Подпрограммы

Подпрограммы

Подпрограммы – это средство структурирования программ, идея которого заключается в том:

- чтобы программа состояла не из громадного количества операторов;
- чтобы программа состояла из относительно самостоятельных частей;
- чтобы каждой части назначена отдельная, сравнительно узкая роль;
- чтобы программа не состояла из многочисленных переходов

- Программы, которые сделаны удобочитаемыми еще на этапе их написания, также легки при отладке и в обслуживании.

Преимущества технологии программирования с использованием подпрограмм

- организация работы нескольких программистов над одной программой с последующим объединением отдельно отлаженных и относительно независимых блоков в единое целое;
- отладка отдельных блоков и только после этого программы в целом;
- значительная экономия памяти, так как многократно используемый участок заносится в память только один раз;
- упрощение внесения изменений в программу, так как исправление ошибки в одном блоке не вызывает корректировку других блоков

Определение

- **Подпрограмма** – функционально самостоятельная часть программы, обладающая собственным именем и набором локальных имен.

Объявление подпрограммы

- Подпрограмма, чтобы ее можно было вызвать в программе, должна быть объявлена в разделе описания программы
- Объявить подпрограмму – значит указать ее заголовок (с используемыми в ней формальными параметрами), описать локальные переменные и, наконец, задать ее тело.

В языке программирования Паскаль приняты два вида подпрограмм: **функции и процедуры**

Особенности подпрограммы - функции

- Результат работы – единственное значение, которое передается в программу
- Это значение несет имя функции

Структура подпрограммы - функции

Function ИМЯ (формальные параметры):

тип результата;

Var

блок описания локальных переменных

Begin

тело **подпрограммы - функции**

ИМЯ:=результат;

End;

Вызов подпрограммы - функции

- Вызов функции пользователя осуществляется как вызов любой стандартной функции из любых точек программы и любое количество раз

f:= sqrt(a+s);

Переменная :=

имя функции (фактический параметр1, фактический параметр2,. . .);

Пример

Подпрограмма:

function **имя (формальный параметр1, формальный параметр2…): тип;**

```
function SUMMA (A:MAS; N,M: INTEGER): REAL;
……
        SUMMA:=……..
           end;
```

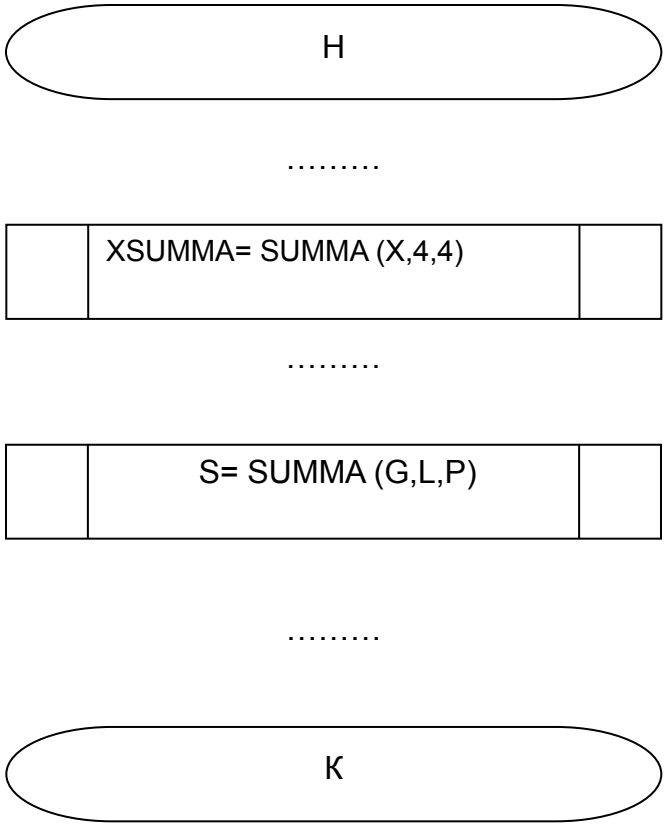
Примечание:
TYPE MAS = array [1..10, 1..10] of real;

