора "ОБЪЕДИНИТЬ_2". Этот оператор преобразует координаты присоединяемого лекала, рассчитывая сдвиг и поворот. Так же преобразуются и внутренние линии присоединяемого лекала, точки надсечек, точки припусков. Если после преобразования с контуром лекала нужно дальше работать, то желательно иметь возможность вместе с контуром присоединяемого лекала преобразовывать дополнительные вспомогательные точки и линии. Для этого в операторе ОБЪЕДИНИТЬ_2, как и в других операторах преобразования, можно дописать список преобразуемых переменных и строку, добавляемую к названиям, или список новых названий переменных:

цч:=ОБЪЕДИНИТЬ_2(зп,(Б2,Н2),бч2,(Т40,Т10), ((Т10,Т30,Т20,Т40,С_В11_В3) , (Т10_ц,Т30_ц,Т20_ц,Т40_ц,Tww_ц)));

или

цч:=ОБЪЕДИНИТЬ_2(зп,(Б2,Н2),бч2,(Т40,Т10), ((Т10,Т30,Т20,Т40,С_В11_В3) , "_ц"));

## 10.5. РАЗВЕДЕНИЕ

Оператор РАЗВЕСТИ_Л работает с целыми лекалами приблизительно так, как обычно рисуют в книжках. Предыдущий оператор РАЗВЕСТИ, работающий с линиями, был тоже хорош, но для его правильного использования требовалось хорошее воображение: нужно было представить лекало и направления реза разводимых участков. С новым оператором ничего представлять не нужно и линии реза система определит сама. Поэтому его описание и интерпретация чисел несколько иная. Задается участок лекала (пара точек) , относительно которого разводим и участок лекала (другая пара точек), который разводим. Угол разведения, как и раньше угол разведения, а знак величины угла определяет в какую сторону будем разводить: слева - направо или справа налево, то есть какие точки и участки лекала будут неподвижны, а какие будут разворачиваться.

РАЗВЕСТИ_Л (ид_лек, (т1,т2), (т3,т4), уг, коэф);

Как и в операторе ОБЪЕДИНИТЬ_2 можно задать список переменных, которые будут преобразованы. Если есть необходимость, их можно использовать для формирования неразведенных внутренних линий и добавить их в описание через оператор ЛЕКАЛО_П.

РАЗВЕСТИ_Л (ид_лек, (т1,т2), (т3,т4), уг, коэф, ((Т10,Т30,Т20,Т40,С_В11_В3) , "_ц"));

В операторе РАЗВЕСТИ_Л есть ограничение: не рекомендуется большая разница (более 90 градусов) между начальным и конечным направлениями реза разведения. Если такое нужно, то стоит разбить разведение на два участка и два оператора, с использованием дополнительной промежуточной точки.

# 11. ДОПОЛНИТЕЛЬНЫЕ ВОЗМОЖНОСТИ ПРЕОБРАЗОВАНИЯ ОБЪЕКТОВ

При симметрии, параллельном переносе или повороте может возникнуть необходимость преобразования самих переменных без сохранения образов переменных. Такая ситуация может возникнуть, например, при множественном преобразовании, таком, как моделирование драпировки защипами или складками. Создание большого количества точек, отрезков увеличит количество определяемых и используемых переменных и усложнит работу. Для упрощения работы Вы можете при преобразовании "перемещать" переменные, указав в качестве добавляемой строки "пустую" подстроку "". Например:

*поворот((т234,о23),т34,30,"");*

точка т234 будет повернута на угол 30° относительно точки т34. Описанной возможностью нужно пользоваться очень аккуратно и в разумных пределах, т.к. ее использование может привести к побочным ошибкам. Например, если Вы определите отрезок о_точ1_точ2 :

*отрезок(точ1,точ2);*

затем повернете первую точку с перемещением:

*поворот((точ1),точ2,40,"");*

то точка точ1 уже не будет началом отрезка о_точ1_точ2, и на начало отрезка Вы можете сослаться только через квалификаторы о_точ1_точ2.х1 и о_точ1_точ2.у1.

# 12. ПРОСМОТР ЛЕКАЛ

Вы можете изменять положение активных и фоновых лекал при просмотре. Для этого необходимо выделить нужное лекало, указав на его центр (желтый квадратик), и нажать правую кнопку мыши. Используя следующие функциональные клавиши, Вы можете:

поворачивать лекало – Ctrl+стрелки (поворот на 10 градусов) или просто стрелками,

F7 – симметрия,

перемещать лекала – просто тянуть его мышью,

увеличивать фрагмент изображения – с помощью функциональной кнопки панели инструментов,

форматировать изображение конструкции под размер экрана – F11.

При этом после перепостроения (клавиша F9) активные лекала вернутся на исходную позицию, фоновые же останутся там, куда были перемещены.

Если при обработке алгоритма построения лекала была обнаружена ошибка, то в окне просмотра Вы можете увидеть часть построения и лекала, построенные системой до появления ошибки. Возможность просмотра уже построенной части лекала поможет Вам при поиске ошибок, особенно ошибок типа "отсутствие пересечения".

При этом следует помнить, что рисунок чертежа является только рисунком, и по нему не следует судить о соотношениях, пропорциях, пересечениях и совпадениях, так как практически любой ЭЛТ-монитор имеет линейные и нелинейные искажения. Основой при построении лекала должна быть последовательность операций, определяющая, в свою очередь, жесткую последовательность действий. Именно продуманный алгоритм построения лекала позволит Вам эффективно использовать все преимущества компьютера, иначе система ничем не будет отличаться от карандаша и бумаги.

# 13. ТОЧНОСТЬ РАСЧЕТОВ

Заметим, что часть построений в системе ведется не на основе точных формульных расчетов, а численными методами. Это следует учитывать при построении, особенно тем, кто хочет использовать "чисто" геометрические построения на основе точных формул. Т.е., например, при попытке найти пересечение окружности и ее касательной система может выдать сообщение "Нет пересечения". Для обхода таких "ошибок" при построении следует использовать другие способы поиска пересечений. В случае с касательной ее (касательную) лучше заменить нормалью к этой касательной, проведенной через центр окружности. Точка пересечения в этом случае не меняется, и система определит ее с приемлемой для построения лекала точностью. Можно также найти точку пересечения, отложив требуемое расстояние от исходной точки, рассчитав его по формуле.

Следующий пример – касательные к сплайнам. Если вы задали сплайн, то углы касательных берутся из оператора задания сплайна. Если построить ломаную на основе сплайна, то углы будут рассчитываться исходя из крайних точек ломанной, и из-за дискретности задания ломаной будут отличаться от значений углов сплайна. Поэтому если необходимо использовать точное значение угла, то следует использовать или начальное значение при задании сплайна, или угол касательной сплайна.

Желающим работать с "чистой" геометрией рекомендуем использовать построения через расчет координат по точным аналитическим зависимостям. В этом случае ошибка вычисления будет определяться только дискретностью представления чисел в компьютере.

Численное решение некоторых задач приводит не только к погрешностям, связанным с численными методами, но и в некоторых случаях - к большим затратам времени.

Задача точности вычислений в системе ЛЕКО тесно связана с "грубостью" (робастностью) алгоритма построения лекала по отношению к исходным данным (измеряемым размерным признакам). Размерные признаки, снятые различными людьми, могут отличаться, даже если измерялся один и тот же человек. Алгоритм конструирования лекал должен быть построен таким образом, чтобы небольшие различия в снимаемых размерных признаках не приводили к значительному изменению лекал. Алгоритмы, не отвечающие требованию "грубости" по отношению к исходным данным (робастности), заранее обречены на провал при массовом использовании, хотя

и могут успешно применяться отдельной группой модельеров-конструкторов, например, при построении на размерные признаки по ОСТ.

# 14. НОВЫЕ ЭЛЕМЕНТЫ ЯЗЫКА

## 14.1. НОВЫЙ ТИП ДАННЫХ

В язык построения лекал введен новый тип переменных - строковые переменные (текст). Для построения лекал строковые переменные не нужны. Они нужны для оформления лекал (дополнительные надписи на лекалах) и формирования отчетов (дополнительной обработки). Для записи в текстовую переменную используется оператор:

записать_т(ид, строковое_выражение);

Строковое_выражение – это переменные и текст в двойных кавычках, между которыми ставится знак "+", например:

*«длина шва»+дл_шва*
*«ширина плеча »+шр_пл+«см.»*

Далее строковые переменные можно использовать при выводе текста на лекалах, в операторе "ПРЕДУПРЕДИТЬ" и при записи в дополнительные параметры лекал.

Оператор "ПРЕДУПРЕДИТЬ" позволяет упростить отладку методик и предупредить конструктора или конечного пользователя об ошибке или особых ситуациях. Как правило, оператор "ПРЕДУПРЕДИТЬ" используется совместно с условными операторами. Например:

*если больше(шр_плеча, 14) то*
*предупредить («слишком широкое плечо »+ шр_плеча);*
*иначе*
*конец_если;*

## 14.2. НОВЫЕ ОПЕРАТОРЫ

Для решения новых задач и упрощения решения существующих были введены новые операторы. Среди них один из самых полезных – "Л_ФНК", линейная функция. Оператор предназначен для создания переменных коэффициентов. Пример:

*пр_1:= рз_16*л_фнк(рз_16,((80,0.01), (88,0.01), (100,0.05), (120,0.1), (140,0.1)));*

где пр_1 – идентификатор переменной,
запись «л_фнк(рз_16,((80,0.01), (88,0.01), (100,0.05), (120,0.1), (140,0.1)))» означает:
рз_16 – размерный признак, в зависимости от которого выбираются коэффициенты,

в скобках указаны значения размерного признака (80,88,100,120 и т.д.) и соответствующие им коэффициенты (0.01,0.05,0.1 и т.д.).

В приведенном примере определена прибавка *пр_1*, зависящая от обхвата груди 3, а также "переменного" коэффициента, который на размере 80 равен 0.01, на 88 – 0.01, на 100 - 0.05, на 120 - 0.1, на 140 - 0.1, а на промежуточных размерах рассчитывается по линейной зависимости. Например, на размере 110 коэффициент будет равен 0.075, а прибавка - 8.25.

Внимание! Использование оператора "Л_ФНК" таит в себе и некоторые сложности. Если результат выражения

*пр_1:= рз_16* 0.1;*

можно предсказать на любом размере, то результат простого выражения

*пр_1:= рз_16*л_фнк(рз_16,( (80,0.01), (100,0.05)));*

предсказать достаточно сложно, а именно:

| Размер | Коэффициент | Прибавка |
|---|---|---|
| 60 | -0.03 | -1.8 |
| 70 | -0.01 | -0.7 |
| 80 | 0.01 | 0.8 |
| 90 | 0.03 | 2.7 |
| 100 | 0.05 | 5 |
| 110 | 0.07 | 7.7 |
| 120 | 0.09 | 10.8 |
| 130 | 0.11 | 14.3 |

Т.е. при выходе за диапазон 80-100 коэффициент становится или слишком маленьким (отрицательным) или слишком большим, и, соответственно, слишком сильно меняется прибавка. Чтобы исправить положение, необходимо добавить в описание функции значение коэффициента для очень маленького и очень большого размеров:

*пр_1:= рз_16*л_фнк(рз_16,( (10,0.01), (80,0.01), (100,0.05), (160,0.05)));*

Тогда таблица будет иметь следующий вид

| Размер | Коэффициент | Прибавка |
|---|---|---|
| 60 | 0.01 | 0.6 |
| 70 | 0.01 | 0.7 |
| 80 | 0.01 | 0.8 |
| 90 | 0.03 | 2.7 |
| 100 | 0.05 | 5 |
| 110 | 0.05 | 5.5 |
| 120 | 0.05 | 6.0 |

| | | |
|---|---|---|
| 130 | 0.05 | 6.5 |

Т.е. значения коэффициента необходимо задавать для максимально большего диапазона, тогда значение функции можно будет сразу оценить на любом размере, и Вы будете застрахованы от подобных ошибок.

Следующий оператор, существующий во многих предшествующих версиях, но подробно не описанный, – условный оператор "ЕСЛИ". Для программистов ничего необычного в этом операторе нет, однако для конструктора этот оператор может таить некоторые сложности при отладке методики. Поэтому условный оператор отсутствовал в описании языка в прошлых версиях. Форма записи оператора следующая:

```
если условие то
  действие_1;
иначе
  действие_2;
конец_если;
```

или

```
если условие то
  действие_1;
конец_если;
```

В условии можно сравнивать числа или арифметические выражения, сравнивать строки, совмещать несколько проверок в одном условии. Для арифметических выражений предусмотрены следующие сравнения:

```
Больше(ар_выр_1, ар_выр_2)
Меньше(ар_выр_1, ар_выр_2)
равно(ар_выр_1, ар_выр_2)
Больше_р(ар_выр_1, ар_выр_2)
Меньше_р(ар_выр_1, ар_выр_2)
```

При сравнении результатов вычислений арифметических выражений при помощи операторов «равно», «больше_р», «меньше_р», перед сравнением результат выражений сначала округляется до целого числа, а затем происходит сравнение этих результатов.

Для строковых выражений предусмотрены следующие сравнения:

```
Т_совпадает(стр_выр_1, стр _выр_2)
не_совпадает(стр_выр_1, стр _выр_2)
Т_содержит(стр_выр_1, стр _выр_2)
не_содержит(стр_выр_1, стр _выр_2)
```

Условные операторы - удобный способ обойти сложные универсальные построения, разбивая их на несколько менее универсальных и более простых построений, действующих на узком диапазоне. Однако использование условных

операторов требует проверки методики на всех возможных диапазонах и сочетаниях размерных признаков, когда построение проходит по всем «веткам» вложенных условных операторов.

Следующий оператор, существующий в системе начиная с версии 6.7 и тоже подробно ранее не описанный, – оператор цикла. Форма записи оператора следующая:

*ц_начало;*
*оператор1;*
*оператор2;*
*оператор3;*
*……*
*ц_конец;*

При входе в цикл система последовательно выполняет все операторы до оператора "Ц_КОНЕЦ" и переходит на начало цикла, выполняя все операторы заново. Для выхода из цикла используется оператор "Ц_ПРЕКРАТИТЬ". Для перехода на начало выполнения цикла – "Ц_ПРОДОЛЖИТЬ". Циклы - обычный оператор для языков программирования. В традиционных языках программирования циклы используются для обработки массивов, списков, «деревьев» и т.д. В системе ЛЕКО циклы появились для организации нового метода конструирования - циклического построения.

Методика использования оператора цикла для конструирования одежды следующая:

*Задаем начальные параметры;*
*Ц_начало;*
*Необходимые построения;*
*Проверка условий выхода из цикла (ц_прекратить);*
*Изменение значений параметров;*
*Переход на начало цикла;*
*Ц_конец;*

Параметры построения и способ их изменения могут быть различными. Преимущества такого подхода в том, что не нужно думать о формулах; необходимо только указать способ, прием устранения недостатков наподобие схемы: «мало - прибавить, много - подрезать». Конкретных способов устранения дефектов и подгонки конструкции под заданные условия может быть бесконечно много. Основные критерии, по которым можно выбрать способ изменить параметры, это сходимость процесса и прозрачность для конструктора. Сходимость процесса означает, что «прибавляя и подрезая» 100-200 раз, система выполнит все условия сопряжения и подгонки участков и выйдет из цикла. Прозрачность для конструктора означает, что условия выхода из цикла и способ изменения параметров ему понятны.

Для более качественного оформления лекал введен оператор формирования надписей «нарисовать_текст»

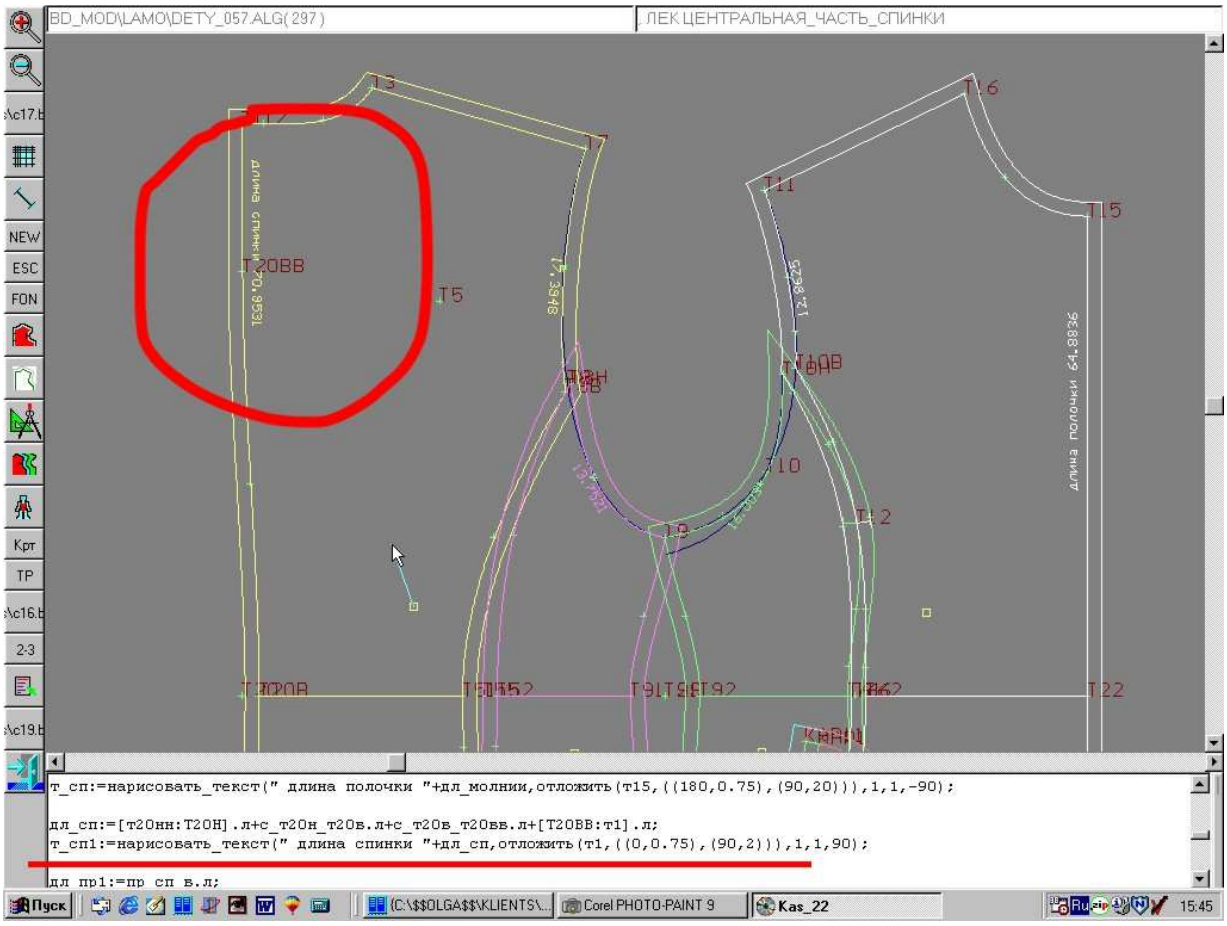
*нарисовать_текст(стр_выр, точка, ширина_сим, высота_сим, угол);*

60

где

*стр_выр* – строковое выражение,
*точка* – идентификатор или определение точки,
*ширина_сим*, *высота_сим* – ширина и высота символа текста,
*угол* – направление надписи.

Пример:




# 14.3. НОВЫЕ ВОЗМОЖНОСТИ ЗАПИСИ ОПЕРАТОРОВ

В операторах, запись которых требует координат точек, можно использовать вместо названий точек их определение, например:

*С1:=сплайн_к(точка(10,10), точка(15,40),0,90,1);*
*о1:=отрезок(точка(10,10), точка(15,40));*
*записать(имя=»квадрат»,*
*контур=( точка(0,0), точка(0,10), точка(10,10), точка(10,0)));*

Расширены функции операторов "разделить****" (разделть, разделить_н, разделить_кс, разделить_ксу). Часто эти операторы используются для модификации линии участка лекала (например, выравнивание длин или моделирование). При

использовании этих операторов, особенно разделить_н, нужно следить, чтобы соблюдались условия и точка разделения была на линии, иначе возникает ошибка построения. Если рассматривать градацию на широкий диапазон размеро-ростов или построение на индивидуальные размеры, то предсказать заранее все варианты взаиморасположения точек достаточно сложно. Для решения этой задачи вводятся расширения операторов разделения. Первое - это проверка условия, что разделение выполняется (без самого разделения)

```
если разделить_н(С_Т221_Т91Н,Н1,[ Т20НН:Т91НН ].ф) то
   разделить_н(С_Т221_Т91Н,Н1,[ Т20НН:Т91НН ].ф,н4,Т221_н4,н4_Т91Н);
иначе
   пересечение_н(Н1,[ Т20НН:Т91НН ].ф,Т91Н,С_Т221_Т91Н.ф2,н4);
конец_если;
```

Подразумевается, что мы знаем о том, что проблема может возникнуть в окрестности второго конца сплайна. Если условие выполняется и линию можно разделить, то мы ее делим, если не выполняется, то мы ищем точку пересечения направлений вдоль касательной во второй точке сплайна. В данном примере считалось, что нам нужна была только точка н4 и разделенные участки Т221_н4 и н4_Т91Н были не нужны для дальнейших построений. Если, допустим, нужен первый участок, то в случае отсутствия пересечения сплайна и направления мы можем оформить участок ломаной, как в следующем примере:

```
если разделить_н(С_Т222_Т92Н,Н4,0) то
   разделить_н(С_Т222_Т92Н,Н4,0,н3,Т222_н3,н3_Т92Н);
иначе
пересечение_н(Н4,0,Т92Н,С_Т222_Т92Н.ф2,н3);
   Т222_н3:=ломаная(С_Т222_Т92Н,н3);
конец_если;
```

Можно проблему отсутствия пересечения решить по другому: заранее продлить линию, так чтобы пересечение гарантировано было (продлили сплайн во второй точке вдоль касательной на 50 см)

```
Лн_1:=ломаная(С_Т222_Т92Н, отложить(Т92Н, С_Т222_Т92Н.ф2,50));
разделить_н(Лн_1,Н4,0,н3,Т222_н3,н3_Т92Н);
```

это можно записать как

```
разделить_н(ломаная(С_Т222_Т92Н,
отложить(Т92Н,С_Т222_Т92Н.ф2,50)),Н4,0,н3,Т222_н3,н3_Т92Н);
```

или опустить слово "ломаная"

```
разделить_н(  (С_Т222_Т92Н,  отложить(Т92Н,
С_Т222_Т92Н.ф2,50)),Н4,0,н3,Т222_н3,н3_Т92Н);
```

в этом случае мы теряем визуальный контроль за целиковой ломаной, но можем видеть ее компоненты. В данном случае целью является повышение качества и

надежности алгоритма, поэтому отсутствие целиковой ломанной можно считать допустимым.

