1. **Основные положения структурного программирования.**

1) Любая программа представляет собой структуру, построенную из трех типов базовых конструкций:

А) Последовательно исполнение - однократное выполнения операций в том порядке, в котором они записаны в тексте программы.

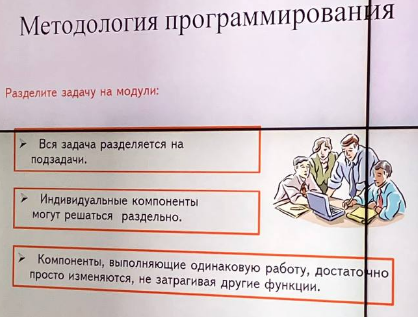
Б) Ветвление - однократное выполнение одной из двух или более операций в зависимости от выполнения некоторого заданного условия.

В) Цикл - многократное исполнение одной и той же операции до тех пор, пока выполняется некоторое заданное условие (Условие продолжение цикла).

2) Повторяющиеся фрагменты программы (либо, не повторяющиеся, но представляющие собой логически целостные вычислительные блоки) могут оформляться в виде подпрограмм (процедур или функций).

3) Разработка программы ведётся пошагово, методом “сверху вниз”.

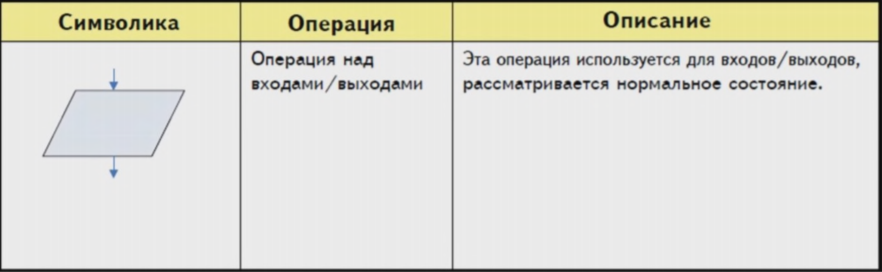
1. **Методология структурного программирования.**

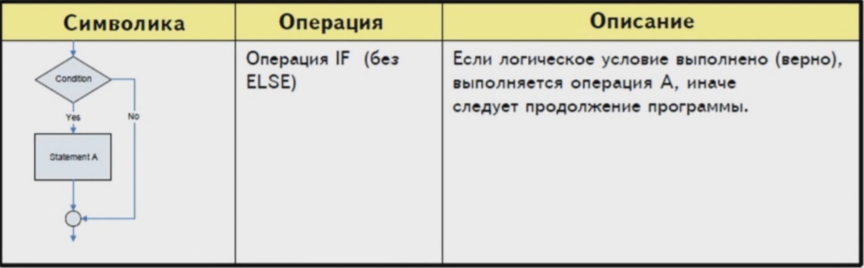


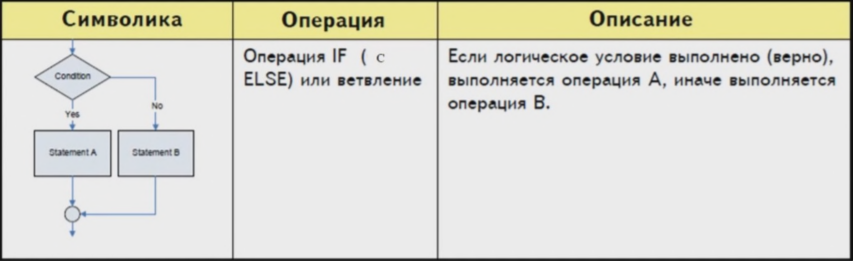
Вся задача разделяется на подзадачи «опускаясь» до формулировок, имеющих прямое соответствие в командах языка программирования. Схожие компоненты объединяются в модули, что упрощает их дальнейшую отладку. Малоотличающиеся модулю так же целесообразно объединять в единые блоки.