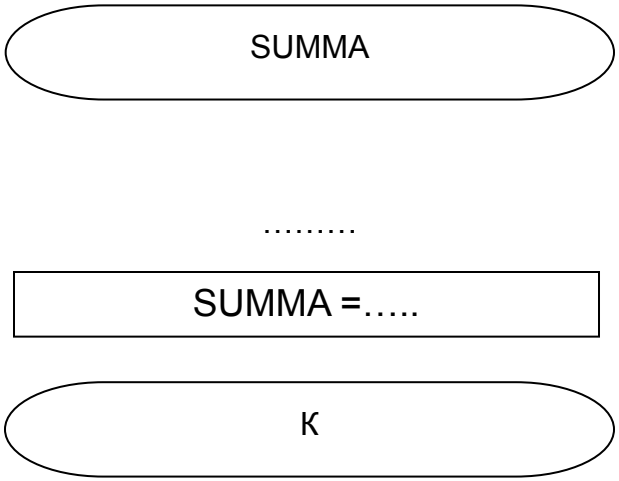
Вызов подпрограммы:

Идентификатор:=
имя функции (фактический параметр1, фактический параметр2. . .);

```
 XSUMMA:= SUMMA (X,4,4);
 S:= SUMMA (G,L,P);
WRITELN ('Сумма элементов массива C =', SUMMA (C,3,5):10:2 );
```

Схемы алгоритмов основной программы и подпрограммы





Структура процедуры

Procedure имя (формальные параметры);

Var

блок описания локальных переменных

Begin

тело процедуры

End;

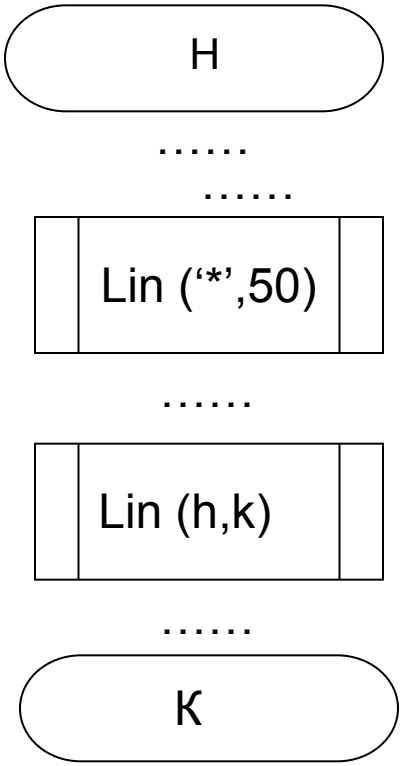
Вызов процедуры

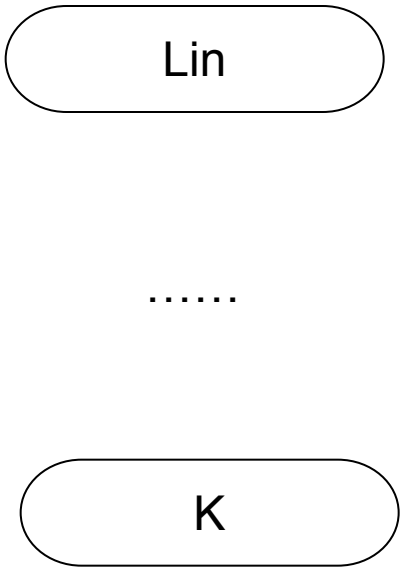
- Вызов осуществляется по имени процедуры из любых точек программы и любое количество раз

Имя_процедуры (параметр1, параметр2, . . .);

Схема алгоритма

(основная программа и подпрограмма)





Параметры

Параметры обеспечивают механизм замены, который позволяет выполнять процедуру с различными данными.