Еще одна особенность последней записи оператора. Когда мы пишем

*разделить_н( С_Т222_Т92Н, Н4,0,н3,Т222_н3,н3_Т92Н);*

то точка н3 будет принадлежать участкам Т222_н3,н3_Т92Н и линии С_Т222_Т92Н.

при записи

*разделить_н( (С_Т222_Т92Н, отложить(Т92Н, С_Т222_Т92Н.ф2,50)),Н4,0,н3,Т222_н3,н3_Т92Н);*

точка н3 будет принадлежать участкам Т222_н3,н3_Т92Н но не будет принадлежать линии С_Т222_Т92Н, так как делится временно созданная ломаная. Это может быть важно при установке точек надсечек.

## 14.4. ОПЕРАТОР "ЗАПИСАТЬ"

Изменения оператора "ЗАПИСАТЬ" направлены на то, чтобы упростить операцию описания лекала. Первое косметическое изменение - поле "Имя": в качестве имени можно использовать строковое выражение.

В контуре лекала помимо имен точек и линий можно использовать их определения.

```
записать(имя=»квадрат»,
  контур=( точка(0,0), точка(0,10), точка(10,10), точка(10,0),
    сплайн_к(точка(10,10), точка(15,40),0,90,1)));
```

Точки надсечек могут не принадлежать контуру. В качестве точки надсечки берется проекция заданной точки на линию контура. С одной стороны, это удобно: не надо следить за включением точек надсечек в линию контура. С другой стороны, любая ошибочно заданная точка будет иметь проекцию на контур лекала, и сообщение об ошибке выдаваться не будет.

В качестве внутренних линий можно использовать оператор описания метки, например:

*Внтр=((метка(т1,4,90,1,1)), (метка(т2,4,90,1,1)), (метка(т3,4,90,1,1))),*

## 14.4. 3D ОПЕРАТОРЫ

Операторы для трехмерных построений появились достаточно давно, еще в версии 6.7. Но как другие "секретные" операторы не были представлены в описании.

В настоящий момент, трехмерные построения начали активно использоваться при конструировании лекал и снятии размерных признаков, поэтому необходимо дать описание основных операторов для трехмерных построений.

Трехмерные операторы работают аналогично плоскостным, только добавляется третья координата и самое главное (и сложное) понятие угол заменяется на понятие направление.

Угол определялся числом, а трехмерное направление определяется трехмерной точкой (три координаты), трехмерным отрезком (шесть координат) или через существующие направления. К величине угла можно было добавить или вычесть величину другого угла. С направлениями так не получиться. Направление можно повернуть в пространстве относительно другого направления на какой-то угол. Для того, чтобы понять, что получиться из этого поворота, нужно представить конус, который будет образовываться при вращении одного направления относительно другого.

Итак операторы (рассматриваем на примере манекена woman):

вх:=точка_3(10,0,0);

для задания трехмерной точки необходимо использовать ключевое слово точка_3 и задать три координаты. Кстати, слово точка_3 нельзя использовать в качестве индентификатора, как это дано в нашем описании. В приведенном операторе задается точка вх, которая в дальнейшем будет использоваться только как направление по оси X.

тог4:=отложить_3(т4,((ПВР_Н(ву,вх,-3),-рз_1/38*1)));

здесь два новых оператора. Отложить_3 - аналог оператора отложить для трехмерного пространства. Он отличается от плоскостного аналога только тем, что вместо угла задается направление. А вот оператор ПВР_Н (поворот направления) требует пояснения. В качестве параметров записываются: направление, которое мы хотим повернуть, направление, относительно которого мы поворачиваем, и угол поворота. Результат поворота подставляется на место оператора ПВР_Н.

Т.е. при записи

тог4:=отложить_3(т4,(ву,-рз_1/38*1)));

мы откладывали точку тог4 вертикально вниз от точки т4. А при записи

тог4:=отложить_3(т4,((ПВР_Н(ву,вх,-3),-рз_1/38*1)));

мы отложили не вертикально вниз, а чуть-чуть подвернули направление в плоскости Y-Z.

Следующий оператор

сп_ш_10:=сплайн_3(ТОГ5,т10,ву,ву,0.3);

полностью аналогичен плоскостному сплайну. Если точки и направления лежат в одной плоскости, то сплайн тоже будет плоским. Если точки и направления не лежат в одной плоскости, то сплайн становится объемным и понять его внешний вид можно только при повороте или автоматическом вращении.

СИММЕТРИЯ_3((сп_ш_1),плоскость(т10,вз),"л");

Плоской осевой симметрии в пространстве соответствует симметрия относительно плоскости. Плоскость задается точкой, через которую эта плоскость проходит, и нормалью к плоскости (направлением нормали). Нормаль - достаточно важное понятие, которое будет использоваться при развертке лекал на плоскости. На плоскости нормали не использовались. Были касательные и для получения перпендикуляра (нормали) можно было прибавить или вычесть 90 градусов. В пространстве при построении перпендикуляров мы получаем не линию а целую плоскость.

Следующий оператор

РАЗДЕЛИТЬ(сп_ас1,рз_39*0.5,т_202,сп_ас1_1,сп_гр234);

Даже по форме записи он не отличается от плоскостного. Делим плоскую линию - получаем плоские точку и две части линии. Делим пространственную - получаем пространственные.

Так же в операторе

отложить_в(СП_Т3,рз_18*0.15,т_36);

можно использовать плоские и пространственные линии
и это все операторы, которые нужны для построения сеточной модели манекена.
В следующем письме рассмотрим как "раскрасить" манекен.

# 14.4. УСТАРЕВШИЕ ОПЕРАТОРЫ

Как и в любом языке программирование в системе ЛЕКО так же есть устаревшие операторы и формы записи. Это значит, что пользоваться ими еще можно, но смене версий системы велика вероятность что они будут удалены.

В некоторых случаях гладкую линию сплайна необходимо «искривить» на некоторых участках. Для корректировки сплайнов под «ручные» кривые линии можно использовать оператор *сплайн_р*, т.е. разделить сплайн на любое количество частей и в каждой точке задать необходимые изменения:

*идентификатор сплайна:=сплайн_р[90] (Т1, Т2, кас1, кас2, К, ((к1,к2,к3,к4),( к5,к6,к7,к8),( к9,к10,к11,к12)….));*

где *Т1* и *Т2* - идентификаторы двух крайних точек сплайна,
*кас1, кас2* - арифметические выражения или числа, определяющие углы наклона касательных на концах сплайна,
*К* - коэффициент выпуклости,
*[90]* - количество отрезков, в данном случае – 90,
до скобок – это сплайн_к, в скобках записываются отклонения от него:
*к1,к5,к9* – коэффициенты, определяющие точки деления сплайна, задаются в пропорции от длины исходного сплайна,

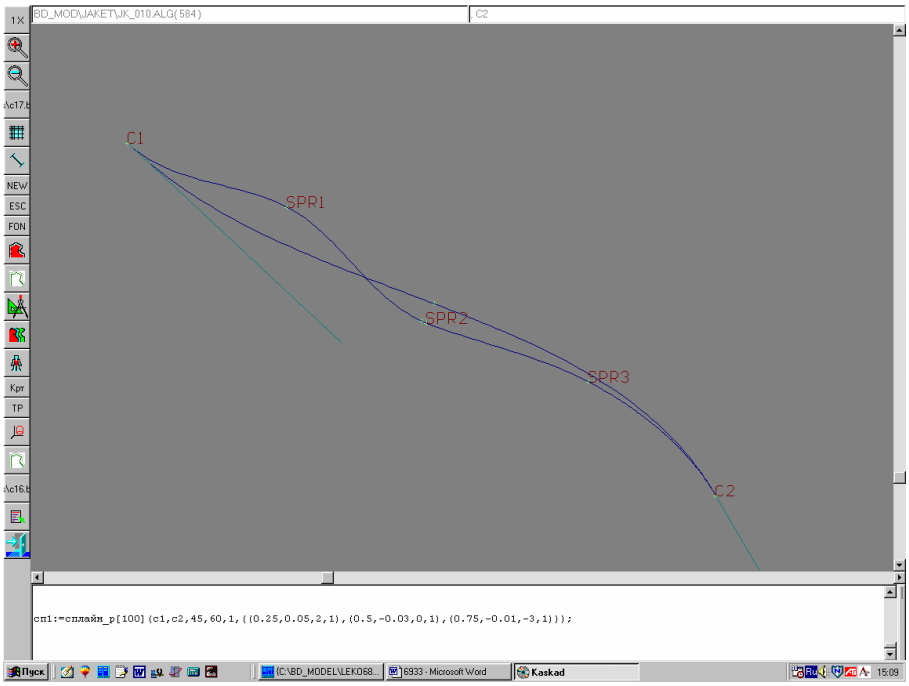
*к2,к6,к10* – коэффициенты, определяющие отклонение в точке деления сплайна, задаются в пропорции от длины исходного сплайна,

*к3,к7,к11* – коэффициенты, определяющие отклонение касательной соответствующего участка сплайна, задаются в градусах относительно исходного значения,

*к4,к8,к12* – коэффициенты выпуклости соответствующего участка сплайна.

Пример:

*сп1:=сплайн_р[100](с1,с2,45,60,1,((0.25,0.05,2,1),(0.5,-0.03,0,1),(0.75,-0.01,-3,1)));*




В программе восстановлены старые операторы <сплайн_т> и <разделить_кс>.

Оператор <сплайн_т> используется для <построения гладкой кривой, проходящей через точку>. Такую фразу часто пишут в методиках, ориентированных на ручное построение. При построении сплайна программа подбирает коэффициент выпуклости сплайна в операторе <сплайн_к> так, чтобы линия сплайна прошла как можно ближе к заданной точке. Порядок записи

сплайн_т(точка1, точка2, уг_кас_1, уг_кас_2, точка_касания);

следует учитывать, что этот оператор выполняется дольше, чем обычный <сплайн_к>, поэтому не рекомендуется использовать его в циклах.

Оператор <разделить_кс> - разделить линию касательной.

разделить_кс(ломанная, точка, точка_касания, часть1, часть2);

назначение этого оператора достаточно понятно - разделить ломанную линию на две части касательной проведенной из точки. С точки зрения чистой геометрии здесь все ясно. Но на практике линия может иметь произвольную форму, у нее может не быть касательной, или может быть несколько локальных касательных. При построении касательной система будет действовать следующим образом. Будет проводиться линия из заданной точки к первой, второй и т.д. точкам ломанной (напомним, что сплайн, дуга и даже отрезок - это тоже ломанные) и как только ломанная окажется по одну сторону линии, текущая точка ломанной и будет считаться точкой касания.

Укажем об особенностях этих операторов и причине, по которой они были в свое время удалены из описания. Все линии в компьютере описываются набором точек. Окружности, дуги, плавные кривые выглядят гладкими только потому, что состоят из большого числа отрезков. При большом увеличении можно разглядеть, что они состоят из отрезков. А так как линии состоят из отрезков, то операторы будут вести себя достаточно <дискретно>, т.е. небольшом изменении начальных данных результат будет меняться не непрерывно, а скачками (может и небольшими). В большинстве случаев это не будет оказывать существенного влияния на результат. Для уменьшения влияния <дискретности> обычно достаточно увеличить количество точек линии.

Несмотря на то, что операторы <официально> возвращены в язык, пользоваться ими нужно с достаточной степенью осторожности.

# ПРИЛОЖЕНИЕ

Элементы языка описания и построения лекал.

## Разделители :

:= ; [ ] . | { }

## Ключевые слова:

| КЛЮЧЕВОЕ СЛОВО | НАЗНАЧЕНИЕ |
|---|---|
| ТОЧКА | Оператор установки точки, может использоваться внутри других операторов |
| ОТРЕЗОК | Оператор соединения точек отрезком, на практике используется достаточно редко |
| ДУГА | Оператор построения дуги |
| | |
| СПЛАЙН_К | Построение сплайна с заданным коэффициентом выпуклости, рекомендуется для построения |
| СПЛАЙН_КК | Построение сплайна с заданным коэффициентом выпуклости и регулируемой степенью прилегания к касательным в первой / второй точке |
| СПЛАЙН_Р | Модификация сплайна под «ручные» линии |
| | |
| ЛОМАНАЯ | Оператор объединения нескольких кривых в одну непрерывную линию |
| | |
| ПЕРЕСЕЧЕНИЕ | Поиск пересечения двух объектов |
| ПЕРЕСЕЧЕНИЕ_Н | Поиск пересечения двух линий |
| ПЕРЕСЕЧЕНИЕ_Д | Поиск пересечения двух дуг |
| ПЕРЕСЕЧЕНИЕ_ДН | Поиск пересечения дуги и линии |
| | |
| ОТЛОЖИТЬ | Оператор построения точки |
| ОТЛОЖИТЬ_В | Оператор построения точки на кривой |
| СИММЕТРИЯ_О | Оператор центральной симметрии |
| СИММЕТРИЯ_Л | Оператор осевой симметрии |
| ПЕРЕНОС | Оператор параллельного переноса |
| ПОВОРОТ | Оператор поворота относительно точки |
| СЖАТЬ | Оператор сжатия/растяжения относительно точки в двух направлениях |
| РАЗДЕЛИТЬ | Оператор разделения сплайна/ломаной по длине |
| РАЗДЕЛИТЬ_Н | Оператор разделения сплайна/ломаной направлением |
| УДАЛИТЬ | Оператор удаления переменных |
| | |
| КОНЕЦ | Оператор окончания построения |
| | |
| РАЗМЕРЫ | Оператор чтения текущих размеров из базы данных |
| | |
| ЗАПИСАТЬ | Оператор записи лекала |
| ИМЯ | Имя лекала |
| КОД | код лекала |

| | |
|---|---|
| *ЦВЕТ* | *Цвет лекала* |
| *КОНТУР* | *Определение контура лекала по линии шва* |
| *ВНТР* | *Внутренние линии* |
| *ДОЛЕВАЯ* | *Направление долевой* |
| *НАДСЕЧКИ* | *Точки надсечек* |
| *ПРИБАВКА* | *Общая прибавка на швы* |
| *ПРИБАВКА_Т* | *Оформление уголков* |
| *ПРИБАВКА_У* | *Прибавка на швы по участкам* |
| *ПРИБАВКА_Л* | *Встраиваемая линия прибавки на швы* |
| | |

# Квалификаторы

| | |
|---|---|
| х | горизонтальная координата |
| у | вертикальная координата |
| $х_1$ | горизонтальная координата начальной точки |
| $у_1$ | вертикальная координата начальной точки |
| $х_2$ | горизонтальная координата конечной точки |
| $у_2$ | вертикальная координата конечной точки |
| $ф_1$ | угол касательной в начальной точке |
| $ф_2$ | угол касательной в конечной точке |
| к | коэффициент выпуклости сплайна |
| л | длина |

# КОНСТРУИРОВАНИЕ В СИСТЕМЕ ЛЕКО

***Принципиально новые возможности системы помогут Вам воплотить свои замыслы***

В первых частях описания системы ЛЕКО изложен общий порядок работы с системой, даны ее характеристики и возможности.

В предшествующей части описания системы ЛЕКО представлен язык описания построения лекал, даны синтаксис и семантика всех языковых конструкций, предназначенных для описания построения лекал, и приведены краткие ссылки на связь этих построений с конструированием и моделированием лекал швейных изделий.

В предлагаемой части описания приведены конкретные примеры алгоритмов построения лекала, даны рекомендации по конструированию и моделированию с учетом последующего технического размножения лекал по размеро-ростам (градации), параметризация построения лекал, "алгоритмизация" построения лекал по готовым образцам, описаны новые возможные подходы к конструированию и моделированию лекал швейных изделий с использованием системы ЛЕКО. В настоящей части описания системы ЛЕКО не рассматриваются общие вопросы конструирования и моделирования одежды, а описано лишь то, что касается конструирования и моделирования в системе ЛЕКО.

Как было сказано в первой части описания, в основе подхода к конструированию и моделированию одежды в системе ЛЕКО лежат расчетные методы конструирования. С одной стороны, такой подход не является новым, и очень многие методики построения лекал основаны на расчетах. С другой стороны, расчетные методы конструирования, ориентированные на ручное построение, не доведены до полностью формализованных схем, позволяющих любому пользователю построить на их основе требуемое лекало. Ориентация этих методик на самые простые инструменты построения (линейка и циркуль) и ручной расчет требуемых величин (даже без калькулятора) можно отнести к достоинству и недостатку таких методик. Как правило, методики подразумевают наличие у пользователя определенного опыта по доработке лекала и посадке изделия на фигуру.

Используемые в системе ЛЕКО методики построения лекал должны быть полностью формализованы, вся последовательность построения должна быть четко выражена в терминах системы. Это требует определенной квалификации конструктора, работающего с системой. Мощная вычислительная база системы ЛЕКО предоставляет новые возможности по созданию расчетных алгоритмов построения лекал, обеспечивающих точное соответствие фигуре, согласованность всех лекал изделия между собой, учет технологии изготовления изделия, автоматическое размножение лекал на любой размеро-рост.

Если Вы хотите качественно и быстро разрабатывать и внедрять новые модели швейных изделий - используйте систему ЛЕКО.

# 1. ПОСЛЕДОВАТЕЛЬНОСТЬ РАЗРАБОТКИ АЛГОРИТМА ПОСТРОЕНИЯ ЛЕКАЛА

Последовательность разработки алгоритма построения лекала швейного изделия, на наш взгляд, должна быть следующей:

- анализ внешнего вида или эскиза модели, предложенной художником, установка связи характерных точек и линий модели с характерными антропометрическими точками и линиями, например, в виде: "линия талии изделия завышена на 2 см относительно антропометрической линии талии", "пройма спущена на 1/2 до линии талии", "длина изделия на 3-5 см ниже колена", "рукав на 5-7 см ниже локтевой точки";
- перевод абсолютных значений (выраженных в сантиметрах) в доли соответствующего размерного признака (разработка лекала в пропорциях), например: "рукав на 5 см ниже локтевой точки (на фигуре 164/46)" преобразуется в "рукав на 5/(53.4-30.0)* (Дрзап-Дрлок) ниже локтевой точки", где 53.4 см (Дрзап) - длина руки до запястья для фигуры 164/96, 30.0 см (Дрлок) - длина руки до локтя для фигуры 164/96;
- разбор построения модели на конструктивные и модельные элементы;
- выбор наиболее подходящей стандартной или собственной методики построения основы для построения модели;
- доработка основы: выбор размерных признаков, относительно которых будет вестись конструирование, и контроль построения, согласование соединяемых участков, контроль за выбранными угловыми величинами и т.д.
- нанесение модельных элементов на конструкцию основы, контроль согласованности участков лекал;
- параметризация построения, конструирования и моделирования, учета технологических особенностей;
- макетирование отдельных деталей и участков, проверка на манекене, внесение изменений в построение или параметры;
- отшив образца, проверка на манекене, внесение изменений в программу;
- доработка текстового описания программы, документации: установка комментариев на отдельные участки построения; описание содержания, диапазона изменения и ограничений на настроечные параметры модели; описание допущений, используемых при построении.

Система ЛЕКО является универсальной программной инструментальной средой, позволяющей реализовать различные подходы работы с ней. Система не содержит каких-либо жестких неизменяемых методов. Практически каждое действие можно выполнить несколькими различными способами. Поэтому все наши пожелания относительно порядка работы с системой и методов построения лекал носят рекомендательный характер. Более того, так как мы предлагаем новый подход к конструированию лекал, основанный на широком применении расчетных формул и численных методов, то и приводимые рекомендации ориентированы в основном на этот подход. Если Вы хотите реализовать в системе ЛЕКО свой метод конструирования, то можете выбрать из предлагаемого Вам описания только то, что Вас заинтересует.

Для эффективной работы с системой ЛЕКО необходимо внимательно изучить ее возможности, особенно возможности записи построения на языке системы, методы записи построения и возможности по повышению качества отработки лекал. Первый

этап в обучении - это запись существующих методик построения лекал различных изделий на языке описания геометрических построений. Рассмотрим конкретный пример:

# 2. ОСНОВА КОНСТРУКЦИИ БРЮК

Если Вы внимательно прочитали два предыдущих раздела, то, конечно, заметили, что переложение хорошо написанной подробной "ручной" методики построения лекала на текст программы не вызывает никаких затруднений. При определенном навыке можно достаточно быстро разбираться в методике.