1. **Основные элементы блок-схем.**

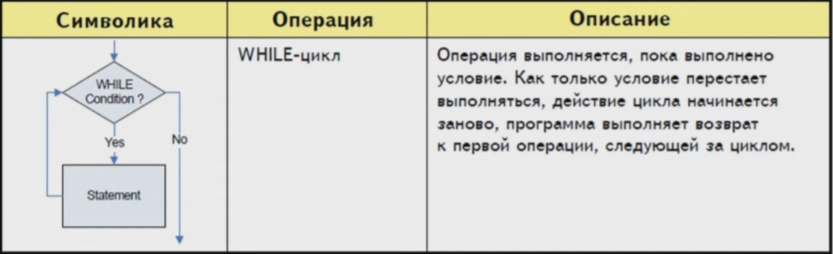
****

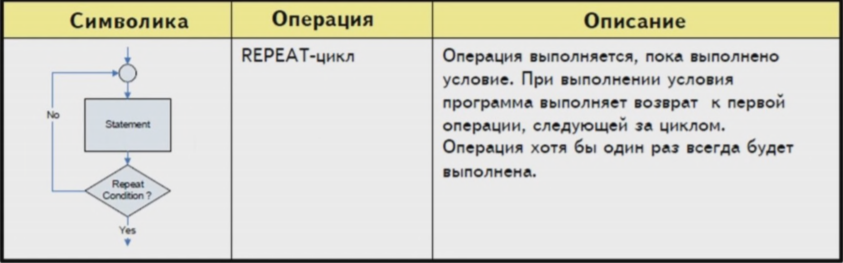
****

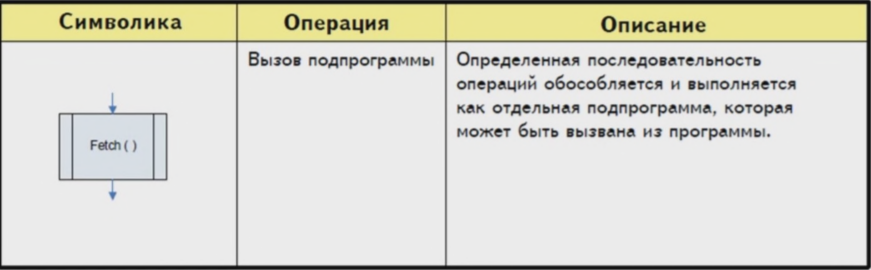
****

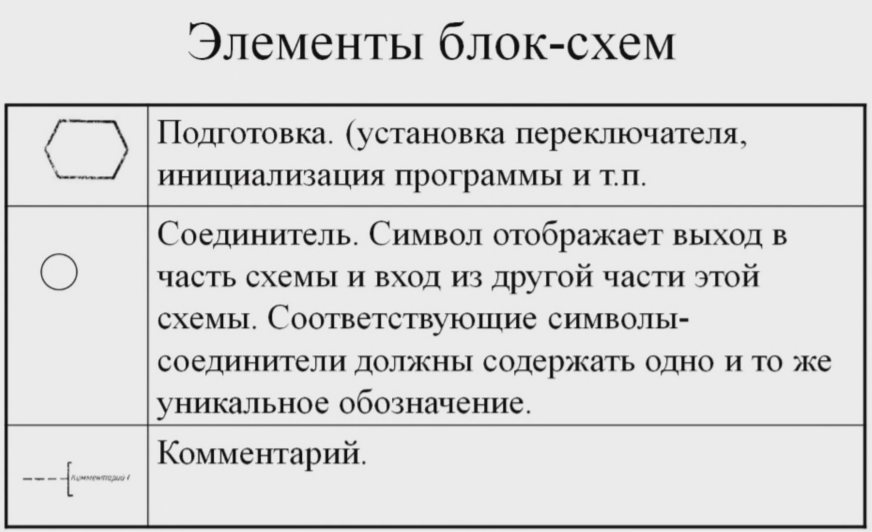
****

****

****

****

****

****

1. **Какой основной подход к выбору имён переменных при программировании?**
   * Имена для данных должны выбираться по смыслу
2. **Применение оператора SIGNAL.**

Оператор SIGNAL используется для привязки входных и выходных сигналов к определенным портам.