Между фактическими параметрами в операторе вызова процедуры и формальными параметрами в заголовке описания процедуры устанавливается соответствие:

- по количеству;
- типу (real, integer…);
- по сущности (переменная, массив…)

Пример

формальные параметры

Процедура **ЭКЗАМЕН** (студент, предмет, дата, оценка)
ВЫЗОВ

фактические параметры

ЭКЗАМЕН(Иванов, Программирование, 13.01, оценка);
ЭКЗАМЕН(Михеев, Информатика, 38.02, оценка);
студент := Василевский;
предмет := Математический анализ;
ЭКЗАМЕН(студент, предмет, 24.12, оценка);
ЭКЗАМЕН(23.09, Сидоров, История, оценка);

Параметры-значения
Параметры - переменные

Если процедура возвращает в программу какие-то значения, соответствующие переменные должны быть описаны как параметры-переменные с использованием оператора Var, исходные данные для подпрограммы – параметры-значения.

Procedure center (a:real, Var h:real, y: real);

Var предшествует тем параметрам, значения которых должны быть (или могут быть) изменены процедурой!!!

ПРИ ВЫЗОВЕ ПРОЦЕДУРЫ ВЫПОЛНЯЮТСЯ СЛЕДУЮЩИЕ ДЕЙСТВИЯ

1. Формальные параметры заменяются фактическими.
2. Выполняется тело процедуры.
3. Происходит возврат в вызывающую программу.
4. После вызова процедуры выполняется оператор, следующий за вызовом.

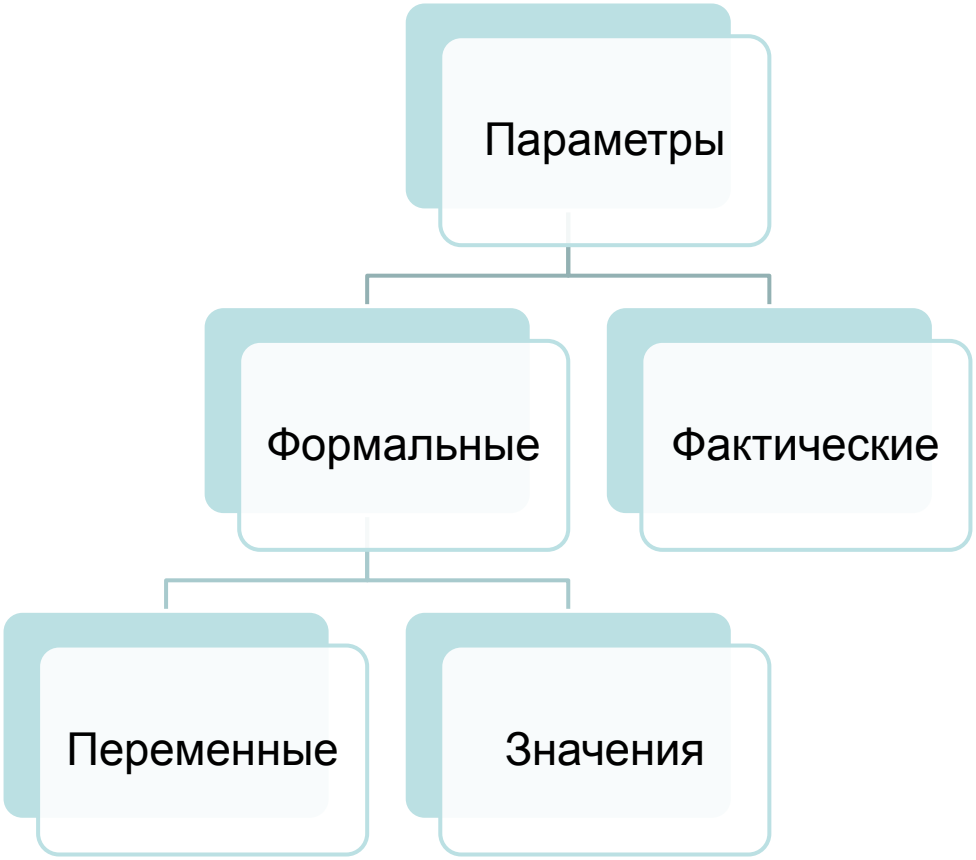
ПРИНЦИПИАЛЬНАЯ СТРУКТУРА ПРОГРАММЫ

PROGRAM Имя программы;
USES
    Список используемых библиотек (модулей);
CONST
    Определение констант программы;
TYPE
    Описание типов;
VAR
    Определение глобальных переменных программы;