Затруднения начинают возникать тогда, когда в методике не указано точно, как именно необходимо проводить построение, а дается только краткое пояснение, каким должен быть конечный результат. Особенно часто это затруднение возникает при описании криволинейных участков. В системе ЛЕКО криволинейные участки - сплайны - задаются начальной, конечной точками и углами касательных. Если в методике нет никаких ссылок на углы касательных, то их необходимо определить, исходя из построения (например, когда кривая линия сопрягается с отрезком или линией) или по внешнему виду чертежа, особенно если явно видно, что касательная горизонтальная или вертикальная (хотя этот способ не очень надежен). В приведенном примере построения конструкции брюк ничего не сказано об углах линии среднего среза в точке Я2 (в точке Я51, судя по чертежу, можно считать касательную горизонтальной), шагового шва в точках Я2, Я51, бокового шва в точках Т6 и Т3. Отклонение касательной шагового шва в точках Я2, Я51 небольшое по величине, и эту величину отклонения касательной можно приблизительно принять как равную двойной величине угла между отрезком К4 _Я51 и Раб2_Я51, но по поводу угла касательной линии среднего среза в точке Я2 хотелось бы иметь рекомендации о его величине.

**Брюки**

| Построение чертежа основы | Ст:=34; |
|---|---|
| конструкции основные измерения фигуры:<br>- полуобхват талии Ст,<br>- полуобхват бедер Сб,<br>- длина брюк Дб. | Сб:=48;<br>Ди:=100;<br><br>Шк:=22;<br>Шн:=20; |
| Брюки различной формы строят на единой конструктивной основе. Форма их зависит от прибавок на свободное облегание на свободных участках. Ширину внизу определяет модный силуэт и индивидуальные данные фигуры. | Пст:=1;<br>Псб:=2;<br>Пди:=1; |
| Прибавки на свободное облегание: по талии и бедрам по 1 см | Пр_защипы:=6;<br>Пр_выт:=4.5; |
| Основа конструкции брюк дана с небольшим наполнением по линии бедер за счет защипов и дополнительного расширения передней половинки со стороны бокового шва. | Пр_колено:=5; |
| Строят прямой угол с вершиной в точке Т. От нее вниз откладывают:<br>отрезок Т_Н, определяющий длину | т:=точка( 0, 0);<br>н:=точка( т.х, т.у+Ди+Пди);<br>я:=точка( т.х, т.у+0.5*Сб+2); |

изделия;

отрезок Т_Я, определяющий высоту сидения.

Т_Н=Дб + П Т_Я=Сб/2 + 2 см.

От линии высоты сидения вверх откладывают отрезок, соответствующий положению линии бедер, и ставят точку Б.

```
б:=точка( т.х, я.у-(я.у-т.у)/3);
```

Я_Б= Т_Я/3

Уровень линии колена зависит от модели: для узких брюк он приближается к естественному уровню, для расширенных, как правило, смещается выше на 3-5 см.

```
к:=точка( т.х, б.у+0.5*(н.у-б.у)-
Пр_колено);
```

Б_К= Б_Н/2 - 5 - величина, равная завышению линии колена. Через точки Б, Я, К и Н проводят горизонтальные линии.

Передняя половина брюк. Ширина передней половинки брюк на линии сидения

```
я2:=точка(
я.х+0.5*(Сб+Псб)+0.1,я.у);
```

Я_Я2=1/2*(Сб+Пб)+0.1 см.

Ст= 1/2*(Сб+1)+0.1*Ст.

Линия сгиба передней половинки брюк.

```
я3:=точка( (я.х+я2.х)/2, я.у);
```

Я_Я3= 1/2*Я_Я2

Через полученную точку Я3 проводят вертикальную линию, пересечение которой с линиями талии, бедер, высоты сидения, колена и низа обозначают соответственно точками Т2, Б2, Я3, К1, Н1.

```
т2:=точка( я3.х, т.у);
б2:=точка( я3.х, б.у);
к1:=точка( я3.х, к.у);
н1:=точка( я3.х, н.у);
```

Влево от точки Я2 откладывают отрезок, равный ширине шага, и обозначают точкой Я1.

```
я1:=точка( я2.х-0.1*(Сб+Псб),
я.у);
отложить(я1, -45, 3.1, д);
```

Я2_Я1= 0.1*(Сб+Пб)

Из точки Я1 вверх к прямой Я_Я1 восстанавливают перпендикуляр, точки пересечения которого с горизонтальными линиями талии и бедер обозначают Б1, Т1.

```
т1:=точка( я1.х, т.у);
б1:=точка( я1.х, б.у);
```

Ширину брюк определяют по модели. При построении чертежа брюк, плотно прилегающих, дополнительно измеряют:

обхват бедер (Об) - на уровне подъягодичной складки строго горизонтально;

обхват колена (Ок) – через центр коленной чашечки строго горизонтально;

обхват под коленом (Опк) - под коленом строго горизонтально;

обхват икры (Ои) – в месте наибольшего развития строго горизонтально;

обхват щиколотки (Ощ) - под стопой в самом узком месте ноги.

Измерения используют в зависимости от формы и длины брюк. Ширина передней половинки брюк равна:

Н2_Н3 = Шн - 2см ,

где 2 см - разница между шириной задней и передней половинок брюк внизу. От точки Н1 влево и вправо по горизонтали откладывают отрезки Н1_Н2 и Н1_Н3 = 1/2*Н2_Н3.

Ширину по линии колена (Шк) определяют по модели. Она может быть больше или меньше ширины брюк внизу, но не меньше измерения обхвата ноги в колене (Ок).

Оформление бокового и шагового срезов. Находят вспомогательные точки. К1_К2 = К1_К3 (по модели).

Точки К2 и Н2 соединяют прямой линией, которую продолжают до пересечения с горизонталью из точки Я и обозначают точкой Я4. Соединяют прямой линией также точки К3 и Н3. От точки К3 влево откладывают отрезок К3_К31, равный 2 см.

Ширину передней половинки брюк на линии талии определяют следующим образом. От точки Т1 влево по Т11, равный 1 см. Данная величина может быть изменена в зависимости от выпуклости живота - Т1_Т11=1см.

По горизонтали от точки Т11 влево откладывают отрезок Т1_Т3, определяющий ширину передней половинки брюк на линии талии.

```
н2:=точка( н1.х+(Шн-2)/2, н1.у);
н3:=точка( н1.х-(Шн-2)/2, н1.у);
к2:=точка( к1.х+(Шк-2)/2, к1.у);
к3:=точка( к1.х-(Шк-2)/2, к1.у);
```

```
к31:=точка( к3.х-2, к1.у);
```

```
т11:=точка( т1.х-1, т.у);
```

```
т3:=точка( т1.х-
((Ст+Пст)/2+Пр_защипы),т.у);
```

Т11_Т3 = 1/2*(Ст+Пт)+Пр, где

Пр - припуск на защипы, равный 6 см (по 3 см на каждый). Прибавка по линии талии (Пт) равна 1 см. От точки Т3 вверх откладывают 0.5 см и ставят точку Т31.

```
т31:=точка( т3.х, т3.у-0.5);
```

Точки Т11 и Т31 соединяют плавной линией. Количество, глубина и оформление вытачек на линии талии зависят от формы брюк. Для данного варианта запроектированы два защипа по 3 см, один расположен по линии сгиба, второй - на 2 см от первого в сторону бокового среза.

Для оформления бокового среза передней половинки дополнительно находят отрезок, равный 3.5 см.

Б_Б0 = 3.5 см.

```
б0:=точка( б.х-3.5, б.у);
отрезок(т31, т11);
отрезок( к2, я2);
```

Боковой срез передней Половинки брюк оформляют через точки Т31, Б0, К31,Н3, как показано на чертеже.

```
пп1:=сплайн( к2, я2, пп1.ф,
пп1.ф+2*(о_к2_я2.ф-пп1.ф));
пп2:=сплайн_т( б1, я2, пп3.ф,
пп2.ф2+90, д);
пп3:=сплайн( т31,
т11,2*о_т31_т11.ф,
        пп3.ф-90);
пп4:=сплайн( т31, б0,
2*о_т31_т11.ф+90, 90);
```

Шаговой шов оформляют через точки Н2, К2 и Я2 плавной вогнутой линией. Для оформления линии среднего среза определяют точку Д1 на биссектрисе угла из Я1.

```
пп5:=сплайн( б0, к31,
90,пп6.ф+180);
пп6:=сплайн( н3, н2, -10, 10);
```

Я1_Д = 3.1 см.

Линию среднего среза проводят через точки Т11, Б1, Д и Я2 плавной вогнутой линией. Линия низа оформляется вогнутой линией. Величина прогиба посередине 1 см.

```
записать(имя=(передняя_полови
на_брюк),
контур=(пп1,-пп2,-пп3,пп4,
пп5,пп6),
цвет=11);
```

Задняя половинка брюк.

Положение среднего среза на линии талии определяет отрезок Т2_Т4, равный Т2 _Т1/2.

```
т4:=точка( (т1.х+т2.х)/2, т.у);
```

Из точки Т4 восстанавливают вверх перпендикуляр, на котором откладывают отрезок, определяющий баланс брюк.

Т4_Т5 = 0.1*Сб -1.5см.

```
т5:=точка( т4.х, т4.у-(0.1*Сб-1.5));
```

От точки Я1 влево откладывают 1 см и ставят точку Я11.

```
я11:=точка( я1.х-1, я.у);
```

Я1_Я11 = 1 см.

Точки Т5 и Я11 соединяют Вспомогательной прямой и на пересечении с линией бедер ставят точку Б3. Ширину шага задней половинки брюк определяет отрезок Я1_Я5, который откладывают от точки Я1 вправо.

Я1_Я5 = 0.2(Сб+Пб).

Для оформления среднего шва находят вспомогательные отрезки Я2_ Д1, равный 1 см (на биссектрисе), и Я2_Я21, равный 1.3 см, который откладывают вниз по шаговому срезу передней половинки.

Я5_Я51 = 2 см вниз по перпендикуляру.

Затем определяют ширину задней половинки брюк по линии бедер.

Б3_Б4 = (Сб+Пб) - Б_Б1.

Ширину задней половинки брюк на линии талии Т5_Т определяют следующим образом:

Т5_Т6 = 1/2*(Ст+Пт)+Пр+0.5см,

где Пр - припуск на вытачку, равный 4.5 см.

Положение точки Т6 находят пересечением двух дуг: дуги из точки Б4 радиусом, равным Б0_Т31, и дуги из точки Т5 радиусом, равным Т5_Т6.

Чтобы определить ширину задней половинки брюк на уровне линии колена (К), к чертежу добавляют отрезки.

К2_К4 = 2 см и К31_К5 = 1.5 см.

Ширину задней половинки брюк по линии низа увеличивают относительно половинки на отрезки, равные Н2_Н4 и Н3_Н5 =2см.

Боковой срез оформляют выпуклой линией через точки Т6, Б4, К5 от колена вниз к точке Н5 прямой линией, линию низа - прямой линией.

Для оформления шагового среза точку Я31 соединяют с точкой К4, затем находят точку 1.

1=Я51_К4/2.

От точки 1 влево откладывают 1.3 см и получают точку 2. Шаговой срез оформляют вогнутой линией от точки Я51,

```
отрезок ( б, б1);
отрезок ( т5, я11);
пересечение(о_б_б1, о_т5_я11, б3);
я5:=точка( я1.х+0.2*(Сб+Псб), я.у);

отложить(я2, 135, 1, д1);

я51:=точка( я5.х, я5.у+1.2);

б4:=точка(б3.х-(Сб+Псб-(б1.х-б0.х)), б.у);

отрезок ( б0, т31);
дуга_п( т5, 0.5*(Ст+Пст)+Пр_выт+0.5, б4, о_б0_т31.л, -1, т6);

к4:=точка( к2.х+2, к.у);
к5:=точка( к31.х-1.5, к.у);

н4:=точка( н2.х+2, н.у);
н5:=точка( н3.х-1.5, н.у);

отрезок ( я51, к4);
отложить(я51, о_я51_к4.ф, о_я51_к4.л/2, раб1);
отложить(раб1,о_я51_к4.ф+90,1.3,раб2);
```

| | |
|---|---|
| 2, К4 и от К4 до Н4 по прямой.<br>Вытачку проектируют посередине верхнего среза, перпендикулярно к линии талии<br><br>Т6_Т7 = Т6_Т5/2.<br><br>Длина вытачки - 12 см, раствор вытачки - 4 см. Ширина бочка кармана вверху - 2.5 см. Ширина пояса - 3 см. | отрезок ( т5, т6);<br>отложить(т5,о_т5_т6.ф, о_т5_т6.л/2,т7);<br>отложить(т7, о_т5_т6.ф-90, 12, т8);<br>отложить(т7,о_т5_т6.ф,-Пр_выт/2,т7с);<br>отложить(т7,о_т5_т6.ф,Пр_выт/2, т7сс);<br>зп1:=сплайн_т( к4, я51, зп2.ф, о_я51_к4.ф+180+1.3/о_я51_к4.л*2*рад, раб2);<br>зп2:=сплайн_т(б3,я51,зп4.ф,0,д1) ;<br>зп3:=сплайн(т6,б4,зп9.ф-90+10,90);<br>записать(имя=(задняя_половина _брюк),<br>контур=(н5,н4,зп1,-зп2,т5,т7с,т8, т7сс,зп3,к5),<br>цвет=14); |

Мы рассмотрели пример реализации на языке методик построения изделий. Вы, возможно, обратили внимание на то, что, кроме непосредственно функций построения различных геометрических фигур (точка, отрезок, дуга и т.д.), в программах часто используются формулы.

# 3. ЗАПИСЬ ФОРМУЛ

Система ЛЕКО представляет собой мощный вычислительный инструмент, позволяющий при построении вычислять формулы любой степени сложности. Возможность вычислять формулы позволяет использовать не приближенные формулы расчета, основанные на заранее рассчитанных коэффициентах (использующие "чужие" аппроксимационные коэффициенты, предлагаемые в какой-либо методике), а вводить и получать свои, точные формулы расчета. Основной вопрос в том, откуда брать эти формулы. При обучении, а также в существующих методиках, как правило, приводятся только приближенные формулы (причем обычно не поясняется, как эти формулы были получены). Точные формулы Вы можете получить, если подумаете о взаимозависимостях при построении лекала и расположении линий (швов) изделия в пространстве. Такой подход может потребовать от Вас знания тригонометрии, соотношений сторон и углов в треугольнике, планиметрии, и это может разочаровать в нем. Однако спешим заверить:

***хотите Вы или нет, Вы будете соблюдать все эти соотношения, однако путь к этому - через расчет формул или подбор и отшив пробных образцов - Вы можете выбрать сами. Законов тригонометрии и планиметрии никто не отменит. Учесть эти законы явно при построении позволяет система ЛЕКО, предоставляя вычислительный инструмент.***

Использование некоторых законов тригонометрии позволяет сократить запись алгоритма и сделать программу более понятной. Рассмотрим некоторые из них.

Перевод линейных величин в угловые и наоборот.

При ручном построении лекал конструктор чаще всего оперирует линейными величинами, даже в случаях работы с вытачками для построения берется не угол вытачки, а раствор вытачки в сантиметрах. Однако линейными величинами не всегда удобно пользоваться при разработке лекал, тем более что система располагает мощным инструментом работы с угловыми величинами: отложить вдоль направления, построить касательную, повороты, касательные сплайнов. Поэтому часто в программах возникает необходимость перевода линейных величин в угловые. Покажем, как это можно сделать.

При малых углах (до 30 градусов) величина раствора вытачки (величина отведения линии и т.п.) может быть определена как произведение длины и угла раствора, выраженный в радианах. Если длина вытачки L, а раствор D, то угол раствора вытачки будет равен:

```
U:=D/L*53.27
```

Например, если Вы хотите построить плечо с заданной длиной вытачки и заданным раствором, то такое построение может выглядеть следующим образом:

*o_m11_m14* - отрезок с нераскрытой вытачкой

```
{ находим точку вытачки на плече }
отложить( m11, o_m11_m14.ф, o_m11_m14.л/3, m_выт_1);
{ находим центр вытачки
  (от первой точки вертикально вниз на длину вытачки) }
m_выт_ц:=точка( m_выт_1.x, m_выт_1.y+L);
{ раскрытие вытачки }
поворот((m_выт_1,m14),m_выт_ц,D/L*53.27,(m_выт_2,m14c);
{ оформление плеча и вытачки отрезками }
отрезок(m11,m_выт_1);
отрезок(m_выт_1,m_выт_ц);
отрезок(m_выт_ц,m_выт_2);
отрезок(m_выт_2,m14c);
```

Кроме приведенной нами формулы существует множество других способов вычисления угла. Достаточно вспомнить программу средней школы:

Соотношения в прямоугольном треугольнике.

Нами был рассмотрен случай малых углов. Если же используются углы более 30 градусов, то необходимо при расчетах пользоваться тригонометрическими функциями SIN, COS и TN. Напомним Вам основные формулы. Если обозначим:

ABC - прямоугольный треугольник (прямой угол в точке B)

тогда

AC = Sqrt(AB*AB+BC*BC);
AB = Sqrt(AC*AC-BC*BC);
BC = Sqrt(AC*AC+AB*AB);
AB = AC * Cos(A);
BC = AC * Sin(A);
AB = AC * Sin(B);
BC = AC * Cos(B);
AC = AB / Cos(A);
AC = BC / Sin(A);
BC = AB / Tn(A);
BC = AB * Tn(B);
AB = BC / Tn(B);
AB = BC * Tn(A);

Этих формул не так уж и много, и, кроме того, поверьте: на практике Вы будете использовать лишь малую их часть.

Приведем примеры применения этих соотношений. Пусть требуется провести построение: точку Г2 отложить на 10 см вправо от точки Г1, точку Л6 отложить вверх от точки Г2 так, чтобы расстояние от Г1 до Л6 было равно рз12/2. Такое построение может быть записано как:

*Г2:=точка(Г1.х + 10, Г1.у);*
*Л6:=точка(Г2.х , Г2.у - Sqrt((рз12/2)*(рз12/2)-(Г1.х-Г2.х)*(Г1.х-Г2.х)));*

Можно записать это построение с использованием дополнительной переменной - вычисленного расстояния между точками Г2 и Л6:

*Г2:=точка(Г1.х + 10, Г1.у);*
*р_Г2_Л6:=Sqrt((рз12/2)*(рз12/2)-(Г1.х-Г2.х)*(Г1.х-Г2.х));*
*Л6:=точка(Г2.х , Г2.у - р_Г2_Л6);*

При ручном построении необходимо было бы для поиска точки Л6 дополнительно провести вертикальную линию или отрезок из точки Г2, дугу из точки Г1 и найти их пересечение. В программе это было бы записано как:

*Г2:=точка(Г1.х + 10, Г1.у);*
*Л6_п:=точка(Г2.х, Г2.у+10);*
*отрезок(Г2, Л6_п);*
*дуга(Г1, 10, 270, 360);*
*пересечение(о_Г2_Л6_п, д_Г1, Л6);*

Особый случай прямоугольного треугольника - равнобедренный прямоугольный треугольник. Углы при гипотенузе в нем составляют 45 градусов, и соотношение гипотенузы и катетов (SIN и COS) равно 1/Sqrt(2) = 0.7071 (обратная величина равна Sqrt(2)=1.414). Например, часто для криволинейных участков лекал устанавливаются промежуточные точки в виде: от точки Г2 отложить 3.5 см по биссектрисе прямого угла и поставить точку Г6. Такая операция может быть записана как

*отложить(Г2, 45, 3.5, Г6);*

или через преобразование координат:

*Г6:=точка(Г2.х + 3.5*0.71, Г2.у + 3.5*0.71);*

Линейные функции.

При описании построения лекал часто используются линейные зависимости, записываемые в виде:

*расстояние = постоянная + коэффициент * размерный_признак*

К такому виду приводятся многие соотношения, приводимые в методиках, и использование более сложных зависимостей при конструировании лекал не всегда целесообразно.

Зная значение расстояния для двух размеров, Вы можете определить значения "постоянной" и "коэффициента" и затем определять это расстояние для любого значения размера. Если Вам известны: расстояние1, размерный_признак1, расстояние2, размерный_признак2, то

*коэффициент =*
*(расстояние1-расстояние2)/(размерный_признак1-размерный_признак2)*
*постоянная = расстояние1 - коэффициент * размерный_признак1*

Рассмотрим, как найти "постоянную" и "коэффициент" в приведенном выражении, если Вы имеете готовые лекала. Допустим, у Вас два лекала - на 46 и 50 размеры. Вы хотите найти зависимость ширины лекала от размерного признака Ог3 (обхват груди 3).

*Ог3_46 = 92* *Шл_46 = 97*
*Ог3_50 = 100* *Шл_50 = 104*
*к = (97-104)/(92-100) = 0.825*
*п = 97 - 0.825*92 = 16.5*

т.е. ширина лекала изделия будет определяться по формуле

*Шл := 16.5 + Ог3 * 0.825;*

Эта формула на размере 46 даст 101 см, на размере 50 - 104 см, а на промежуточных значениях:

| Размер | Ширина | Ширина-Ог3 |
|---|---|---|
| **42** | 90 | 6 |
| **44** | 93.5 | 5.5 |
| **46** | 97 | 5 |
| **48** | 101.5 | 4.5 |
| **50** | 104 | 4 |

| | | |
|---|---|---|
| **52** | 107.5 | 3.5 |
| **54** | 111 | 3 |

# 4. ИСПОЛЬЗОВАНИЕ БАЗЫ ДАННЫХ РАЗМЕРНЫХ ПРИЗНАКОВ

Создание лекал на типовые фигуры, в отличие от индивидуальных фигур, удобно тем, что известны достаточно точно измеренные размерные признаки и имеются манекены, на которых можно, не торопясь, отработать посадку изделия. Наличие в системе ЛЕКО доступа к базе данных размерных признаков позволяет использовать в формулах любые размерные признаки, инициализированные в базе данных для любого типа фигуры (размерные признаки по ГОСТу и ОСТу и те, которые Вы заведете самостоятельно). Существующие методики, как правило, используют подмножество размерных признаков, рекомендуемых ОСТом для разработки моделей швейных, трикотажных и меховых изделий. Точное измерение и достаточно хорошая согласованность между собой размерных признаков позволяет использовать их практически без дополнительной обработки (чего нельзя делать при построении лекал по индивидуальным размерным признакам).