1. **Зачем применять отступы в тексте программ. Как это сделать?**

Инструмент для придания наглядности программе – это выделение частей программы с помощью отступов.

Для выделения структур в пределах программных модулей рекомендуется размещать вложенные иерархические последовательности команд с одинаковым отступом (от края, непосредственно одну под другой)

Делают отступы при помощи пробела.

1. **Какие правила указания комментариев? Как это сделать?**

В ходе отладки программы необходимо следить за актуальностью содержимого комментария и при внесении изменений в программу изменять содержимое комментариев

В крл применяются три способа задания комментариев.

⁃ простановка в любой части программы точки с запятой и написания после нее комментария. Данное действие возможно только в режиме эксперта и выше.

⁃ Вторым способом является размещение нового комментария через формуляр. Команды-комментарий-нормальный. Возможно только на новую строке, посреди строки нельзя. (команды – комментарий – нормальный)

⁃ Комментарий штамп. Идея как в предыдущем, но появляется время, дата и кто сделал. Имя заполняется в последствии автоматически (команды – комментарий – штамп)

1. **Чем отличается стандартный комментарий от штампа?**

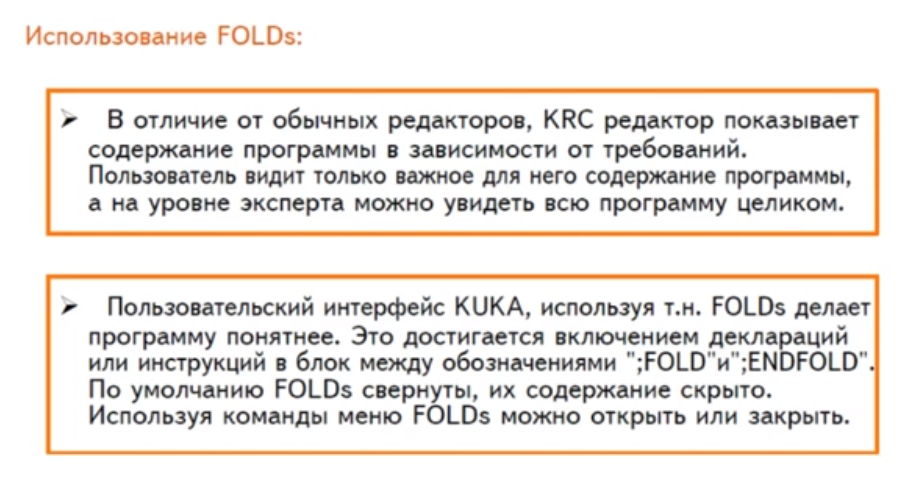
Стандартный комментарий имеет только одно окно ввода комментария в своем формуляре. В то время как штамп имеет дату и системное время, окно для имени пользователя и окно текста комментария или изменения по тексту, которые вы привнесли.

1. **Как сделать комментарий внутри строки?**

В режиме эксперта «Expert mode» программист может преобразовать любую строку в комментарий добавлением символа точки с запятой - «;».

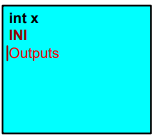
Так же с помощью этого символа возможно отделения комментария в конце командной строки.