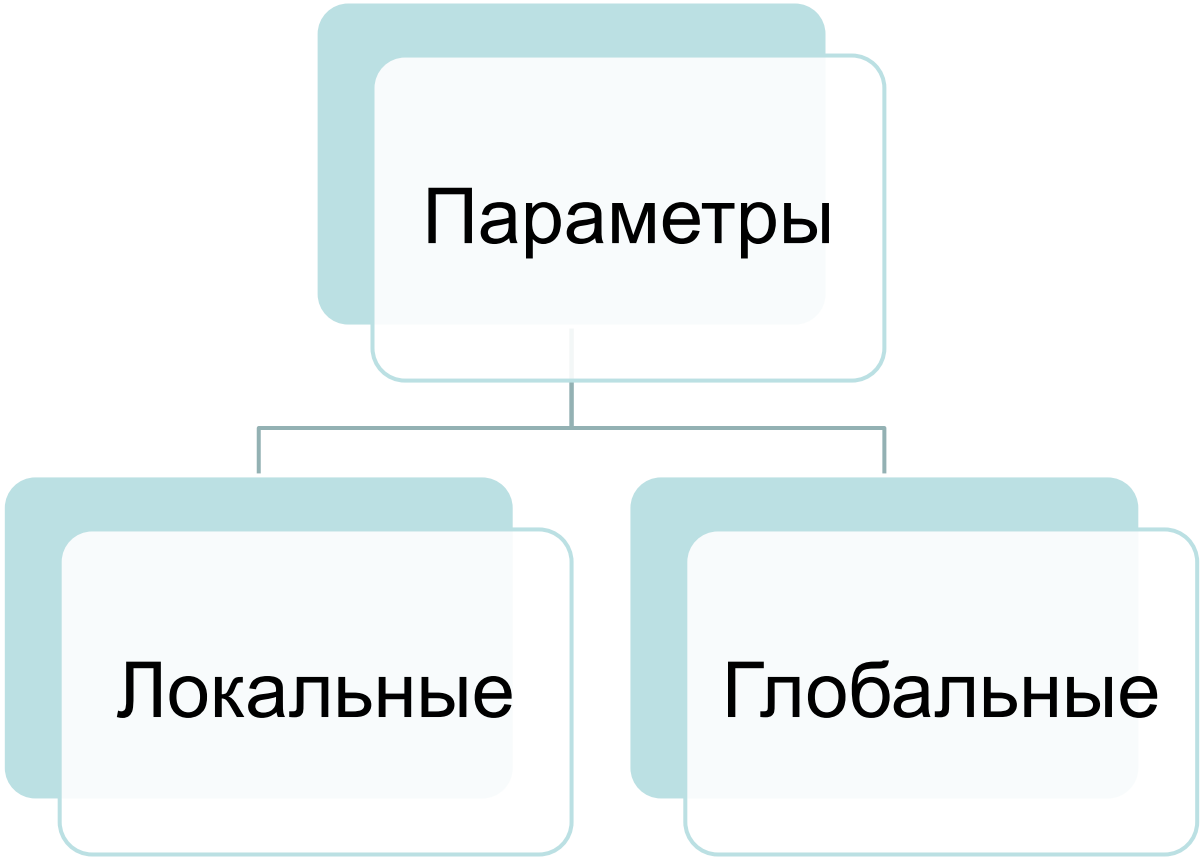
ОПРЕДЕЛЕНИЕ ПРОЦЕДУР (заголовки и, возможно тела процедур)
ОПРЕДЕЛЕНИЕ ФУНКЦИЙ(заголовки и, возможно тела функций)

BEGIN
    Основной блок программы (тело программы)
    ……
END.

Параметры процедур



Описание параметров



Основные отличия между **Function** и **Procedure**

- Вызов
  - **f:= DAG(a,s);**
  - **ST(X,NR,0,K);**
- Результат
  - **Только один** (имени функции присваивается результат)
  - Любое количество (или отсутствие)
- Описание
  - **Указание типа результата**
  - Указание типа для параметров-переменных