При построении отдельных участков лекал необходимо использовать соответствующие этим участкам размерные признаки. Например, при построении плеча следует использовать размерный признак "ширина плеча" (ном.31), добавляя или вычитая модельную и технологическую прибавку.

Некоторые размерные признаки можно использовать как контрольные при отработке изделия. Например, если при построении плеча Вы не использовали размерный признак "высота плеча косая" (ном.41), то этот признак может использоваться для дополнительного контроля построения. Вы можете вычислить высоту плеча косую, получившуюся на Вашем лекале по построению и по разнице (или относительной разнице) между точным и расчетным значением определить, правильно ли проведено построение. Допустим, Вы строили модель с учетом подплечника, но построенная величина оказалась равной расчетной. Это должно навести Вас на мысль, что подплечник или не поместится, или изменит проектируемую посадку изделия.

Проверка расстояний и длин в лекале до того, как оно будет напечатано или выведено на графопостроитель, значительно сократит время на поиск грубых просчетов при построении лекала, позволит Вам перейти к примерке при достаточно проработанном построении.

Наша система предлагает базу данных для проектирования швейных изделий. В базу данных включена информация по размерной типологии взрослого населения по ОСТ 17325 - 86 "Изделия швейные, трикотажные, меховые. Фигуры мужчин типовые. Размерные признаки для проектирования одежды", по ОСТ 17326 - 81 "Изделия швейные трикотажные, меховые. Фигуры женщин типовые. Размерные признаки для проектирования одежды".

База данных содержит все необходимые величины размерных признаков для конструирования одежды. База данных размерных признаков и методов их измерения соответствует действующим государственным и отраслевым стандартам.

Если же Вы пользуетесь другой базой размерных признаков, то можете ввести ее сами. Для этого предусмотрен раздел "Работа с базой размерных признаков".

# 5. "АЛГОРИТМИЗАЦИЯ" ПОСТРОЕНИЯ ЛЕКАЛА

"Алгоритмизация" построения лекала необходима, если у Вас есть отработанное лекало на базовый размеро-рост и Вы хотите ввести его в систему ЛЕКО, чтобы в последующем размножать его на другие размеро-роста (не только на смежные). Для этого Вам необходимо найти закономерности изменения характерных точек лекала в зависимости от размерных признаков. По описанным выше причинам проще "алгоритмизировать" лекало на типовую фигуру. Сначала необходимо найти коэффициенты, связывающие основные размеры лекала с размерными признаками, т.е. перейти от абсолютных значений расстояний к описанию лекала в пропорциях. Рассмотрим это на примере изделия с втачным рукавом.

Коэффициенты, определяющие связь "ширина плечевого шва изделия/измеряемая ширина плеча (ном.31)", "обхват груди изделия (1, 2, 3)/измеряемый обхват груди (ном.14, 15, 16)" находятся простым делением величин, измеренных по лекалу и взятых из базы данных (эту операцию удобно вести в подсистеме работы с базой данных размерных признаков в режиме калькулятора). Далее определим коэффициенты, связывающие высоту оката и высоту линии проймы (ном. 39) при высокой пройме. Если линия проймы занижена, то вместо высоты линии проймы можно использовать сумму высоты линии проймы и высоты линии талии. Если пройма изделия доходит до линии талии, то необходимо брать для расчета коэффициента высоту линии талии. При пересчете лекала на другой размеро-рост в зависимости от увеличения или уменьшения величины высоты линии проймы пропорционально будет изменяться высота оката и всех промежуточных точек. Для вычисления изменения ширины оката рукава может использоваться коэффициент, рассчитанный по исходной ширине проймы и передне-заднему диаметру руки (ном.57).

Описанная выше последовательность действий подобна заданию правил в существующих системах градации лекал (через задание приращений конструктивных точек), только отличается большей наглядностью и более высокой точностью и, следовательно, будет действовать на большем диапазоне размеро-ростов.

# 6. ПОСТРОЕНИЕ ЛЕКАЛ С ИСПОЛЬЗОВАНИЕМ УГЛОВЫХ ВЕЛИЧИН

Угловые величины, используемые при построении лекал (углы наклона плеча, рукава, углы раствора вытачек) отличаются достаточной стабильностью и слабо меняются при изменении не только размеро-ростов, но и типов фигур. Например, угол наклона плеча (определяемый шириной плеча и высотой плечевой точки относительно точки основания шеи) меняется от 23 градусов для 44 размера до 21 градуса для 60 размера. Поэтому построение лекала с использованием угловых величин, рассчитанных по величинам размерных признаков, позволяет писать универсальные программы построения лекал, пригодные при построении любых изделий. Вы можете провести эксперимент и проверить получающиеся углы на лекалах Вашего

ассортимента. Скорее всего, Вы достаточно просто сумеете определить коэффициент зависимости углов вытачек в зависимости от типа фигуры в следующем виде:

поправка:=размерный_признак * коэффициент;
угол_выт:= постоянный_угол + поправка;

Например, рассмотрим данные по таблицам, представленным в книге Братчик И.М. "Легкая женская одежда":

| размер | раствор нагрудной вытачки | высота груди |
|---|---|---|
| 44 | 9 | 25.6 |
| ..... | ..... | ..... |
| 60 | 17 | 32.5 |

При пересчете величины раствора нагрудной вытачки в величину угла получим следующую таблицу:

| размер | угол раствора нагрудной вытачки | соотношение уг/Сг2 |
|---|---|---|
| 44 | 19 | 0.43 |
| ..... | ..... | ..... |
| 60 | 28 | 0.46 |

Как видите, угол раствора изменяется в меньших пределах, а соотношение угол/размер практически постоянно.

Удобная работа с угловыми величинами в системе ЛЕКО и большая стабильность угловых величин в зависимости от размеро-роста делают перспективным подход разработки конструкций лекал швейных изделий с широким использованием угловой информации.

Фирма "Вилар" планирует провести анализ угловых величин, получаемых при использовании различных методик построения лекал, на всем диапазоне антропометрических данных и выпустить отчет и рекомендации по использованию угловой информации при построении лекал швейных изделий.

# 7. ПАРАМЕТРИЗАЦИЯ ПОСТРОЕНИЯ МОДЕЛЕЙ

Как было сказано в первой части, система ЛЕКО позволяет разрабатывать параметризованные алгоритмы моделей, которые позволяют изменять внешний вид модели путем изменения величины отдельных настроечных параметров. В зависимости от желания, возможностей и разрабатываемой модели число настроечных параметров и вид настройки может значительно меняться: от простого изменения высоты манжета до общего стиля модели. Например, для кокетки на спине могут быть введены следующие параметры:

| *вид детали* | *параметры* |
|---|---|
| прямая кокетка | высота кокетки |
| симметричная кокетка (низ) | высота кокетки |
| Оформлен двумя отрезками | угол (у проймы или середины спины) |
| симметричная кокетка (низ) | высота кокетки |
| Оформлен двумя кривыми | два угла у проймы и у середины спины |

Параметризация построения позволяет разрабатывать не одно, а целую серию изделий, отличающихся величинами прибавок, коэффициентами посадки, высотой и шириной деталей, сохраняя количество и взаимное расположение деталей неизменным, и затем в зависимости от требуемых характеристик модели задавать эти параметры. Приведем пример заголовка программы с настроечными коэффициентами:

```
{ платье с двумя кокетками }
{ Платье прямого силуэта, застегивается на пять пуговиц, с  подплечниками. Длина плечевого среза увеличена до 14 см.(о121_14). Полочки с кокетками, параллельными плечевому срезу, на нижней части  полочек защипы по срезу кокетки. Спинка с кокеткой (нижний срез кокетки параллелен линии низа). На нижней части спинки по срезу кокетки сборка. Рукав со сборкой между точками кокеток, на манжете.  }
{ ---------------------------------------------------------- }
{ основа построения - доработанная БК платья для женщин - ЕМКО СЭВ }
{ ---------------------------------------------------------- }
{  настроечные параметры модели выбираются по желанию заказчика    }
{ ---------------------------------------------------------- }
{ длина плечевого среза  }
дп:=14.0;
{ ширина  кокетки на спинке }
ш_к_сп:=8.0;
{ высота подплечника }
в_пп:=1.5;
{ посадка спинки }
пс:=1.5;
{ высота кокетки на переде }
в_кп:=6.0;
{ расстояние между защипами на полочке }
рз:=2;
{ припуск к линии полузаноса }
```

```
плп:=3;
{ расположение петель }
{ расстояние от точки основания горловины полочки до верхней петли }
н:=2.0;
{ расстояние между петлями }
рп:=13.0;
{ ширина петли }
шп:=1.2;
{ припуск к высшей точке оката - изменение высоты оката }
по:=1.0;
{ коэффициент посадки оката рукава }
пор:=1.5;
{ коэффициент посадки низа рукава }
пнр:=1.5;
{ длина манжеты }
дм:=26.5;
{ ширина манжеты }
шм:=5;
{ расширение оката рукава }
расш_о:=10;
{ расширение низа рукава }
расш_н:=8;
```

Возможны более гибкие настроечные коэффициенты, которые меняют внешний вид изделия. Например, в основе куртки, разработанной фирмой "Вилар", в качестве настроечного коэффициента задавался угол отклонения рукава от вертикальной оси изделия. Задавая значение угла из определенного диапазона (10-80 градусов), можно получить рукав рубашечного покроя (небольшая высота оката, величина угла рукава 50-80 градусов) или типа пиджачных изделий (большая высота оката, величина угла рукава - 10-30 градусов).

# 8. СТАНДАРТИЗАЦИЯ МЕТОДОВ ПОСТРОЕНИЯ И ОПИСАНИЯ

Отдельные методы построения могут быть стандартизованы в рамках предприятия исходя из вида швейных изделий, оборудования и технологии, применяемой на предприятии. Например, если предприятие специализируется на выпуске мужских костюмов и имеет специализированное оборудование для выполнения отдельных операций, то конструктор может заложить специфику этого

оборудования в программу при помощи коэффициентов, припусков и преобразований и затем использовать эти типовые построения в дальнейших разработках.

С точки зрения системы ЛЕКО, в организации работы со стандартизованными построениями может помочь использование внешних файлов. Например, Вы можете создать поддиректорию \OSNOVA и записать в нее алгоритмы построения основ по всему ассортименту выпускаемых Вашим предприятием изделий. Затем, чтобы включить это построение в свою методику, Вам необходимо написать команду "выполнить", указав полное имя файла или имя файла относительно текущей директории, например:

*выполнить(OSNOVA\vsh_r.alg);*

и в тексте Вашего построения будет алгоритм построения основы с вшивным рукавом. Если в различных файлах Вы будете хранить различные построения, то меняя названия файлов в команде "выполнить", Вы будете получать моделирование на различных основах. Еще одно преимущество такого метода хранения построения основы заключается в том, что если Вы доработали методику построения основы и записали ее в файл vsh_r.alg, то это улучшение попало во все построения, использующие этот файл.

Важным элементов стандартизации методов построения и описания моделей является

# 9. УНИФИКАЦИЯ ОБОЗНАЧЕНИЙ

Прежде чем Вы перейдете к созданию базы данных по моделям, Вам необходимо решить вопрос унификации обозначений: выработать методику наименования размерных признаков, прибавок, точек, линий, отрезков в соответствии с конструктивными поясами или какими-либо другими соображениями. Например, размерные признаки в ЕМКО СЭВ используются в основном по порядковому номеру, в соответствии с ГОСТ 17521-72. Использование условных обозначений размерных признаков, принятое в ОСТ17-325-86 и ОСТ17-326-81, в системе ЛЕКО не совсем удобно т.к. использует и русские, и латинские буквы. В книге Братчик И.М. "Легкая женская одежда" приводится своя методика наименования размерных признаков и прибавок, основных точек. На наш взгляд, такую методику можно взять за основу при создании своей методики обозначений.

Обозначение точек можно производить в соответствии с конструктивными поясами (например, Братчик И.М.) или используя сетку (ЕМКО СЭВ). Основное, что необходимо учитывать - чтобы обозначения конструктивных точек не изменялись от модели к модели. Это даст Вам возможность быстро и удобно собирать построение новой модели из имеющихся текстов программ, не меняя обозначений и затрачивая минимум усилий на их согласование (на уровне имен переменных).

При выборе методики обозначения отрезков, линий, дуг и сплайнов следует учитывать, что система по умолчанию сама дает им названия, собирая их (названия) из названий начальной и конечной точек.

# 10. НАКОПЛЕНИЕ ОПЫТА

Накопление опыта, его передача и использование является одним из важнейших технологических требований при промышленном производстве. Существовавшие в конструировании и моделировании одежды методы накопления опыта позволяли делать эффективное накопление только индивидуального опыта конкретного конструктора, а обобщение и передача его осуществлялась в основном в виде методик, статей и книг. Это достаточно долгий путь, и конечное представление - статьи и книги - является пассивной формой накопления знания, в то время как алгоритм, записанный в виде программы и позволяющий просмотреть порядок построения, а также построить и модифицировать лекало, является более активной формой накопления опыта. Словесное описание методики построения лекала допускает неоднозначность и неточность выражения; программа в системе ЛЕКО - только однозначное толкование. Возможность хранения не только конечного результата - лекала, но и алгоритма построения, замечаний и обоснования записанных в виде комментариев в программе позволяет Вам в любой момент перейти к просмотру построения любой модели, скопировать построение элементов лекала, методов, последовательность построения или просто часть текстового описания.

# 11. ТЕХНИЧЕСКОЕ РАЗМНОЖЕНИЕ ЛЕКАЛ

Построение лекала на основе алгоритма, осуществляющего построение лекала в пропорциях, позволяет значительно проще решить задачу технического размножения лекал по размеро-ростам (градации), а именно строить лекало на любой требуемый размер. При этом не нужно задавать направления изменения положения точек (определяемые недостаточно точно), т.к. эти изменения определяются алгоритмом построения и размерными признаками.

Для хорошего технического размножения лекал по размеро-ростам необходимо соблюдать все рекомендации по разработке и "алгоритмизации" построения лекала, и в этом случае вопроса градации лекал как отдельной задачи просто не возникнет.

# 12. ВЕДЕНИЕ БАЗЫ ДАННЫХ ПО МОДЕЛЯМ

Хранение моделей в виде алгоритма позволяет в любой момент построить требуемое лекало или скопировать элементы построения любого лекала для создания новой модели. Хранение алгоритмов построения лекал может осуществляться на дискетах или жестком диске (винчестере). На дискетах целесообразно хранить все копии алгоритмов, а на жестком диске - те лекала, которые могут потребоваться Вам для работы. В этом случае они практически не занимает места и всегда доступны при работе с машиной. Алгоритмы могут быть распечатаны и сшиты в брошюру, тогда они будут подобны существующим методикам. Используя распечатанные алгоритмы, можно создавать новые модели, выбирая необходимые фрагменты существующих построений, и дополняя их необходимым техническим моделированием.

# 13. САМОДОКУМЕНТИРУЕМОСТЬ

Хорошо написанная программа, как правило, не требует дополнительных материалов для пояснения действий, производимых в ней. Это достигается удачной последовательностью операций, выбором понятных обозначений объектов, подробными комментариями. Дополнительно может документироваться основание для действий, производимых программой. Например, если в программе рассчитывается коэффициент посадки ткани и проводится построение на основе этого коэффициента, то порядок использования коэффициента в дополнительных комментариях не нуждается, а основание для его расчета (статья, методика, диссертация и т.д.) могут быть указаны в ссылке перед расчетом формулы. Основой для самодокументируемости является то, что расчет, построение лекала и методика построения лекала, записанная в виде программы, в системе ЛЕКО неразделимы. Все преимущества самодокументирования Вы оцените, когда в процессе работы начнете возвращаться к своим старым разработкам и использовать их фрагменты для создания новых моделей.

Одна из основных предпосылок к самодокументированию программ в системе ЛЕКО - простой и понятный язык описания построения лекал (язык программирования). Опыт обучения и работы с конструкторами, ни разу не работавшими с компьютером, показывает, что язык описания построения в системе ЛЕКО отвечает требованиям самодокументируемости и удобен при записи методик построения лекал.

# 14. ПРОДАЖА АЛГОРИТМОВ

Хорошо разработанный алгоритм построения лекала может служить объектом продажи. В зависимости от используемой основы конструкции, качества моделирования, подробности описания построения лекала, такой алгоритм может вызвать значительный интерес у других швейных предприятий, особенно тех, которые работают с системой ЛЕКО, и достаточно дорого оцениваться. В отличие от комплекта лекал, вырезанных из картона, алгоритм построения лекала позволяет не только в точности воспроизвести лекало на базовый размеро-рост, но и модифицировать его, доработав под имеющуюся технологию, ткань и т.д. Если разработанный алгоритм параметризован, то, приобретая его, покупатель получает лекала не одной, а целой серии моделей. При приобретении картонного комплекта главным является качество посадки изделия, при приобретении алгоритма построения лекала главным являются принципы обеспечения этого качества посадки, которые затем могут быть перенесены на другие модели, разрабатываемые на предприятии.

Обмен и продажа лекал на уровне алгоритмов упрощает саму физическую передачу лекал и повышает заинтересованность приобретающей стороны. Возможна коммерческая разработка алгоритмов построения в виде программ.

Фирма "Вилар", обладая достаточным опытом по работе в системе ЛЕКО и владеющая новым подходом к конструированию, намерена освоить коммерческое производство алгоритмов построения основ и моделей швейных изделий.

# 15. ДОПОЛНИТЕЛЬНЫЕ ЗАМЕЧАНИЯ

Задача конструирования в системе ЛЕКО не сводится к выбору значений коэффициентов в существующих методиках построения лекал. Как мы видим, главная цель разработки лекал именно в системе ЛЕКО - разработка и формализация принципов конструирования путем учета соотношений и закономерностей (фигуры человека, геометрических, эстетических, эргономических и т.д.).

Первая задача при разработке лекала - выбор принципа конструирования для отдельных узлов, согласование этих узлов меду собой. Для этого могут использоваться:

- учет расположения линий в пространстве;
- принципы плоскостного моделирования;
- использование собственного опыта построения;
- использование готовой методики с готовыми коэффициентами;
- "алгоритмизация" существующих лекал;

После выбора принципа конструирования необходимо выбрать метод и форму записи. Как неоднократно упоминалось, одно и то же действие в системе ЛЕКО можно записать различными способами. Выбор способа записи - личное дело каждого конструктора.

Следующий этап - непосредственная запись программы построения лекала на языке описания и построения лекала. На этом этапе можно проводить упрощение записи построения:

- за счет возможностей языка (неявные построения);
- на основе допущений (например, линейной зависимости, малости углов и т.д.)
- за счет сужения диапазона применимости модели (отбрасывание больших размеров).

При записи текста программы следует учитывать, что текст программы будет использоваться не один раз. Следовательно, текст программы должен быть читабельным, модифицируемым, самодокументируемым. Это достигается за счет использования комментариев, профилирования записи программы, выбора наименования идентификаторов, введения определенной дисциплины записи. В программе должно быть предусмотрено не только само построение, но и возможность контроля, т.е. необходимые для контроля величины и выражения должны быть определены как отдельные переменные.

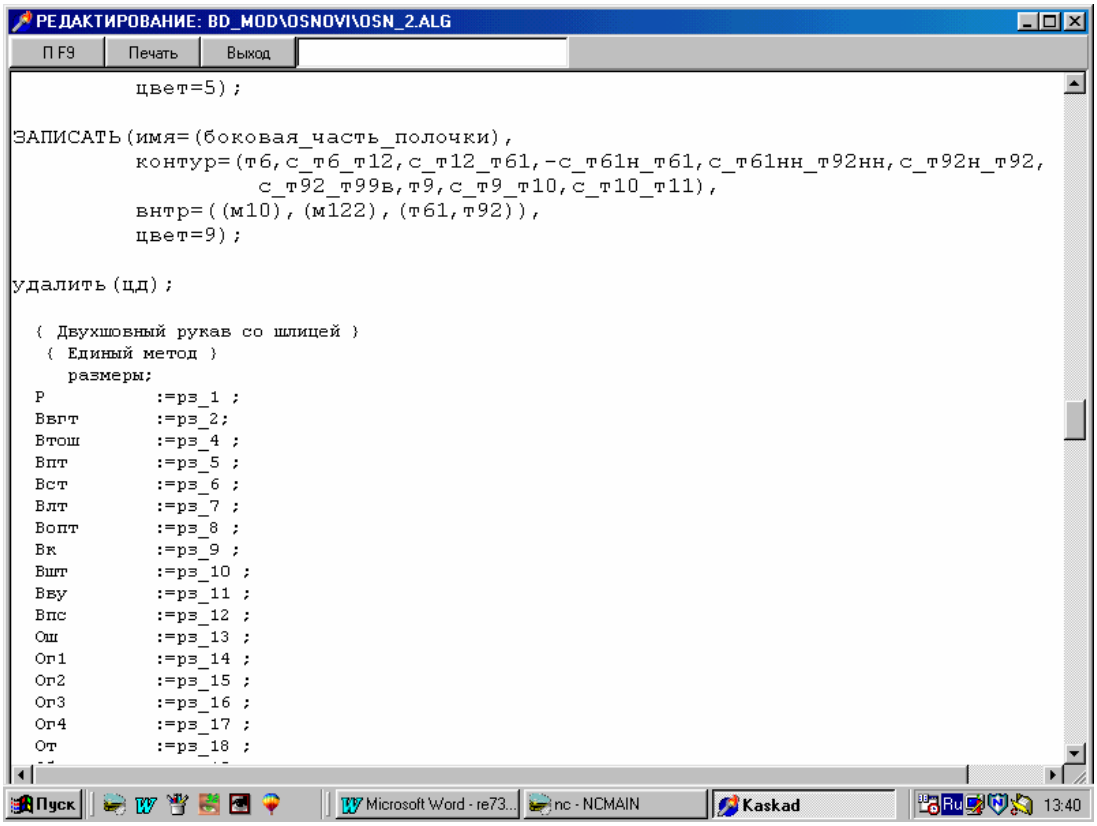
## ПРИЛОЖЕНИЕ

Вопрос: Прошу объяснить более подробно, как произвести вставку детали (например, рукава) в другое лекало (например, основы платья) (ЛЕКО 6.8).