1. **Зачем применяются структуры типа "FOLDs"?**

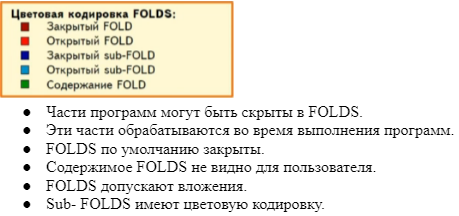


1. **Как открыть или закрыть "FOLDs"? Синтаксис "FOLDs" в тексте программы.**

Программа-Fold-Открыть (Закрыть)

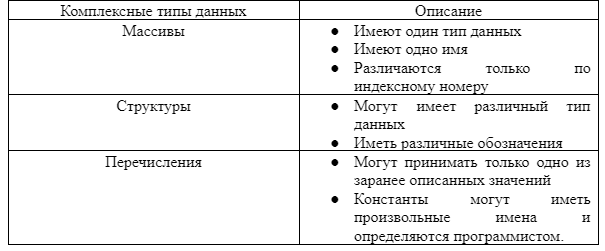
1. **Какие цветовые маркировки "FOLDs"?**



1. **Какие типы простых данных Вы знаете?**



1. **Какие типы комплексных данных Вы знаете?**



1. **Как декларируются переменные в программе?**

Декларирование переменных в SRC файлах происходит в самом начале программы.

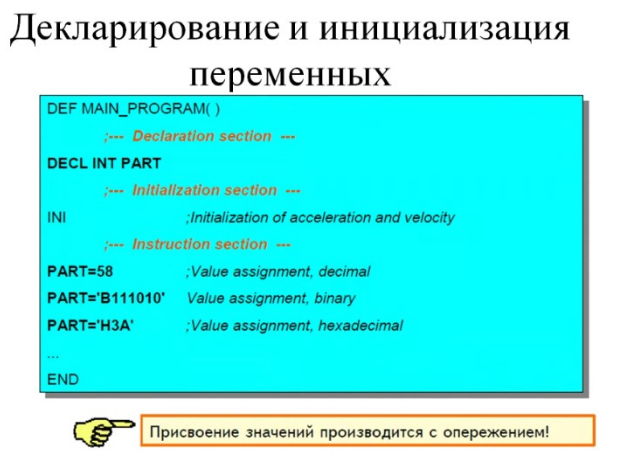
Они доступны только в своих программах и недоступны в локальных подпрограммах.

1. **Как ограничиваются имена переменных в KRL?**
   1. Длинна, не должна превышать 24 символа
   2. Могут состоять из букв (A-Z), цифр (0-9), символа ‘\_’ и ‘$’
   3. Не должны начинаться с цифры
   4. Не должны совпадать с признаком переменной.
2. **Как инициализируются переменные в программе?**

**«Название переменной» = «значение переменной»**

**А = 0**

1. **Как задать бинарное или шестнадцатеричное значение переменной?**

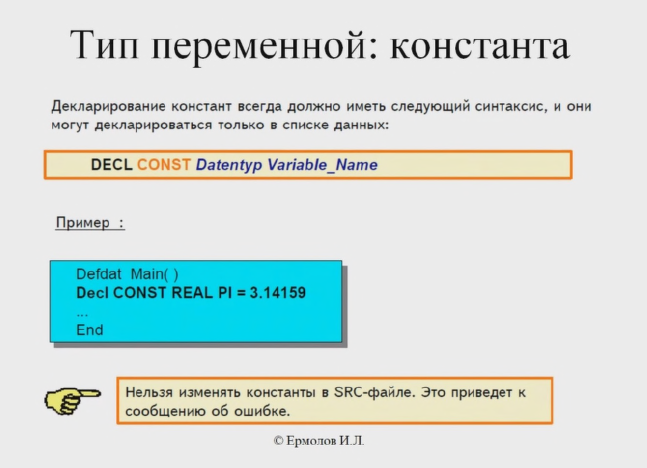


1. **Как доступны переменные, задекларированные в различных файлах?**

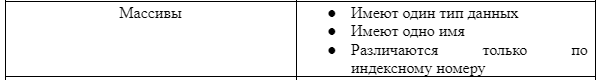
****

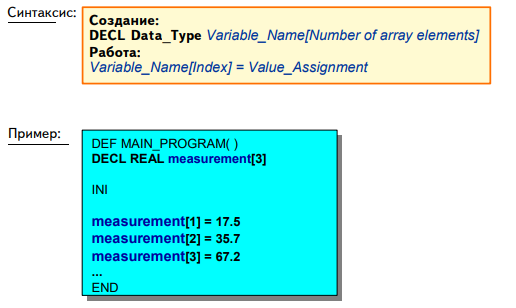
1. **Тип переменной – константы. Способ декларирования и применения.**