Ответ: При вставке детали из другой модели необходимо согласовать длины сопрягаемых участков (пройма-рукав, горловина-линия втачивания воротника и т.п.).

Рассмотрим пример вставки рукава в основу: в основу BD_MOD\OSNOVI\OSN_2.ALG вставим рукав BD_MOD\OSNOVI\OSN_38.ALG.

Выбираем из библиотеки моделей основу, выходим в режим "Работа с конструкцией", опускаем курсор в конец алгоритма - перед оператором "КОНЕЦ". Открываем файл, который необходимо вставить (с помощью клавиши F3), выбираем C:\LEKO68\BD_MOD\OSNOVI\OSN_38.ALG. С помощью клавиш Shift+стрелка выделяем весь алгоритм, записываем его в буфер обмена - Ctrl+Insert, нажимаем кнопку "Выход" в верхнем меню (выходим из алгоритма построения рукава и возвращаемся в построение основы) и вставляем скопированную часть алгоритма - Shift +Insert:

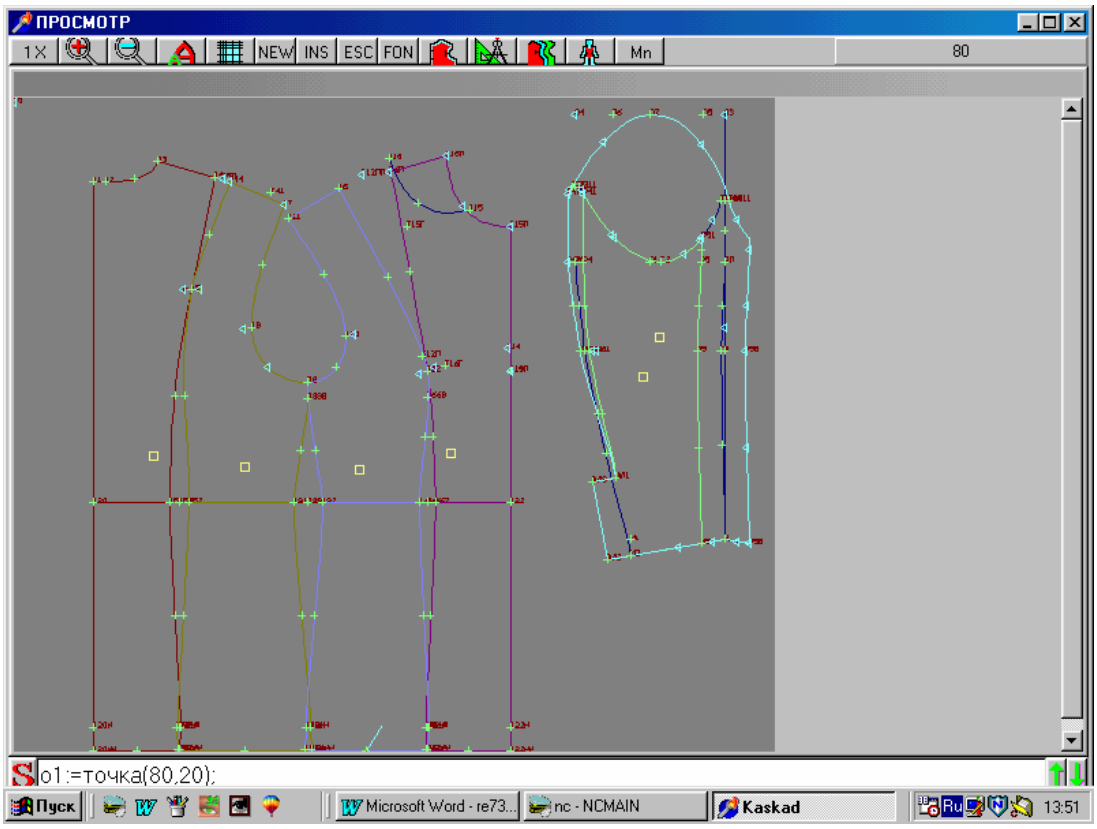



Теперь переходим к согласованию длины проймы и оката рукава. Данный рукав построен под определенную длину проймы, нужно перевести его в относительные величины. Нужно исправить длину проймы, начальную высоту оката, ширину проймы и ее длину по участкам, т.е. все параметры, заданные в сантиметрах.

*длина_проймы:=52.15 ;*
*нач_выс_оката:= 21.5;*
*уменьш_выс_оката:=1+(124-Ог3)*0.0625 ;*
*доб_телосл:=0 ; { от 0.5 до 1 }*
*доб_модели:=0 ; { от -2 до 1.5 }*
*Вок:=нач_выс_оката-уменьш_выс_оката+доб_телосл+доб_модели;*
*норма_пос:=0.05; { от вида ткани }*
*Пос_рук:=норма_пос*длина_проймы;*
*Поп:= 9;*
*Ппн:= 0;*
*Поз:=8 ;*
*шир_рук:=(Оп+Поп)/2 ;*
*дл_пр_1:=13.13 ;*

*дл_пр_2:=13.03;*
*дл_пр_3:=14.73 ;*
*дл_пр_4:=11.90 ;*
*шир_проймы:=14 ;*

Для наглядности чертежа передвинем точку О1 вправо:
о1:=точка(80,20);
Вот чертеж:




Начинаем исправлять параметры оката рукава:

*длина_проймы:=52.15 ; меняем на сумму длин сплайнов проймы:*
*длина_проймы:=с_т7_т8.л+с_т8_т9.л+с_т9_т10.л+с_т10_т11.л;*

Данный рукав построен по методике ЦОТШЛ, начальная высота оката в ней равна вертикальному диаметру проймы, т.е. длине отрезка т9-т11+1.0

*нач_выс_оката:= 21.5;*

меняем на:

*нач_выс_оката:= [т9:т11].л+1.0;*

*дл_пр_1:=13.13 ;*

меняем на соответствующую длину участка проймы:

*дл_пр_1:=c_m7_m8.л ;*

*дл_пр_2:=13.03;*

меняем на:

*дл_пр_2:=c_m8_m9.л;*

*дл_пр_3:=14.73 ;*

меняем на:

*дл_пр_3:=c_m10_m11.л ;*

*дл_пр_4:=11.90 ;*

меняем на:

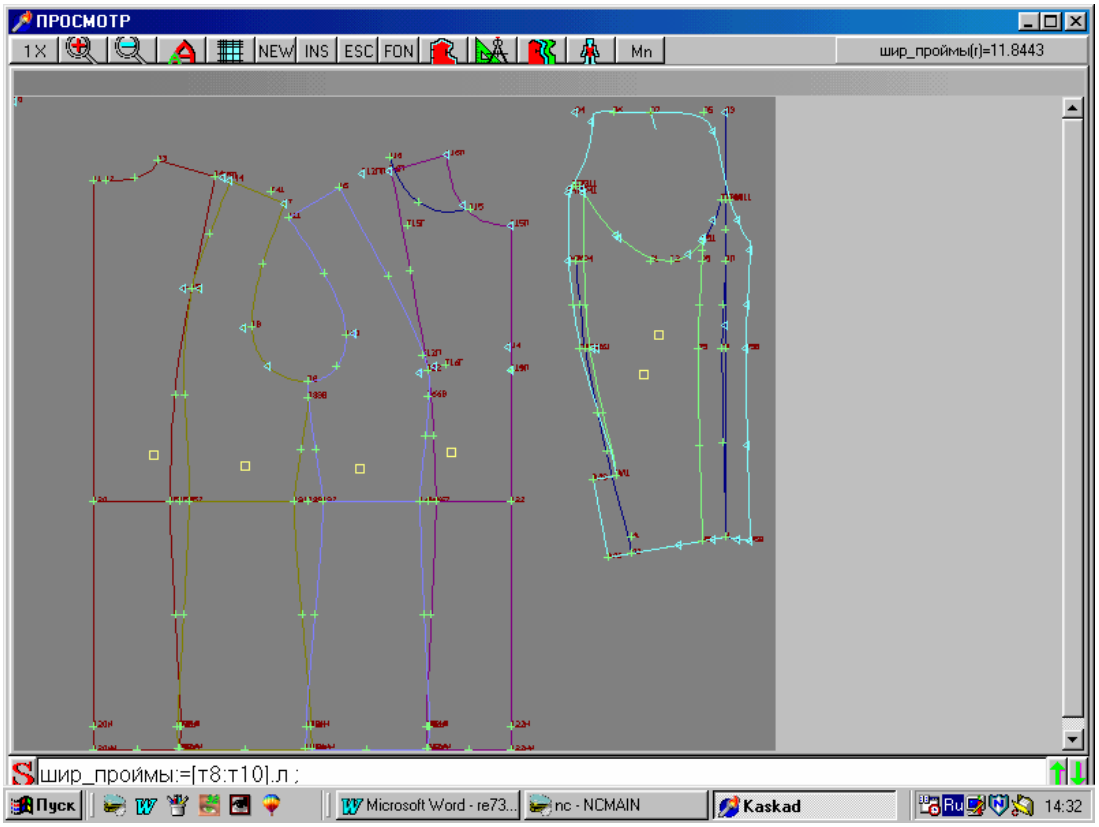
*дл_пр_4:=c_m9_m10.л ;*

*шир_проймы:=14 ;*

меняем на:

*шир_проймы:=[m8:m10].л ;*

После преобразований получаем:




В верхней части оката СПЛАЙН_Д необходимо заменить на СПЛАЙН_К:

*ок_в_з:=сплайн_д(рз1,о2,[рз1 :о6 ].ф1,0,дл_верх_задн_ч_оката);*

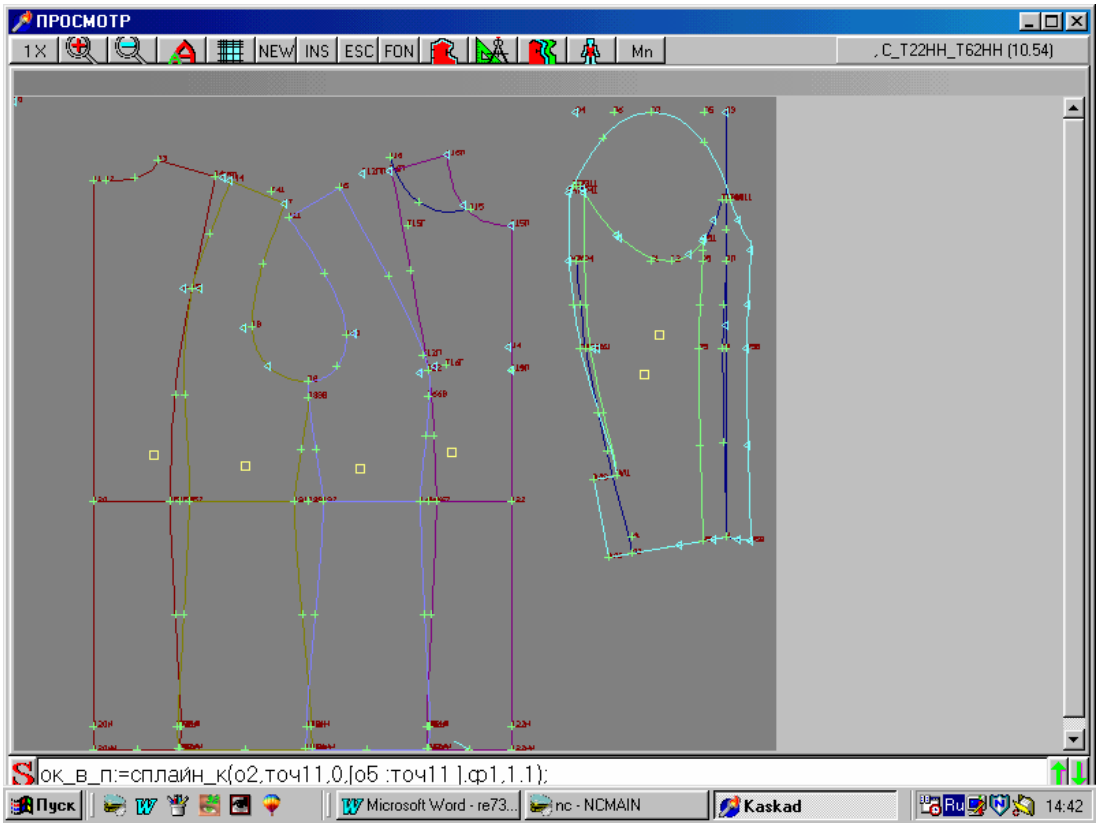
заменяем на

*ок_в_з:=сплайн_к(рз1,о2,[рз1 :о6 ].ф1,0,1.1);*

*ок_в_п:=сплайн_д(о2,моч11,0,[о5 :моч11 ].ф1,дл_верх_пер_ч_оката);*

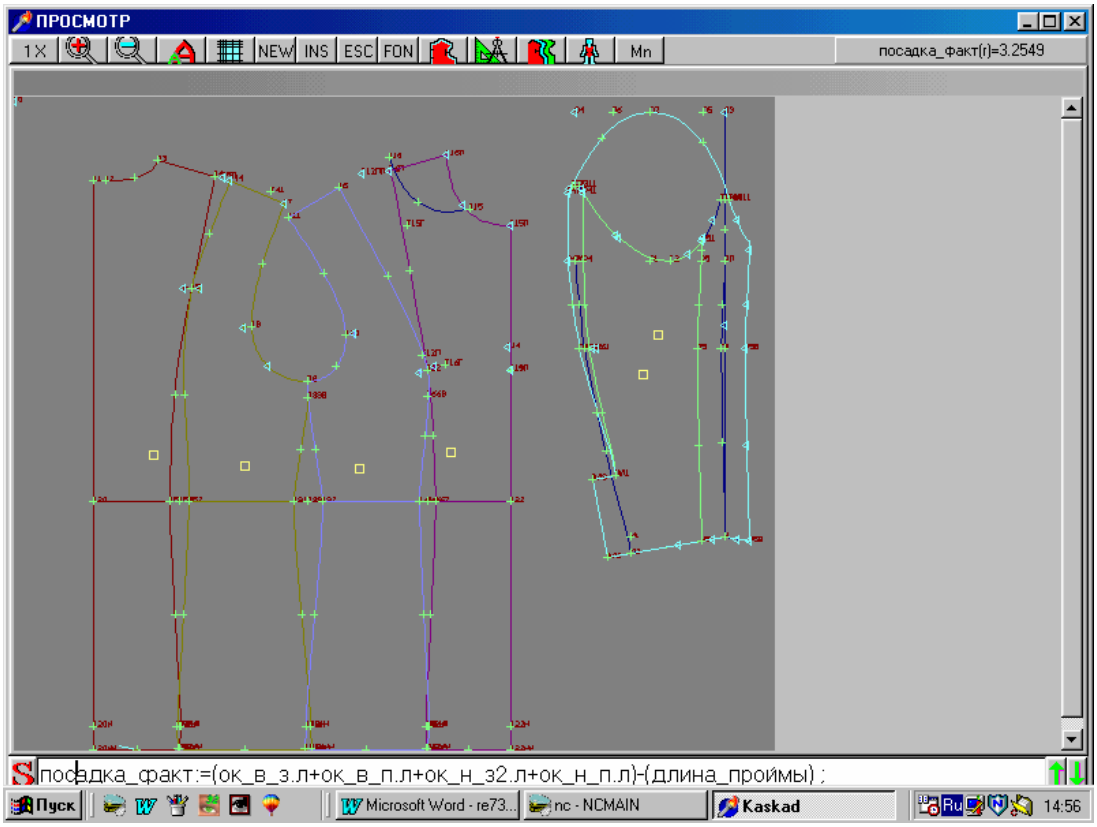
заменяем на

*ок_в_п:=сплайн_к(о2,моч11,0,[о5 :моч11 ].ф1,1.1);*




Теперь можно проверить фактическую посадку рукава - для этого в конец алгоритма добавляем строку

*посадка_факт:=(ок_в_з.л+ок_в_п.л+ок_н_з2.л+ок_н_п.л)-(длина_проймы) ;*




Фактическая посадка равна 3.25 см. Если эта величина не устраивает, изменяют норму посадки или используют оператор "СЖАТЬ".

В заключение хочется отметить, что более поздние алгоритмы (готовые модели) настраиваются проще, т.к. все параметры в них записаны через относительные величины (размерные признаки и коэффициенты) - часто необходимо просто скопировать нужную часть алгоритма, и система автоматически пересчитает алгоритм.

# РАБОТА С БАЗОЙ ДАННЫХ РАЗМЕРНЫХ ПРИЗНАКОВ

***Принципиально новые возможности системы помогут Вам воплотить свои замыслы***

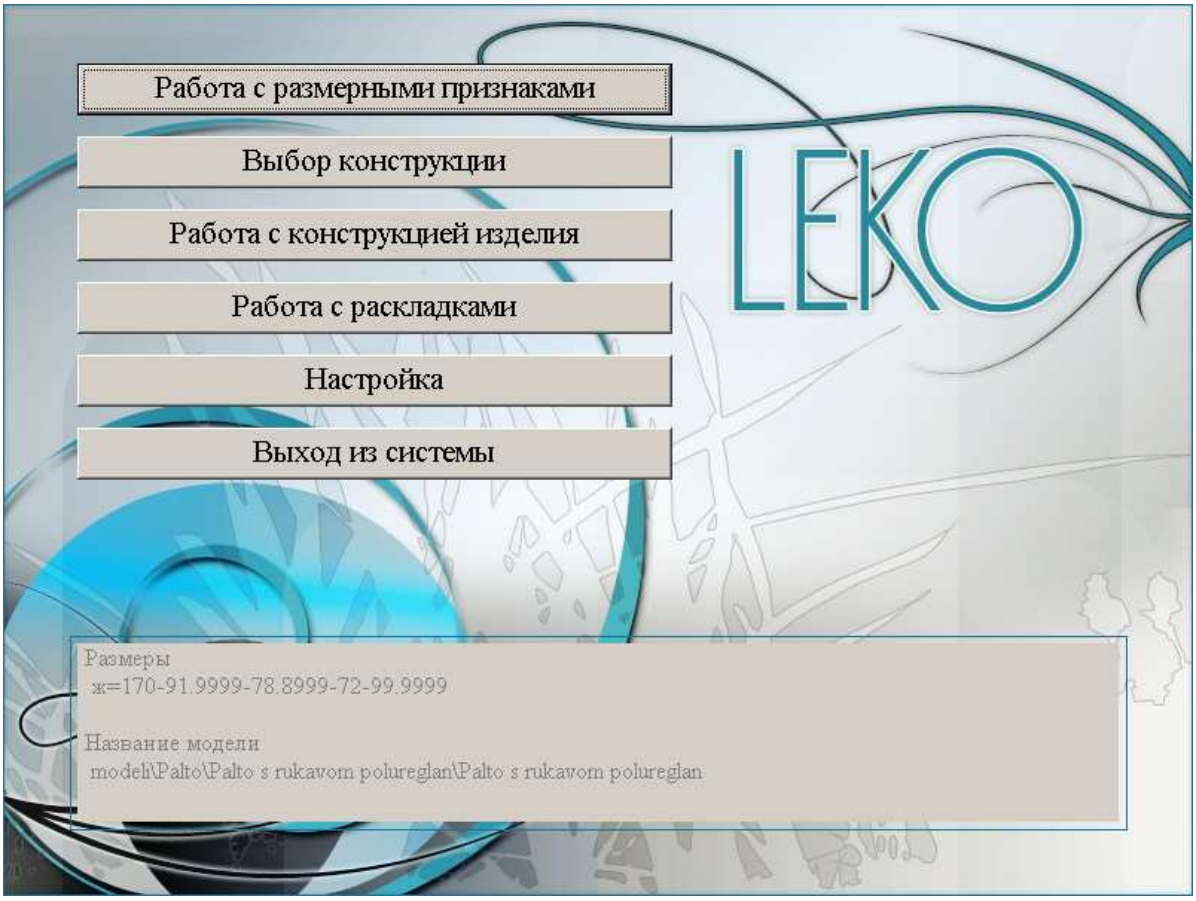
# ВВЕДЕНИЕ

Настоящая часть описания системы ЛЕКО посвящена работе с базой данных значений размерных признаков для типовых фигур. База данных размерных признаков типовых фигур является основой для разработки одежды в промышленности. И это не из-за директивных указаний типа "несоблюдение стандарта преследуется по закону". База данных размерных признаков типовых фигур объективно является для конструктора единственным источником информации о типовых фигурах людей, для которых создается одежда. Манекены, разработанные для нескольких фигур базовых размеро-ростов, не могут полностью отразить все многообразие типовых фигур, предусмотренных ГОСТами и ОСТами.

Использование базы данных значений размерных признаков позволяет конструктору более полно использовать размерные признаки во время построения, не запоминая при этом конкретных значений размерных признаков.

Наличие в системе ЛЕКО базы данных значений размерных признаков и возможность использования этих значений для построения позволяет достаточно просто вести разработку лекал на типовые фигуры, производить автоматическое техническое размножение (градацию) лекал по размеро-ростам, дает возможность аналитического контроля правильности построения лекал.