Для использования констант (цифр, символов и т.д.) в системе управления роботом выделяются области памяти, в которых эти значения могут быть сохранены.



1. **Тип переменной – массив. Способ декларирования и применения.**





1. **Опишите переменную типа "Структура".**

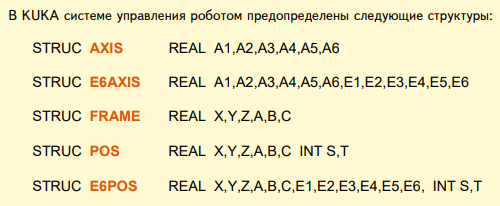
Структура – совокупность различных типов данных, объединённых в группу. Инициализируется структура совокупно. Порядок следования параметров не важен. Все параметры указывать не обязательно. Работать с параметрами можно отдельно.

Наименование созданных пользователем структур должно оканчиваться на «…type»

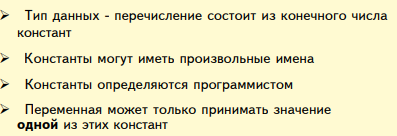
Пример



1. **Какие системные переменные типа "Структура" Вы знаете?**



1. **Опишите переменную типа "Перечисление".**



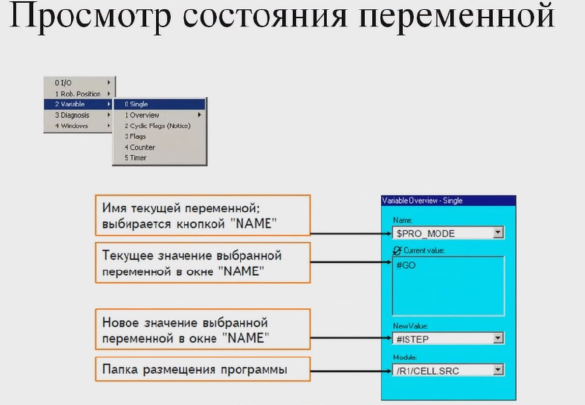


1. **Какие системные переменные типа " Перечисление" Вы знаете?**



1. **Как узнать значение переменной?**

Необходимо зайти в Variable-Single, где и можно узнать текущее значение переменной.



1. **Зачем применяются подпрограммы?**

Подпрограммы используются для частей программ, которые часто используются.

• Подпрограммы сокращают объем ввода текста во время программирования.

• Подпрограммы сокращают длину программы, таким образом, делая программу более ясной.

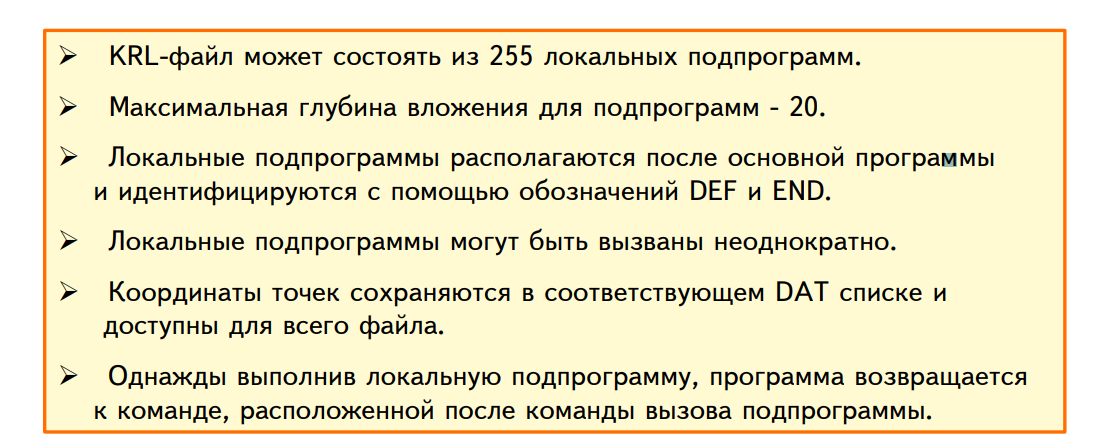
• Подпрограммы могут повторно использоваться в других программах.