Вызов режима работы с размерными признаками осуществляется из Главного меню системы ЛЕКО:




При работе с конструкцией изделия используются таблицы значений размерных признаков, подготовленные в режиме "Работа с размерными признаками".

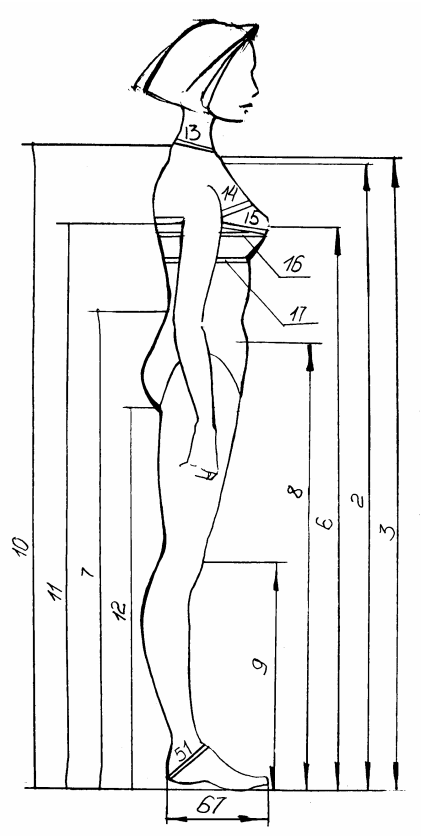
# 2. БАЗЫ ДАННЫХ РАЗМЕРНЫХ ПРИЗНАКОВ, ПОСТАВЛЯЕМЫЕ С СИСТЕМОЙ

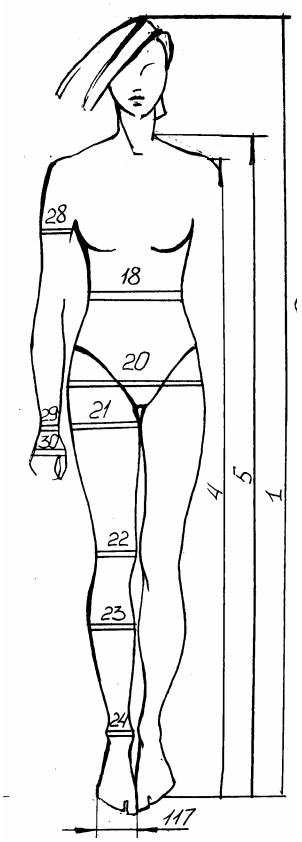
Предлагается база данных для проектирования швейных изделий. В базу данных включена информация по размерной типологии взрослого населения по материалам подготовленным для выпуска последнего ГОСТа.

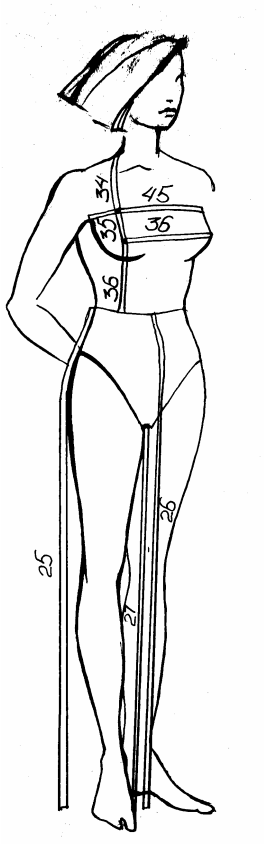
База данных содержит все необходимые величины размерных признаков для конструирования одежды. База данных размерных признаков и методов их измерения соответствует действующим государственным и отраслевым стандартам.

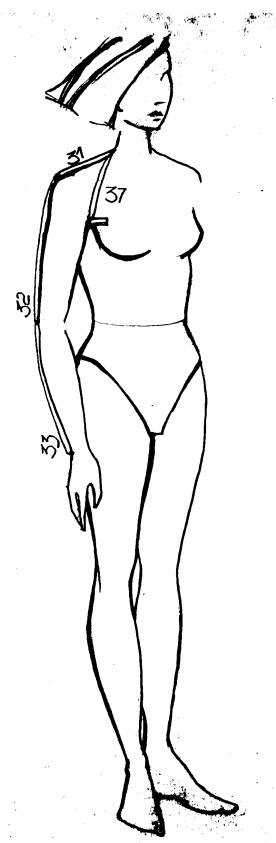
# 3. ПОРЯДОК СНЯТИЯ РАЗМЕРНЫХ ПРИЗНАКОВ

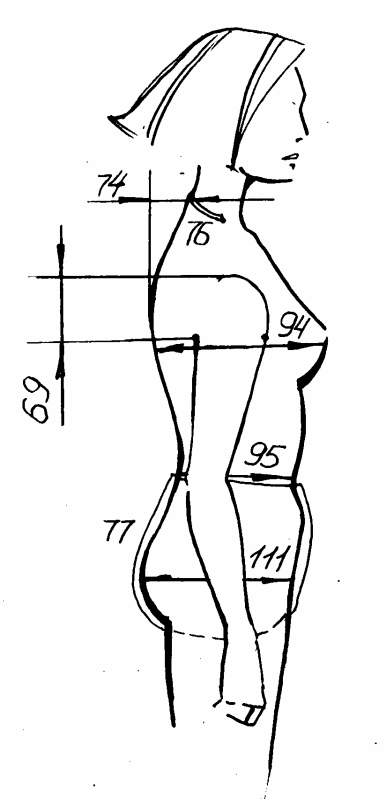
Ниже приведены схемы снятия размерных признаков.

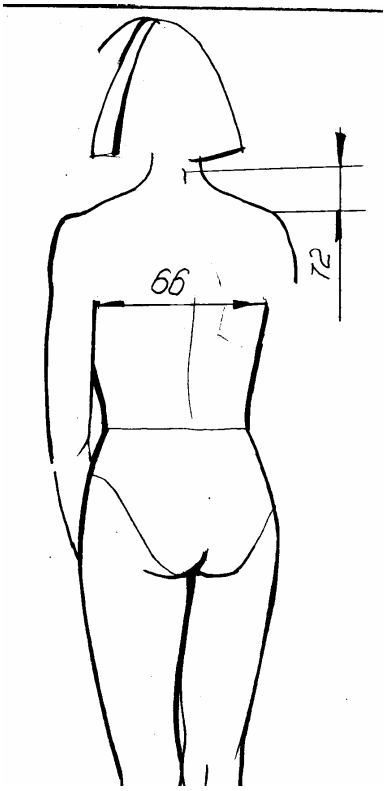

13
14
15
16
17
10
11
7
12
4
8
6
2
3
9
51
67

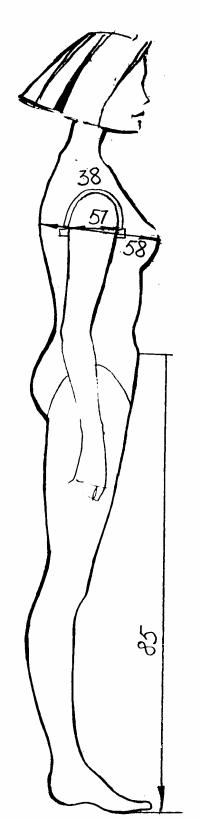

28
18
20
21
29
30
4
5
1
22
23
24
117

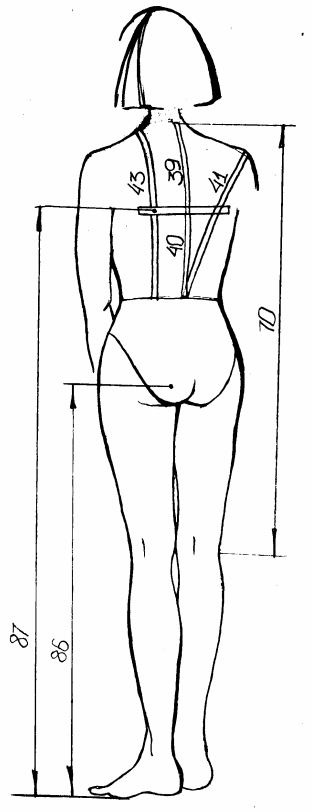

34
45
35
36
36
25
26
27

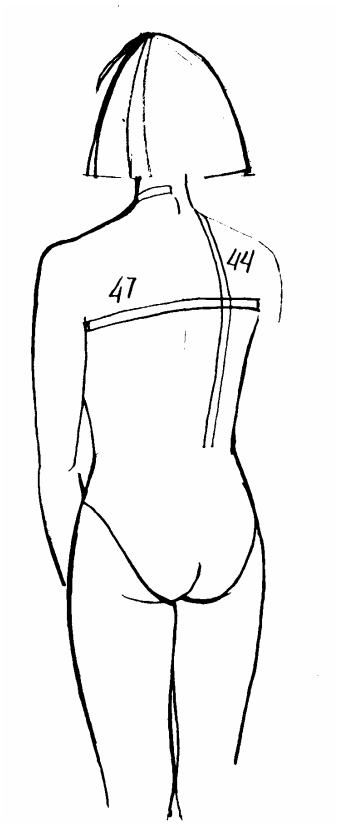

31
37
32
33

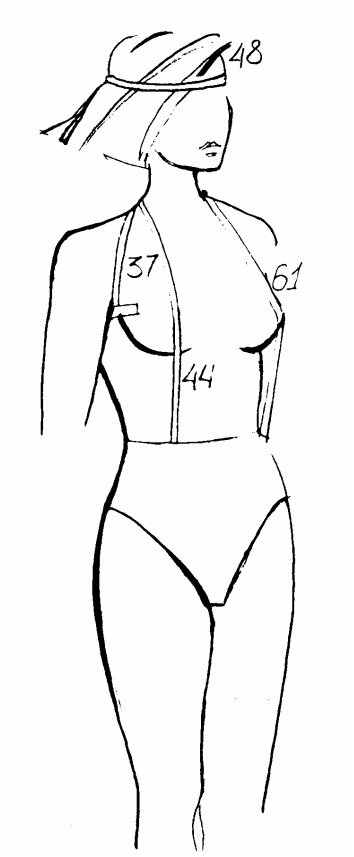

74
76
94
69
95
77
111

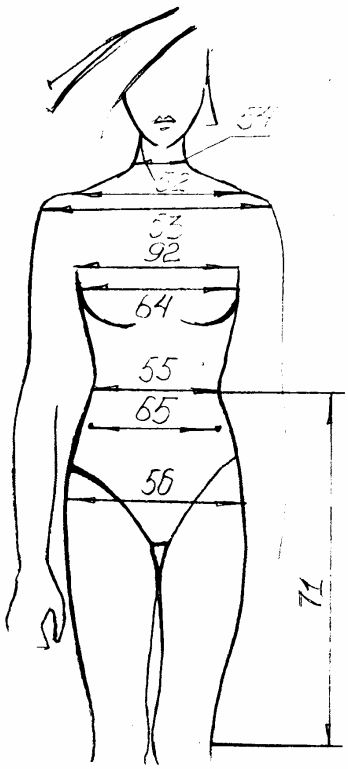

66
72

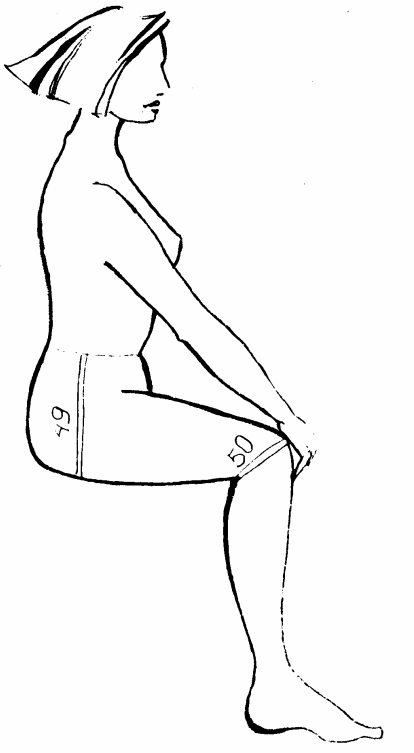

38
51
58
85


43
39
44
40
70
87
86

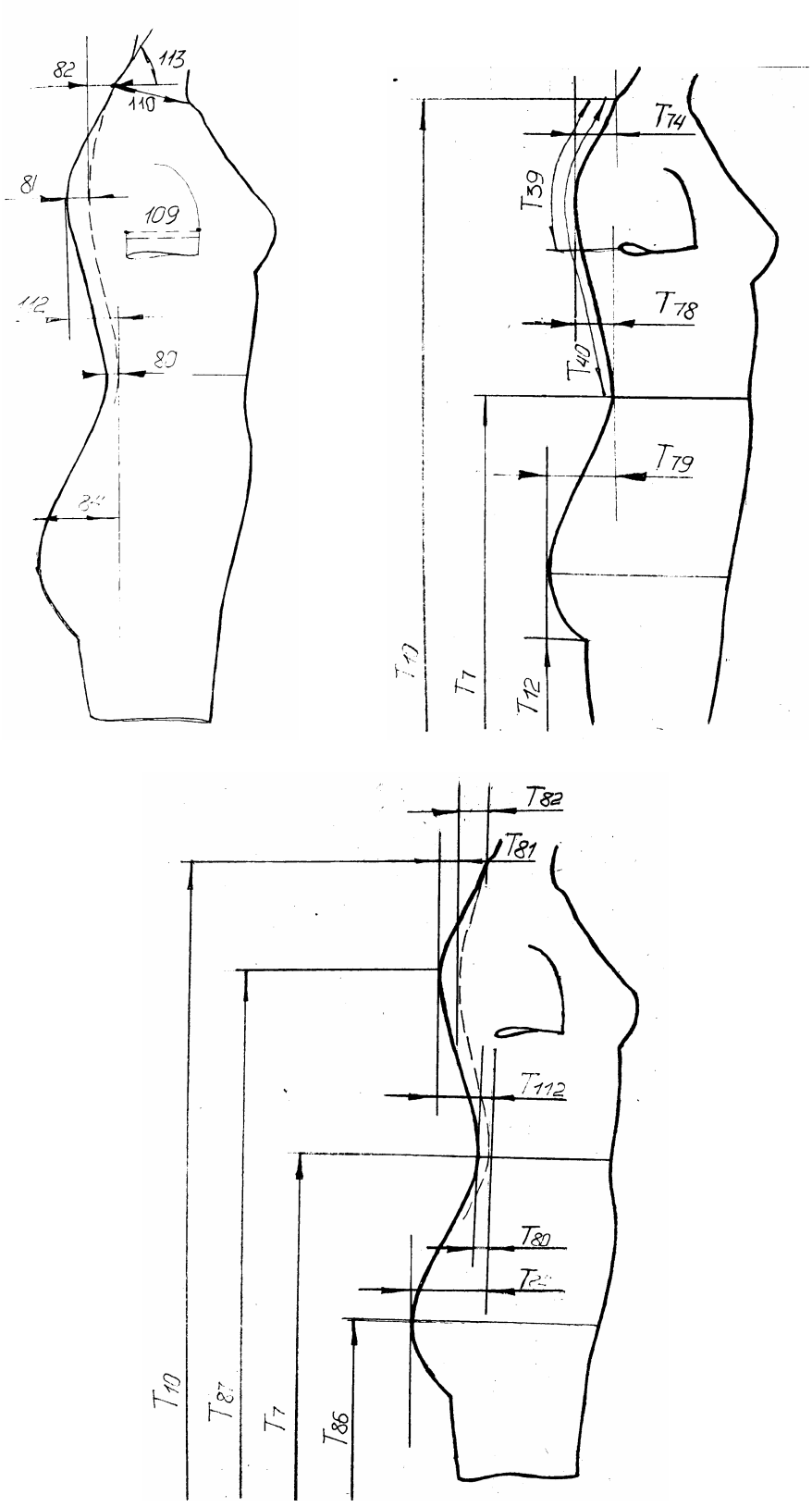

47
44


48
37
61
44


92
64
55
65
56
71


49
50

100




1. Рост - измеряют по вертикали расстояние от пола до верхушечной точки.

3. Высота ключичной точки - измеряют по вертикали расстояние от пола до ключичной точки.

4. Высота точки основания шеи - измеряют по вертикали расстояние от пола до точки основания шеи.

5. Высота плечевой точки - измеряют по вертикали расстояние от пола до плечевой точки.

6. Высота сосковой точки - измеряют по вертикали расстояние от пола до выступающей точки грудной железы.

7. Высота линии талии - измеряют по вертикали расстояние от пола до линии талии.

9. Высота коленной точки - измеряют по вертикали расстояние от пола до коленной точки.

10. Высота шейной точки - измеряют по вертикали расстояние от пола до шейной точки.

11. Высота заднего угла подмышечной впадины - измеряют по вертикали расстояние от пола до заднего угла подмышечной впадины.

12. Высота подъягодичной складки - измеряют по вертикали расстояние от пола до середины подъягодичной складки.

13. Обхват шеи - лента проходит по основанию шеи, плотно прилегая, на уровне яремной точки.

14. Обхват груди 1 - лента по спине проходит горизонтально, касаясь задних углов подмышечных впадин, затем по подмышечным впадинам; спереди проходит над основанием грудных желез.

15. Обхват груди 2 - сзади и по подмышечным впадинам измерение снимается как измерение 14. Спереди лента проходит через выступающие точки грудных желез.

16. Обхват груди 3 - лента проходит горизонтально вокруг туловища через выступающие точки грудных желез.

17. Обхват груди 4 - лента проходит горизонтально вокруг туловища под основанием грудных желез и замыкается на правой стороне груди.

18. Обхват талии - лента должна проходить горизонтально на уровне линии талии.

19. Обхват бедер с учетом выступа живота - ленту накладывают на наиболее выступающие ягодичные точки; она должна проходить горизонтально вокруг туловища, спереди по гибкой пластине, приложенной вертикально к животу для учета выступа живота.

20. Обхват бедер - ленту накладывают на наиболее выступающие ягодичные точки; она должна проходить горизонтально вокруг туловища.

21. Обхват бедра - лента проходит горизонтально вокруг бедра, касаясь верхним краем подъягодичной складки, замыкается на наружной поверхности бедра.

22. Обхват колена - лента проходит горизонтально вокруг ноги на уровне коленной точки, замыкается на наружной поверхности ноги.

23. Обхват икры - измеряют максимальный обхват ноги в области икроножной мышцы. Лента должна проходить горизонтально вокруг ноги и замыкаться на наружной поверхности голени.

24. Обхват щиколотки - лента должна проходить горизонтально вокруг ноги непосредственно над внутренней лодыжкой замыкаться на наружной поверхности голени.

25. Расстояние от линии талии до пола сбоку - измеряют от точки высоты линии талии по боковой поверхности бедра, через наиболее выступающую область бедра и далее вертикально до пола.

26. Расстояние от линии талии до пола спереди - измеряют от линии талии через наиболее выступающую точку живота и далее вертикально до пола.

27. Длина ноги до внутренней поверхности - измеряют по внутренней поверхности ноги от промежности до пола при слегка раздвинутых ногах.

28. Обхват плеча - измеряют перпендикулярно оси плеча. Верхний край ленты должен касаться заднего угла подмышечной впадины. Лента должна замыкаться на наружной поверхности руки.

29. Обхват запястья - измеряют перпендикулярно оси предплечья по лучезапястному суставу через головку локтевой кости. Лента должна замыкаться на наружной поверхности руки.

30. Обхват кисти - измеряют перпендикулярно оси кисти через пястнофаланговый сустав первого пальца. Первый палец должен быть противопоставлен второму и отведен от него на 30-35 градусов. Лента должна замыкаться на наружной поверхности руки.

31. Ширина плеча - измеряют от точки основания шеи до плечевой точки.

34. Расстояние от шейной точки до линии обхвата груди первого спереди (высота проймы спереди) - измеряют от шейной точки через точку основания шеи до отметки на линии обхвата груди первого спереди (размерный признак 14).

35. Высота груди - измеряют от седьмого шейного позвонка через точку основания шеи до выступающей точки грудной железы.

36. Длина талии - измеряют от седьмого шейного позвонка через точку основания шеи и спереди выступающую точку грудной железы и далее параллельно вертикальному положению корпуса до линии талии.

37. Расстояние от шейной точки до уровня заднего угла подмышечной впадины спереди (высота проймы косая) - измеряют от шейной точки через точку основания шеи, далее по направлению к переднему углу подмышечной впадины. Конечная точка измерения находится под передним углом подмышечной впадины на уровне заднего угла. Уровень заднего угла подмышечной впадины должен фиксироваться пластиной шириной до 2 см, которая верхним краем касается заднего угла и идет горизонтально по подмышечной впадине.

38. Дуга через высшую точку плечевого сустава - измеряют в вертикальной плоскости от уровня заднего угла подмышечной впадины через наивысшую точку плечевого сустава до уровня подмышечной впадины спереди.

39. Высота проймы - измеряют расстояние от седьмого шейного позвонка до линии обхватов сзади груди 1 с учетом выступа лопаток (через приложенную к лопаткам линейку).

40. Длина спины - измеряют от линии талии вдоль позвоночника через линейку шириной 2 талии см, наложенную на выступающие точки лопаток до седьмого шейного позвонка.

41. Высота плеча косая - измеряют по кратчайшему расстоянию от пересечения линии талии с позвоночником до плечевой точки.

43. Расстояние от линии талии сзади до точки основания шеи - лента должна проходить сзади от линии талии до точки основания шеи параллельно позвоночнику.

45. Ширина груди - измеряют над основанием грудных желез между передними углами подмышечных впадин.

46. Расстояние между сосковыми точками - измеряют между выступающими сосковыми точками грудных желез.

47. Ширина спины - измеряют горизонтально по лопаткам между задними углами подмышечных впадин на линии обхвата груди 1.

48. Обхват головы - измеряют через наиболее выступающую точку затылочного бугра и центры лобных бугров. Лента должна замыкаться спереди.

49. Расстояние от линии талии до плоскости сидения - измеряют по боку от линии талии до горизонтальной плоскости сидения. Измеряемый должен сидеть на стуле с плоским твердым сидением.

50. Обхват колена в согнутом положении ноги - лента проходит по подколенной ямке через коленную точку и замыкается спереди. Нога должна быть согнута под прямым углом.

51. Обхват подъема стопы - измеряют через заднюю наиболее выступающую вниз точку пятки и наивысшую точку подъема стопы. Лента должна замыкаться спереди.

53. Плечевой диаметр - измеряют спереди расстояние между плечевыми точками.

54. Поперечный диаметр шеи - измеряют между точками основания шеи.

57. Передне-задний диаметр руки - измеряют горизонтально на уровне заднего угла подмышечной впадины.

58. Передне-задний диаметр обхвата груди второго - одну линейку накладывают на выступающие точки грудных желез, другую - на обе лопатки на уровне обхватов груди первого и второго.

61. Расстояние от точки основания шеи до линии талии спереди измеряют расстояние от точки основания шеи через выступающую точку грудной железы, далее параллельно средне-сагиттальной линии до линии талии.

62. Длина руки до локтя - измеряют расстояние от плечевой точки до линии обхвата запястья.

68. Длина руки до запястья - измеряют от плечевой точки до линии обхвата запястья.

69. Вертикальный диаметр руки - определяют вычитанием величины размерного признака 11 из величины размерного признака 5.

70. Расстояние от шейной точки до колена - определяют вычитанием величины размерного признака 9 из величины размерного признака 10.

71. Расстояние от линии талии до колена - определяют вычитанием величины размерного признака 9 из величины размерного признака 26.

72. Высота плеча - определяют вычитанием величины размерного признака 5 из величины размерного признака 10.

73. Высота головы - определяют вычитанием величины размерного признака 10 из величины размерного признака 1.

74. Положение корпуса - Измеряют по горизонтали расстояние от шейной точки до вертикальной плоскости. Плоскость должна касаться наиболее выступающих назад точек обеих лопаток.

75. Дуга плечевого пояса сзади - измеряют по спине между плечевыми точками.

76. Расстояние от шейной точки до точки основания шеи сбоку по линии измерения обхвата шеи - определяют вычитанием величины размерного признака 61 из величины размерного признака 36.

77. Дуга через паховую область - измеряют в вертикальной плоскости от линии талии сзади. Лента должна проходить через тонкую пластинку шириной до 2 см, положенную на выступающие точки ягодицы.

78. Глубина талии 1 - измеряют по горизонтали расстояние от вертикальной плоскости касательной к выступающим точкам лопаток до линии талии.

79. Глубина талии 2 - измеряют по горизонтали расстояние от вертикальной плоскости касательной к наиболее выступающим ягодичным точкам до линии талии.

111. Передне-задний диаметр обхвата талии - измеряют в горизонтальной плоскости. Одну линейку накладывают на переднюю стенку туловища на уровне обхвата талии, другую - на продольные мышцы спины.

# ПРИМЕРЫ

***Принципиально новые возможности системы помогут Вам воплотить свои замыслы***

В настоящей части описания системы ЛЕКО приведены конкретные примеры методик построения лекал и предлагаются некоторые технологические приемы написания методик построения (программ). Схожесть разработки методики построения лекала и программирования делает целесообразным приведение примеров создания методик построения лекал по аналогии с изложенными в учебниках по изучению языков программирования.

Методику построения любой конструкции можно назвать алгоритмом. Это своего рода "рецепт" достижения требуемой цели. Сам термин "алгоритм" имеет древнее происхождение, являясь латинизированной транскрипцией имени великого среднеазиатского ученого IX века Мухаммеда аль-Хорезми. Правила записи алгоритма для выполнения его вычислительной машиной оказываются очень жесткими - автомат не может ничего додумать за человека. Совокупность средств и правил представления алгоритма в виде, пригодном для выполнения его вычислительной машиной, называется языком программирования, а каждый алгоритм, записанный на определенном языке программирования, называется программой.

Программа записывается в виде последовательности символов, к числу которых относятся латинские и русские буквы, арабские цифры, знаки препинания и знаки операций.

Об основных элементах языка построения лекал в нашей системе мы говорили выше. Были приведены примеры реализации некоторых принципов "ручного" построения, продемонстрировано использование новых возможностей, предоставляемых системой и реализуемых при помощи языка построения лекал. Теперь же хотелось бы показать, как составляется сама программа и как с ней работать.

Начнем с простого примера - построения точки.

# 1. НАПИСАНИЕ ПРОГРАММЫ ПОСТРОЕНИЯ ТОЧКИ

Начинать написание любой программы мы рекомендуем с комментариев. Сопровождение построений пояснениями делает программу построения лекал наглядной, легко читаемой, да и помогает Вам находить ошибки или отслеживать отдельные этапы конструирования лекал.

Итак, начнем нашу программу с комментариев:

*{----------Программа построения точки----------}*

Для установки комментария необходимо поставить открывающуюся и закрывающуюся фигурные скобки. В зависимости от используемых на Вашей машине программ русификации, для установки фигурных скобок Вам, возможно, потребуется перейти в латинский режим работы клавиатуры. Если Вы не стирали и не изменяли значения словарей, поставляемых с системой, то можете выбрать заготовку для комментария из словаря. Для переключения словарей нажмите комбинацию клавиш Alt+E или Q или W (сначала нажмите клавишу Alt и затем, не отпуская ее, нажмите клавишу).

Выберите мышкой нужный пункт строки-заготовки для комментария и нажмите левую кнопку. Указанная строка перенесется из словаря в редактируемый текст, и далее Вы можете исправлять или дополнять ее.

Далее следует сам оператор построения (установки) точки:

```
{---------Программа построения точки----------}
 точка_1:=точка(1,1);
```

Напомним, что сначала указывается имя точки или, как это называется в языках программирования, идентификатор точки. В нашем случае точка называется ТОЧКА_1, но можно было бы назвать ее ТОЧ1, А, Н, ТОЧКА_ПЛЕЧА и т.д. На имена не накладывается никаких ограничений, кроме тех, которые имеют идентификаторы, т.е. имя должно начинаться с буквы и содержать только буквы, цифры и знак подчеркивания "_".

Заготовку для оператора построения точки ":=точка( , );" Вы можете выбрать из словаря (Alt+Q) и затем дополнить ее названием точки и координатами.

Рекомендуем Вам в дальнейшем выбрать какой-то один принцип наименования всех элементов построения (точек, отрезков, дуг и т.д.): или взять за основу какую-либо методику (например, ЕМКО СЭВ), или самостоятельно дать названия основным конструктивным точкам лекала и придумать правила наименования промежуточных точек.

Завершает программу ключевое слово "КОНЕЦ":

```
конец
```

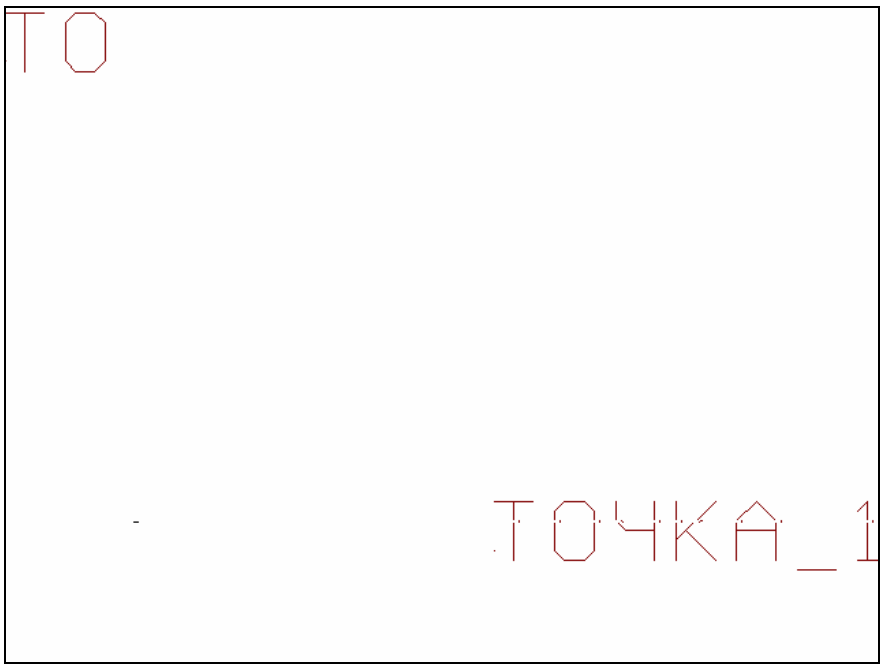
Все ключевые слова в программе Вы можете не набирать вручную, а выбирать заготовки из словарей. В дальнейшем Вы можете исправить словари в соответствии с теми методиками и принципами построения лекал, которые Вы будете использовать.

Теперь мы можем Вас поздравить с первой программой, написанной на языке системы ЛЕКО.

```
{---------Программа построения точки----------}
точка_1:=точка(1,1);
конец
```

Теперь необходимо проверить написанную программу, и в случае, если система не выдаст сообщения об ошибке, можно посмотреть на экране результат построения. Для этого Вам необходимо нажать клавиши F9 (построение) или кнопку "Перепостроение" панели управления.

...Система проверила программу и построила лекала (в нашем случае - только одну точку). Вы можете посмотреть результат работы Вашей программы.




Усложним условие: требуется построить несколько точек с заданными координатами.

## 1.1. РАСШИРЕНИЕ ПРОГРАММЫ

Напишем программу построения трех точек, наложив условия на их взаимное расположение: вторая точка имеет координаты (20,30), а третья должна находиться выше второй на 10 сантиметров.

```
{---------Программа построения точки----------}
точка_1:=точка( 0, 0);
точка_2:=точка( 20, 30);
точка_3:=точка( точка_2.x, точка_2.y-10);
конец
```

Для набора этой программы Вы можете использовать заготовки операторов из словарей или использовать режим копирования фрагментов текста. Опишем один из возможных способов копирования по шагам.

У Вас есть текст первой программы:

```
1 {---------Программа построения точки---------}
2
3 точка_1:=точка(1,1);
4
5 конец
6
```

Подведите курсор в начало третьей строки. Нажмите клавишу Shift и, не отпуская ее, нажмите клавишу "Стрелка вниз". Третья строка при этом поменяет цвет, станет "выделенной". Далее нажмите комбинацию клавиш Ctrl+Ins - при этом выделенный текст попадает в "карман" (clipboard). Теперь у Вас в "кармане" находится фрагмент текста "точка_1:=точка(1,1);".

Опустите курсор на одну строчку вниз. "Выделенный" текст стал обычным. Вставьте дополнительную пустую строку (нажмите клавишу "Enter").

```
1 {---------Программа построения точки-----------}
2
3 точка_1:=точка(1,1);
4
5
6 конец
7
```

Вставьте фрагмент текста из кармана в текущее положение курсора - нажмите Shift+Ins.

```
1 {---------Программа построения точки------------}
2
3 точка_1:=точка(1,1);
4
5 точка_1:=точка(1,1);
6
7 конец
8
```

Подведите курсор и исправьте название точки и координаты.

```
1 {----------Программа построения точки-----------}
2
3 точка_1:=точка(1,1);
4
5 точка_2:=точка(20,30);
6
7 конец
8
```

Установите курсор в начало шестой строки и вставьте дополнительную строчку (нажмите клавишу "Enter").

В Вашем "кармане" по-прежнему находится фрагмент текста: "точка_1:=точка(1,1);".

Нажмите комбинацию клавиш Shift+Ins (вставка фрагмента текста из кармана). При этом Вы второй раз скопируете текст из кармана в редактируемый текст.

```
1 {---------Программа построения точки------------}
2
```

```
3 точка_1:=точка(1,1);
4
5 точка_2:=точка(20,30);
6
7 точка_1:=точка(1,1);
8
9 конец
10
```

Подведите курсор к началу пятой строки. Нажмите клавишу "Shift" и, не отпуская ее, нажмите клавишу "Стрелка вправо". Слово "точка_2" начнет менять цвет, будет становиться выделенным. Не отпуская клавиши "Shif"t, используя клавиши "Стрелка вправо", "Стрелка влево", выделите слово "точка_2" и отпустите клавишу "Shift". Скопируйте выделенный текст в "карман" комбинацией клавиш Ctrl+Ins.

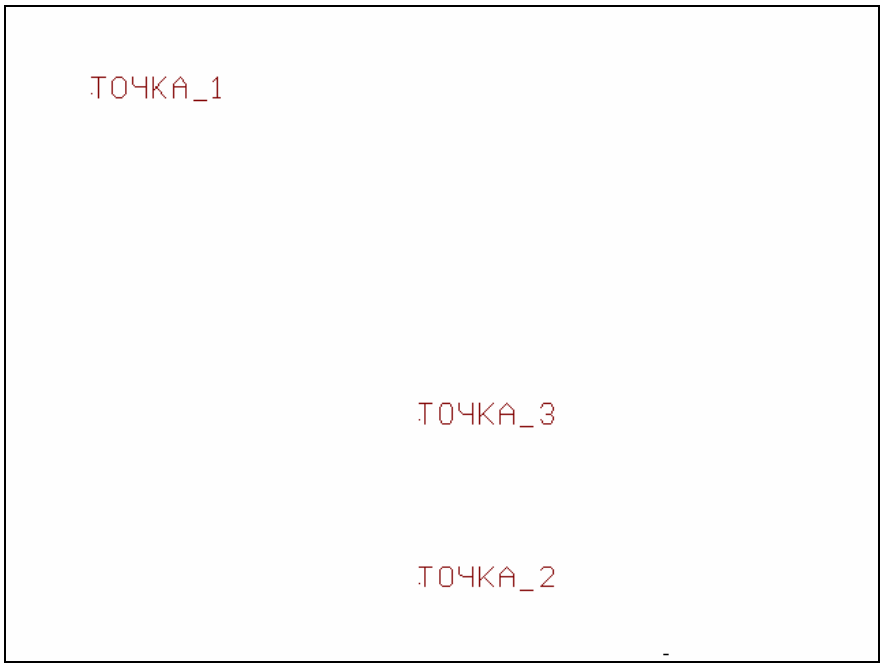
Перейдем к редактированию седьмой строки.

Исправьте название "точка_1" на "точка_3". Подведите курсор к первой координате (1) и сотрите ее (клавиша "Delete"). Вставьте текст из кармана (комбинация клавиш Shift+Ins). Вызовите словарь "W", нажав комбинацию клавиш Alt+W, и выберите заготовку для указания горизонтальной координаты точки ". x". Аналогично замените вторую координату точки, вычтя дополнительно 10 сантиметров для подъема точки вверх.

Вы получите текст второй программы:

```
{----------Программа построения точки----------}
 точка_1:=точка( 0, 0);
 точка_2:=точка( 20, 30);
 точка_3:=точка( точка_2.x, точка_2.y-10);
 конец
```

Проверим программу (F9) и посмотрим результат на экране.



Теперь Вы уже можете перейти к построению простейших геометрических фигур, например, треугольника.

```
{--------Программа построения треугольника----------}
 { начальные условия }
 смещ_по_у_точки_3:=10; { смещение по оси У третьей точки
 относительно второй }
{-------------------------------------------------}
 { расстановка точек }
 точка_1:=точка( 0, 0);
 точка_2:=точка( 20, 30);
 точка_3:=точка( точка_2.x, точка_2.y-смещ_по_у_точки_3);
 { построение треугольника }
 о1:=отрезок(точка_1,точка_2);
 о2:=отрезок(точка_2,точка_3);
 о3:=отрезок(точка_3,точка_1);
конец
```

Обратите внимание на то, что мы не стали использовать абсолютные величины смещения при записи построения точки 3, а вынесли значение смещения в начало программы, определив его через переменную - параметр "смещ_по_у_точки_3". Вам может показаться громоздким название этой переменной, но не бойтесь и не ленитесь написать лишнюю букву или символ в программе. Главное, чтобы Вам было понятно назначение вводимых переменных во время текущей работы и когда Вы обратитесь к этому построению через месяц, полгода или год.

Преимущество использования переменных-параметров Вы почувствуете, когда у Вас будет много программ и они будут занимать значительные объемы или когда Вы возобновите работу со старыми программами, забыв о назначении тех или иных переменных. Введение переменных-параметров вместо абсолютных значений упрощает процесс отработки и внесения изменений в построение лекал. Вам не потребуется вторгаться в само построение, изучая весь полный текст программы, а необходимо будет изменить лишь одно-два значения в начале программы.

Попробуйте теперь описать сплайн, проходящий через вершины треугольника. Сделаем это для каждой пары точек. Дополнительно наложим условия построения сплайнов: касательные в точке начала и конца каждого сплайна должны быть перпендикулярны отрезку, соединяющему эти точки, т.е. соответствующей стороне треугольника.

```
{-----------Программа построения сплайнов---------}
 { расстановка точек }
 точ1:=точка( 10, 0);
 точ2:=точка( 0, 20);
 точ3:=точка( 25, 20);
 { построение треугольника }
 о1:=отрезок(точ1,точ2);
 о2:=отрезок(точ2,точ3);
 о3:=отрезок(точ3,точ1);
 { построение сплайнов }
```

```
спл12:=сплайн_к(точ1,точ2,[точ1:точ2].ф1-
90,[точ1:точ2].ф1+90,1.0);
спл13:=сплайн_к(точ1,точ3,[точ1:точ3].ф1+90,
[точ1:точ3].ф1-90,1.0);
спл23:=сплайн_к(точ2,точ3,[точ2:точ3].ф1-
90,[точ2:точ3].ф1+90,1.0);
конец
```

## 1.2. ПОИСК ПЕРЕСЕЧЕНИЯ ПРЯМЫХ

Одна из наиболее часто используемых операций - поиск пересечения прямых. Если прямые параллельны осям координат, то точку пересечения можно определить через задание координат. Например, при разметке базисной сетки:

```
т1:=точка( 0, 0);
т2:=точка( Об/2, 0);
н1:=точка( Ди, 0);
н2:=точка( т2.х, н1.у);
```

Для поиска пересечения любых объектов можно использовать оператор "ПЕРЕСЕЧЕНИЕ". Для поиска пересечения прямых можно построить два достаточно длинных отрезка и пересечь их. Поиск пересечения прямых удобней производить не явно строя отрезки, а пользуясь функцией "пересечение_н" (пересечение направлений, задаваемых начальной точкой и направлением). Посмотрите на экране результат построения приведенной ниже программы и попробуйте вспомнить, в каких построениях требуется или можно использовать подобные действия.

```
т0:=точка( 0, 0);
т1:=точка( 1, 1);
т2:=точка( 2, 2);
пересечение_н( т1, 0, т2, 90, т3);
пересечение_н( т1, 10, т2, 92, т31);
пересечение_н( т1, 20, т2, 94, т32);
пересечение_н( т1, 30, т2, 96, т33);
пересечение_н( т1, 40, т2, 98, т34);
отрезок(т1, т2);
конец;
```

# 2. ОПЕРАТОР ЗАПИСИ КОНТУРА ЛЕКАЛА

Мы с Вами написали несколько программ построения простейших геометрических форм, которые могут быть использованы в конструировании лекал. Но, помимо самих построений, в программе должен быть оператор записи контура лекала,

который предназначен для вывода на печать. Для этого после завершения всех построений, касающихся данного лекала, следует функция "записать" (подробнее о синтаксисе этой функции можно узнать в разделе, содержащем описание языка).

Попробуйте построить многоугольник и вывести его на печать.

```
{--------Программа построения многоугольника--------}
 { расстановка точек }
 точ1:=точка( 0, 0);
 точ2:=точка( 15, 0);
 точ3:=точка( точ2.x, 15);
 точ4:=точка( (точ1.x+точ2.x)/2, точ3.x+3);
 точ5:=точка( точ1.x, точ3.y);
 { построение многоугольника }
 о1:=отрезок(точ1,точ2);
 о2:=отрезок(точ2,точ3);
 о3:=отрезок(точ3,точ4);
 о4:=отрезок(точ4,точ5);
 о5:=отрезок(точ5,точ1);
 { запись лекала }
 записать(имя = (прямоугольник),
 контур = (точ1,точ2,точ3,точ4,точ5),
 цвет = 10);
 конец
```

Напомним, что при записи контура, который состоит из отрезков, Вы можете указать только имена точек, система сама соединит их в замкнутый контур.

Итак, Вы уже можете строить различные фигуры, применяя те или иные средства языка, и вполне подготовлены к построению конструкций лекал.

# 3. УСТАНОВКА ОПОРНЫХ ТОЧЕК КОНСТРУКЦИЙ

Начнем с самого простого - построения сетки чертежа или установки опорных точек.