• Подпрограммы могут использоваться для структурирования программы.

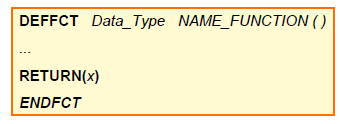
1. **Сколько может быть в программе подпрограмм? Какова глубина вложения подпрограмм?**

KRL-файл может состоять из 255 локальных подпрограмм.

Максимальная глубина вложения для подпрограмм - 20.

источник****

1. **Как идентифицируется подпрограмма?**

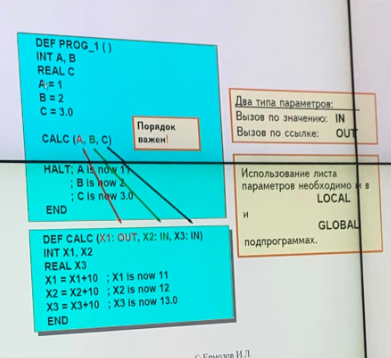


Внутри скобок указываем принимаемые переменные и тип in или out

1. **Чем отличается локальная подпрограмма от глобальной подпрограммы?**

Локальные подпрограммы располагаются после основной программы и идентифицируются с помощью обозначений def и end. Глобальные подпрограммы состоят из отдельного SRC файла

1. **Как осуществляется обмен параметрами между программой и подпрограммой?**



1. **Зачем используются подпрограммы типа "Функция"?**

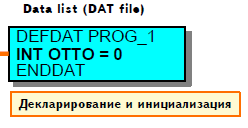
Для того, чтобы производить какие-либо действия(вычисления) и передавать значение обратно в основную программу

1. **Как вызываются подпрограммы типа "Функция"?**

PROG\_1(A,B,C), где АВС – переменные, которые нужно передать функции

1. **Что такое список данных? Какие данные там хранятся? Как они декларируются и инициализируются?**

Хранятся переменные, координаты, сигналы и пр.,



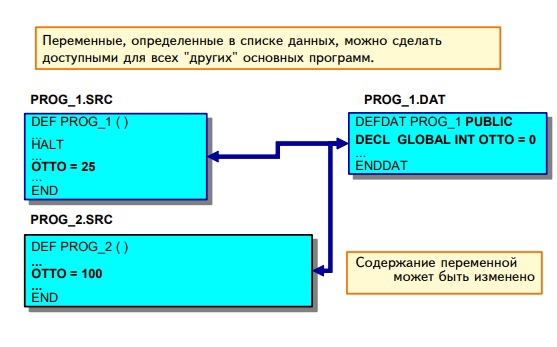
1. **Где хранятся глобальные списки данных?**

В директории STEU: $config.dat, $machine.dat, $custom.dat, $robcor.dat

1. **Как импортируется переменная из другого файла? Какая особенность использования такой переменной?**

IMPORT INT OTTO\_2 IS /R1/PROG\_1..OTTO (Переменные определенные в списке данных могут быть импортированы в другую основную программу, содержание переменной не может быть изменено)

1. **Как создать глобальную переменную?**



1. **Зачем создавать глобальную переменную?**

Для того чтобы они были доступны в каждой программе и использования в окне переменных, а также могли быть вызваны во время выполнения программы.