Попробуем с Вами построить прямую двухшовную юбку. В начале программы определим значения необходимых переменных, измерений и прибавок.

```
{-----------------------------------------------------}
 { построение прямой двухшовной юбки }
{-----------------------------------------------------}
{ ----------измерения---------- }
 рост:=164;
Ст:=36.5;
Сб:=50;
Ди:=60;
 { ----------прибавки---------- }
Псб:=1.5;
Пди:=1;
 { ------------------------------------------- }
 т:=точка( 0, 0);
```

```
б:=точка( 0, 19+0.25*(рост-164)); { точка уровня бедра на задней части }
н:=точка( 0, Ди+Пди); { точка уровня низа на задней части }
б1:=точка( б.х+Сб+Псб, б.у); { точка уровня бедра на передней части }
т10:=точка( б1.х, т.у-0.5); { точка уровня талии на передней части }
н10:=точка( б1.х, н.у+1); { точка уровня низа на передней части }
б2:=точка( б.х+0.5*(Сб+Псб)-1, б.у); { точка уровня бедра на боковом шве }
н2:=точка( б2.х, н.у);
т20:=точка( б2.х, т.у-1.5);
отрезок( т20, т10);
отрезок( т20, т);
сплайн_к( т20, т10, о_т20_т10.ф*2, 0, 1);
сплайн_к( т20, т, о_т20_т.ф*2-180, 180, 1);
Св:=Сб+Псб-Ст;
отложить_в( с_т20_т, Св*0.25, т21, у_т21);
отложить_в( с_т20_т10, Св*0.25, т22, у_т22);
отложить_в( с_т20_т, с_т20_т.л-0.4*(б2.х-б.х)+0.5*Св/3, т5, у_т5);
отложить_в( с_т20_т, с_т20_т.л-0.4*(б2.х-б.х)-0.5*Св/3, т6, у_т6);
б5:=точка( (т5.х+т6.х)/2, б.у-4);
отложить_в( с_т20_т10, с_т20_т.л-0.4*(б1.х-б2.х)+0.5*Св/6, т8, у_т8);
отложить_в( с_т20_т10, с_т20_т.л-0.4*(б1.х-б2.х)-0.5*Св/6, т7, у_т7);
б6:=точка( (т8.х+т7.х)/2, б.у-7);
конец
```

Мы не будем описывать контур лекал и выводить его на печать. Предоставим это Вам, тем более что Вы это уже умеете.

А теперь перейдем к более сложной конструкции - основе женского платья. Для иллюстрации возьмем методику Янчевской Е. А.

```
{=====================================================}
{Основа женского платья с втачными рукавами (Янчевская Е.А.) }
{=====================================================}
 { ---------- измерения ---------- }
Впрз:=17.3;
Дтс:=39.1;
Шс:=17.8;
Сг1:=44.2;
Сг2:=48.4;
Сг3:=46;
Ди:=100;
 перегибистая:=0.5;
 типовая:=1.0;
 сутулая:=1.5;
 да:=0.4;
 нет:=0;
 осанка:=типовая;
          { осанка (перегибистая/типовая/сутулая) }
 жир_отл_ш:=да;
         { наличие жирового отложения в области
           седьмого шейного позвонка (да/нет) }
{ ---------- припуски ---------- }
Пспр:=2;
Псут:=осанка;
Пдтс:=0.5;
Пшс:=0.2*Пг;
Пша:=0.1*Пг;
{ ---- Построение сетки чертежа ---- }
 а:=точка( 0, 0);
 г:=точка( а.х, а.у+Впрз+Пспр+Псут);
```

```
т0:=точка( а.х, а.у+Дтс);
т:=точка( т0.х, т0.у+Пдтс);
б:=точка( т.х, т.у+Дтс/2);
н:=точка( а.х, а.у+Ди);
а0:=точка( а.х, а.у-жир_отл_ш);
а_:=точка( а.х+Шс+Пшс, а.у); { ширина спинки платья }
а_1:=точка( а.х+Сг3+Пг, а.у); { ширина платья }
а_2:=точка( а_1.х-(Шг+(Сг2-Сг1)+Пшг), а_1.у); { ширина переда }
г1:=точка( а_.х, г.у);
г3:=точка( а_1.х, г.у);
г4:=точка( а_2.х, г.у);
б3:=точка( а_1.х, б.у);
н3:=точка( а_1.х, н.у);
отрезок ( а, н);
отрезок ( а, а_1);
отрезок ( а_1, г3);
отрезок ( г, г3);
отрезок ( а_, г1);
отрезок ( а_2, г4);
отрезок ( б, б3);
отрезок ( н, н3);
конец
```

Мы привели пример построения сетки чертежа женского платья. Обратите внимание на использование переменных. С помощью переменных "ОСАНКА" и "ЖИР_ОТЛ_Ш" введена параметризация. Теперь Вы можете, не изменяя построения, строить лекала на различные типы осанки.

При написании программ лучше каждый этап построения, в дополнение к комментариям, разделять на блоки (отделять пустыми строками или линиями). Кроме того, мы рекомендуем при нанесении сетки сначала расставить все основные точки, а затем указать, где необходимо соединить их отрезками, прямыми или сплайнами.

# 4. ПРИМЕР ПОСТРОЕНИЯ ТОЧКИ ПЛЕЧА

Вернемся к предыдущему примеру построения основы женского платья.

```
{ ---------- Чертеж спинки ---------- }
{ горловина спинки }
а1:=точка( а0.х+Сш/3+Пшгор, а0.у);
а2:=точка( а1.х, а1.у-к_плеч*Сш);
{ плечевой срез }
дуга_п( т0, Впк+Пр, а2, Шп, -1, п); { <<<<-------- }
отложить(а2, [а2:п].ф, Шп+раст_выт+пов_плеч, п1);
{ ! при сутулой фигуре берется коэффициент 0.5 }
отложить(а2, [а2:п].ф, 0.3*Шп, и);
дл_плеч_выт:=6;
и2:=точка( и.х, и.у+дл_плеч_выт);
отложить(и, [и:п1].ф, раст_выт, и1);
и3:=точка( и.х, и.у-0.3);
отложить(и2, [и2:и1].ф, |и2:и3|, и4);
отрезок ( а2, и3);
отрезок ( и3, и2);
```

*отрезок ( и2, и4);*
*отрезок ( и4, п1);*

В данном примере для построения плечевой точки используется пересечение двух дуг: из точки основания шеи радиусом, равным ширине плеча и из точки пересечения линии талии и позвоночника радиусом, равным величине измерения "высота плеча косая". Можно было бы выписать отдельно построение дуг и оператор поиска пересечения:

д1:=дуга( т0, Впк+Пр, -80, -40);
*д2:=дуга( а2, Шп, 0, 30);*
*пересечение( д1, д2, п);*

но такая запись занимает много места и загромождает чертеж дополнительными промежуточными построениями.

# 5. ПОСТРОЕНИЕ СПЛАЙНОВ

Сплайн в системе ЛЕКО - это плавная гладкая линия, соединяющая две точки. Для сплайна задаются начальная и конечная точки, касательные в начальной и конечной точках и дополнительное условие, определяющее форму сплайна. Опыт конструирования в системе ЛЕКО показал, что такое задание сплайнов очень удобно при построении лекал, т.к. при изменении координат точек и линий сопряжения сплайн сам автоматически настраивается на эти изменения. Использование дуг влечет за собой множество условий на точки и касательные, и не всегда существует дуга, удовлетворяющая всем конструктивным требованиям. Сплайн удобен еще и тем, что, если условия в начальной и конечной точках удовлетворяют условиям построения дуги, сплайн, построенный по этим условиям практически принимает форму дуги. Поэтому рекомендуем Вам использовать при построениях не дуги, а сплайны.

Самое простое построение сплайна - путем задания коэффициента "выпуклости" сплайна. Если Вы впервые используете сплайны для построения лекала, то возникает естественный вопрос: какой коэффициент задавать? Задайте коэффициент, равный 1 и посмотрите, что получится. Если получившаяся кривая Вас устраивает, то все хорошо и Вы можете продолжать построение лекала дальше. Если кривая слишком выпуклая - уменьшите коэффициент, если слишком плоская - увеличьте. Значения коэффициента, используемые для конструирования лекал, как правило, находятся в диапазоне 0.5 - 2.0. Вы можете задать сразу несколько сплайнов (веер), проходящих через одни и те же точки и имеющих одни и те же касательные, различающиеся значениями коэффициентов (от 0.5 до 2.0) и посмотреть, что получиться. Например:

*с1:=сплайн_к(точ1,точ2, 0, 90, 0.5);*
*с2:=сплайн_к(точ1,точ2, 0, 90, 0.7);*
*с3:=сплайн_к(точ1,точ2, 0, 90, 1.0);*
*с4:=сплайн_к(точ1,точ2, 0, 90, 1.2);*
*с5:=сплайн_к(точ1,точ2, 0, 90, 1.5);*
*с6:=сплайн_к(точ1,точ2, 0, 90, 2.0);*

Коэффициент, равный единице, при построении сплайна на точках окружности достаточно точно аппроксимирует дугу этой окружности. Если Вы используете методику построения лекала, основанную на радиусографии, то советуем Вам вместо дуг окружностей использовать сплайны с коэффициентом, равным 1. Это не только упростит запись методики, но и избавит Вас от поиска центров окружностей. Кроме того, сплайны позволяют проводить произвольную модификацию лекал, изменение координат конструктивных точек без пересчета радиуса и координат точек центра, что затруднительно при использовании дуг.

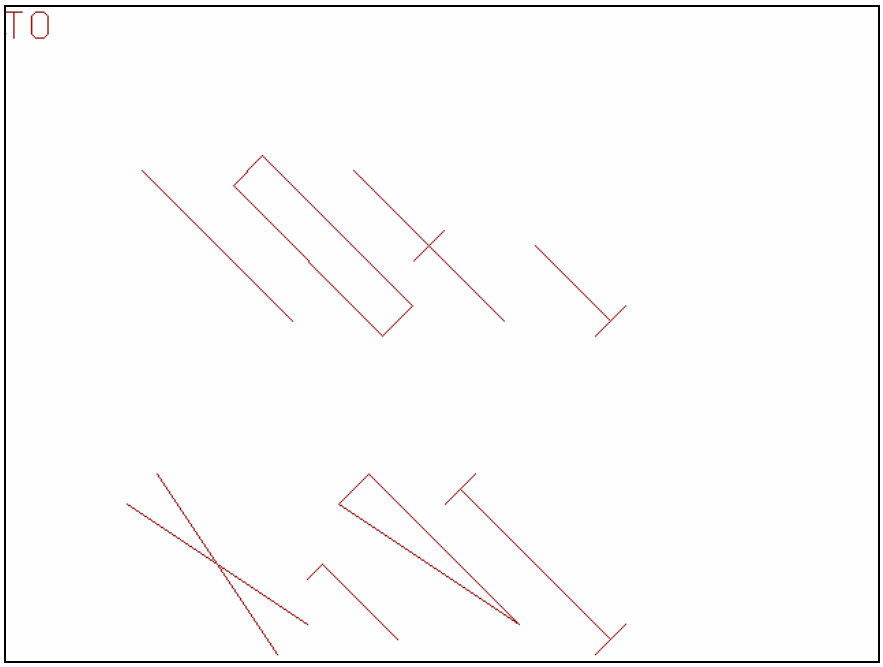
Заметим, что при описании лекала в системе ЛЕКО необходимо не только в точности реализовывать построение, описанное в методике, но и критически подходить к предлагаемому в методике построению. Некоторые методики, предлагая различные построения (простые или сложные), не гарантируют получения высококачественного лекала и подразумевают наличие у конструктора опыта по доработке лекала и посадке его на фигуру (типовую или индивидуальную).

# 6. ИСПОЛЬЗОВАНИЕ МЕТОК

В описании языка приведен список предусмотренных в системе типов меток. Однако для того, чтобы хорошо разобраться, какой тип меток в каком случае нужно использовать и как эти метки выглядят, рекомендуем посмотреть результат построения следующей программы:

```
м1:=метка(точка(10,10),1,45,10,2);
м2:=метка(точка(10,15),2,45,10,2);
м3:=метка(точка(10,20),3,45,10,2);
м4:=метка(точка(10,25),4,45,10,2);
м5:=метка(точка(10,30),5,45,10,2);
м6:=метка(точка(10,35),6,45,10,2);
м7:=метка(точка(10,40),7,45,10,2);
м8:=метка(точка(10,45),8,45,10,2);
конец
```

Метки будут расположены друг под другом, повернуты на 45 градусов. Обратите внимание на расположение метки относительно центра метки.



# 6. ЦИКЛИЧЕСКОЕ ПОСТРОЕНИЕ

Рассмотрим пример использования "циклического построения".

Одно из основных преимуществ такого построения - замена сложных формул набором условий и последовательная подгонка лекал.

Построим клин юбки. В качестве исходных параметров возьмем количество клиньев кл_кл, прибавки по талии и по бедрам. Длину юбки и расстояние от линии талии до линии бедер возьмем в пропорции к росту.

И так построение:

```
{ Ubka }

размеры;

{ ОСНОВНЫЕ ПАРАМЕТРЫ }

кл_кл:=4;

п18:=1;
п19:=5;
ди:=рз_1*0.3;
вб:=рз_1*0.1;
шир_кл_т:=(рз_18+п18)/кл_кл;
шир_кл_б:=(рз_19+п19)/кл_кл;

{ ПОСТРОЕНИЕ ЧЕРТЕЖА }
```

```
т1:=точка(10,10);
т2:=отложить(т1,0,шир_кл_т);
т3:=отложить(т1,90,ди);
т4:=отложить(т3,0,шир_кл_т);
т5:=отложить(т1,90,вб);
т6:=отложить(т2,90,вб);

лт:=отрезок[100](т1,т2);
лб:=отрезок[100](т5,т6);
лн:=отрезок[100](т3,т4);

сч_ц:=0;
ц_начало;
развести((лт,лб,лн),лт,-90,-90,-сч_ц,0,"_р");
если больше(лб_р.л,шир_кл_б) то
ц_прекратить;
иначе
сч_ц:=сч_ц+1;
конец_если;

если больше(сч_ц,300) то
ц_прекратить;
конец_если;

ц_конец;

ЗАПИСАТЬ(имя=(клин),
кд=(кл_кл,0),
полотно=верх,
контур=(ЛТ_Р,-ЛН_Р),
прибавка=1,
цвет=12);

конец
```

Построив конструкцию до цикла (установите курсор на строку до начала цикла и нажмите на клавишу F4) мы увидим прямоугольник. В цикле он разводится на увеличивающееся значение угла сч_ц до тех пор пока длина по линии бедер не станет больше (рз_19+п19)/кл_кл.

Измените значения кл_кл, п18, п19 и посмотрите на результат.

# Содержание

**ЯЗЫК ОПИСАНИЯ И ПОСТРОЕНИЯ ЛЕКАЛ ..... 1**

**ВВЕДЕНИЕ ..... 1**

**1. ОСНОВНЫЕ ЭЛЕМЕНТЫ ЯЗЫКА ..... 3**
1.1. КЛЮЧЕВЫЕ СЛОВА ..... 3
1.2. ИДЕНТИФИКАТОР ..... 4
1.3. ОПРЕДЕЛЕНИЕ ПЕРЕМЕННЫХ ..... 4
1.4. ЧИСЛА ..... 5
1.5. КООРДИНАТНЫЕ ОСИ ..... 6
1.6. ТОЧКИ ..... 6
1.7. ОТРЕЗКИ ..... 8
1.8. ДУГИ ..... 9
1.9. СПЛАЙНЫ ..... 9
1.10. ГЛАДКОСТЬ КРИВЫХ ПРИ ПОСТРОЕНИИ ..... 13
1.11. ЛОМАНЫЕ ..... 13

**2. ОПЕРАТОРЫ ПРОГРАММЫ ..... 14**

**3. КВАЛИФИКАТОРЫ ..... 15**

**4. УГЛЫ ..... 16**

**5. НЕЯВНОЕ ЗАДАНИЕ ОТРЕЗКОВ ..... 18**

**6. КОММЕНТАРИЙ ..... 18**

**7. ПОДКЛЮЧЕНИЕ ТАБЛИЦЫ РАЗМЕРНЫХ ПРИЗНАКОВ ..... 19**

**8. ВСТРОЕННЫЕ ФУНКЦИИ ЯЗЫКА ..... 20**
8.1. СИММЕТРИЯ ..... 20
8.2. ПЕРЕНОС ..... 22
8.3. ПОВОРОТ ..... 23
8.4. СЖАТИЕ ..... 23
8.5. УСТАНОВКА ТОЧКИ ..... 24
8.6. ПЕРЕСЕЧЕНИЕ ..... 28
8.7. ПЕРЕСЕЧЕНИЕ ДУГ ..... 29
8.8. ПЕРЕСЕЧЕНИЕ НАПРАВЛЕНИЙ ..... 29
8.9. ПЕРЕСЕЧЕНИЕ ДУГИ И НАПРАВЛЕНИЯ ..... 31
8.10. СОПРЯЖЕНИЕ ЛИНИЙ ДУГАМИ ..... 32
8.11. ПОДКЛЮЧЕНИЕ ТАБЛИЦЫ РАЗМЕРНЫХ ПРИЗНАКОВ ..... 33
8.12. ИСПОЛЬЗОВАНИЕ ПАРАМЕТРОВ ПОСТРОЕНИЯ ..... 33
8.13. ВЫТАЧКА ..... 35
8.14.ОБРЕЗАТЬ ..... 36
8.15. СОВМЕСТИТЬ ..... 38
8.16. РАЗВЕСТИ ..... 39
8.16. ВОЛАН ..... 41
8.17. НАПЕЧАТАТЬ ..... 41
8.18. ЦВЕТ ЛИНИЙ ..... 43

**9. АВТОМАТИЧЕСКАЯ ГЕНЕРАЦИЯ ИДЕНТИФИКАТОРОВ.....43**

**10. ФОРМИРОВАНИЕ КОНТУРА ЛЕКАЛА.....44**
10.1. МЕТКИ.....46
10.2. ИСПОЛЬЗОВАНИЕ ЦВЕТА.....48
10.3. ФОРМИРОВАНИЕ ПРИБАВОК НА ШВЫ.....49
10.4. СИММЕТРИЯ.....51
10.5. ОБЪЕДИНЕНИЕ ЛЕКАЛ.....52
10.5. РАЗВЕДЕНИЕ.....53

**11. ДОПОЛНИТЕЛЬНЫЕ ВОЗМОЖНОСТИ ПРЕОБРАЗОВАНИЯ ОБЪЕКТОВ .....54**

**12. ПРОСМОТР ЛЕКАЛ.....54**

**13. ТОЧНОСТЬ РАСЧЕТОВ .....55**

**14. НОВЫЕ ЭЛЕМЕНТЫ ЯЗЫКА.....56**
14.1. НОВЫЙ ТИП ДАННЫХ .....56
14.2. НОВЫЕ ОПЕРАТОРЫ .....56
14.3. НОВЫЕ ВОЗМОЖНОСТИ ЗАПИСИ ОПЕРАТОРОВ.....60
14.4. ОПЕРАТОР "ЗАПИСАТЬ" .....62
14.4. 3D ОПЕРАТОРЫ .....62
14.4. УСТАРЕВШИЕ ОПЕРАТОРЫ.....64

**ПРИЛОЖЕНИЕ.....67**
Разделители : .....67
Ключевые слова: .....67
Квалификаторы.....68

**КОНСТРУИРОВАНИЕ В СИСТЕМЕ ЛЕКО .....69**

**1. ПОСЛЕДОВАТЕЛЬНОСТЬ РАЗРАБОТКИ АЛГОРИТМА ПОСТРОЕНИЯ ЛЕКАЛА .....70**

**2. ОСНОВА КОНСТРУКЦИИ БРЮК.....71**

**3. ЗАПИСЬ ФОРМУЛ.....76**

**4. ИСПОЛЬЗОВАНИЕ БАЗЫ ДАННЫХ РАЗМЕРНЫХ ПРИЗНАКОВ..80**

**5. "АЛГОРИТМИЗАЦИЯ" ПОСТРОЕНИЯ ЛЕКАЛА .....81**

**6. ПОСТРОЕНИЕ ЛЕКАЛ С ИСПОЛЬЗОВАНИЕМ УГЛОВЫХ ВЕЛИЧИН 81**

**7. ПАРАМЕТРИЗАЦИЯ ПОСТРОЕНИЯ МОДЕЛЕЙ.....82**

**8. СТАНДАРТИЗАЦИЯ МЕТОДОВ ПОСТРОЕНИЯ И ОПИСАНИЯ .....84**

**9. УНИФИКАЦИЯ ОБОЗНАЧЕНИЙ.....85**

**10. НАКОПЛЕНИЕ ОПЫТА.....86**

**11. ТЕХНИЧЕСКОЕ РАЗМНОЖЕНИЕ ЛЕКАЛ.....86**

**12. ВЕДЕНИЕ БАЗЫ ДАННЫХ ПО МОДЕЛЯМ .....86**

**13. САМОДОКУМЕНТИРУЕМОСТЬ .....87**

**14. ПРОДАЖА АЛГОРИТМОВ.....87**

**15. ДОПОЛНИТЕЛЬНЫЕ ЗАМЕЧАНИЯ ..... 88**

**РАБОТА С БАЗОЙ ДАННЫХ РАЗМЕРНЫХ ПРИЗНАКОВ ..... 95**

**ВВЕДЕНИЕ ..... 95**

**2. БАЗЫ ДАННЫХ РАЗМЕРНЫХ ПРИЗНАКОВ, ПОСТАВЛЯЕМЫЕ С СИСТЕМОЙ ..... 96**

**3. ПОРЯДОК СНЯТИЯ РАЗМЕРНЫХ ПРИЗНАКОВ ..... 96**

**ПРИМЕРЫ ..... 105**

**1. НАПИСАНИЕ ПРОГРАММЫ ПОСТРОЕНИЯ ТОЧКИ..... 105**

1.1. РАСШИРЕНИЕ ПРОГРАММЫ ..... 107

1.2. ПОИСК ПЕРЕСЕЧЕНИЯ ПРЯМЫХ..... 111

**2. ОПЕРАТОР ЗАПИСИ КОНТУРА ЛЕКАЛА..... 111**

**3. УСТАНОВКА ОПОРНЫХ ТОЧЕК КОНСТРУКЦИЙ ..... 112**

**4. ПРИМЕР ПОСТРОЕНИЯ ТОЧКИ ПЛЕЧА..... 114**

**5. ПОСТРОЕНИЕ СПЛАЙНОВ ..... 115**

**6. ИСПОЛЬЗОВАНИЕ МЕТОК ..... 116**

**6. ЦИКЛИЧЕСКОЕ ПОСТРОЕНИЕ..... 117**

**СОДЕРЖАНИЕ ..... 119**