Информатика

ЗАПИСЬ ВСПОМОГАТЕЛЬНЫХ АЛГОРИТМОВ НА ЯЗЫКЕ ПАСКАЛЬ

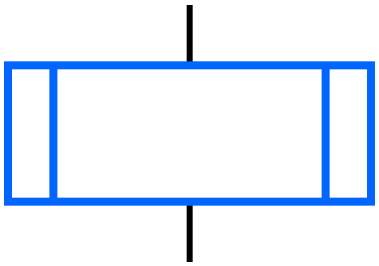
- подпрограмма
- процедура
- функция
- рекурсивная функция

9 класс

Издательство БИНОМ

Вспомогательный алгоритм

Вспомогательный алгоритм - алгоритм, целиком используемый в составе другого алгоритма.



Блок «предопределённый процесс»

Вспомогательный алгоритм делает структуру алгоритма более простой и понятной.

Алгоритм вычисления степени

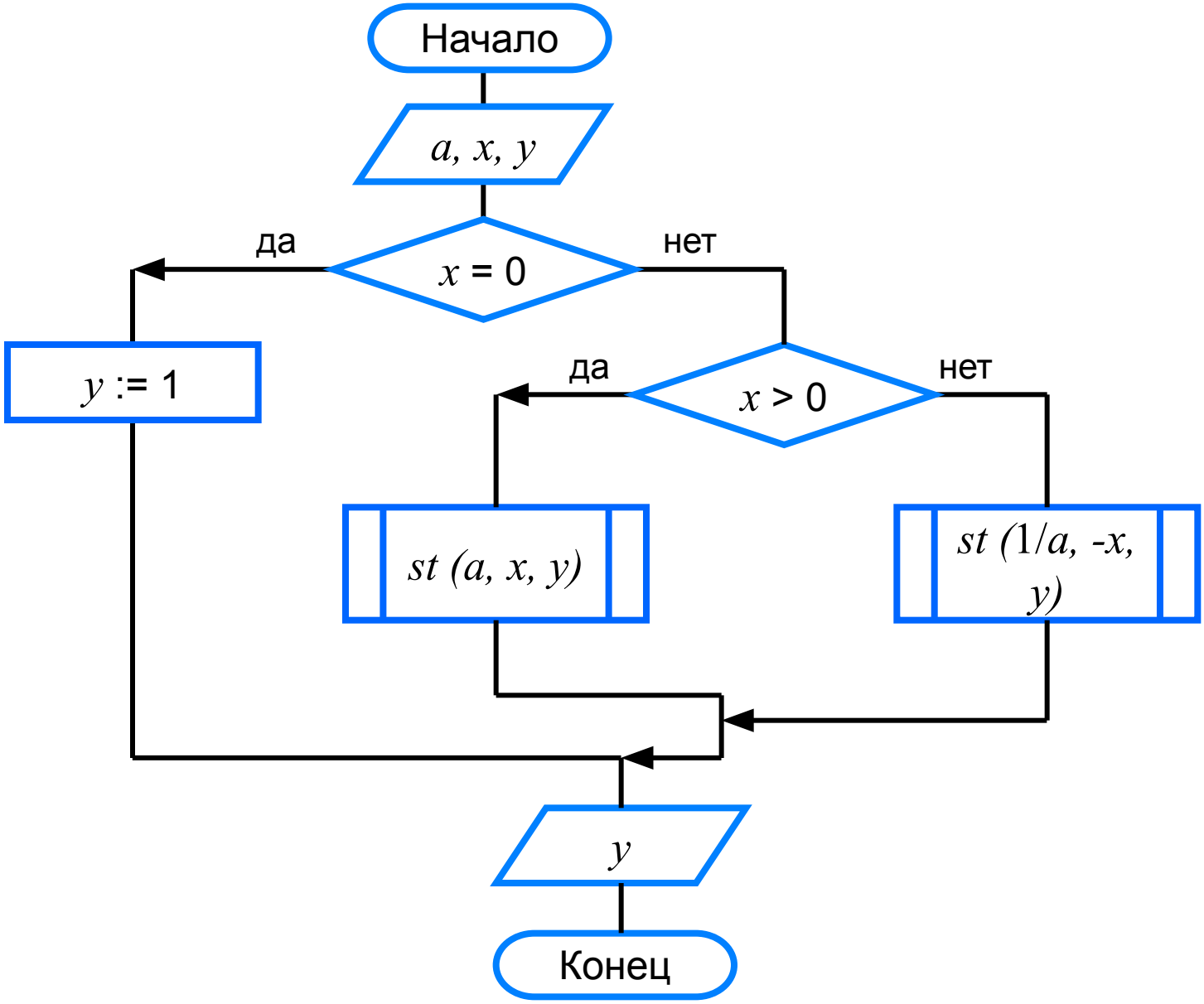
$y = a^x$, где x - целое число, $a \neq 0$.

$$Y=\begin{cases} 1, \text{при } x = 0 \\ a^x, \text{при } x > 0 \\ \frac{1}{a^x}, \text{при } x < 0 \end{cases}$$

Обозначим алгоритм возведения числа в степень ***st(a, n, y).***

Это вспомогательный алгоритм.

Блок-схема решения задачи:



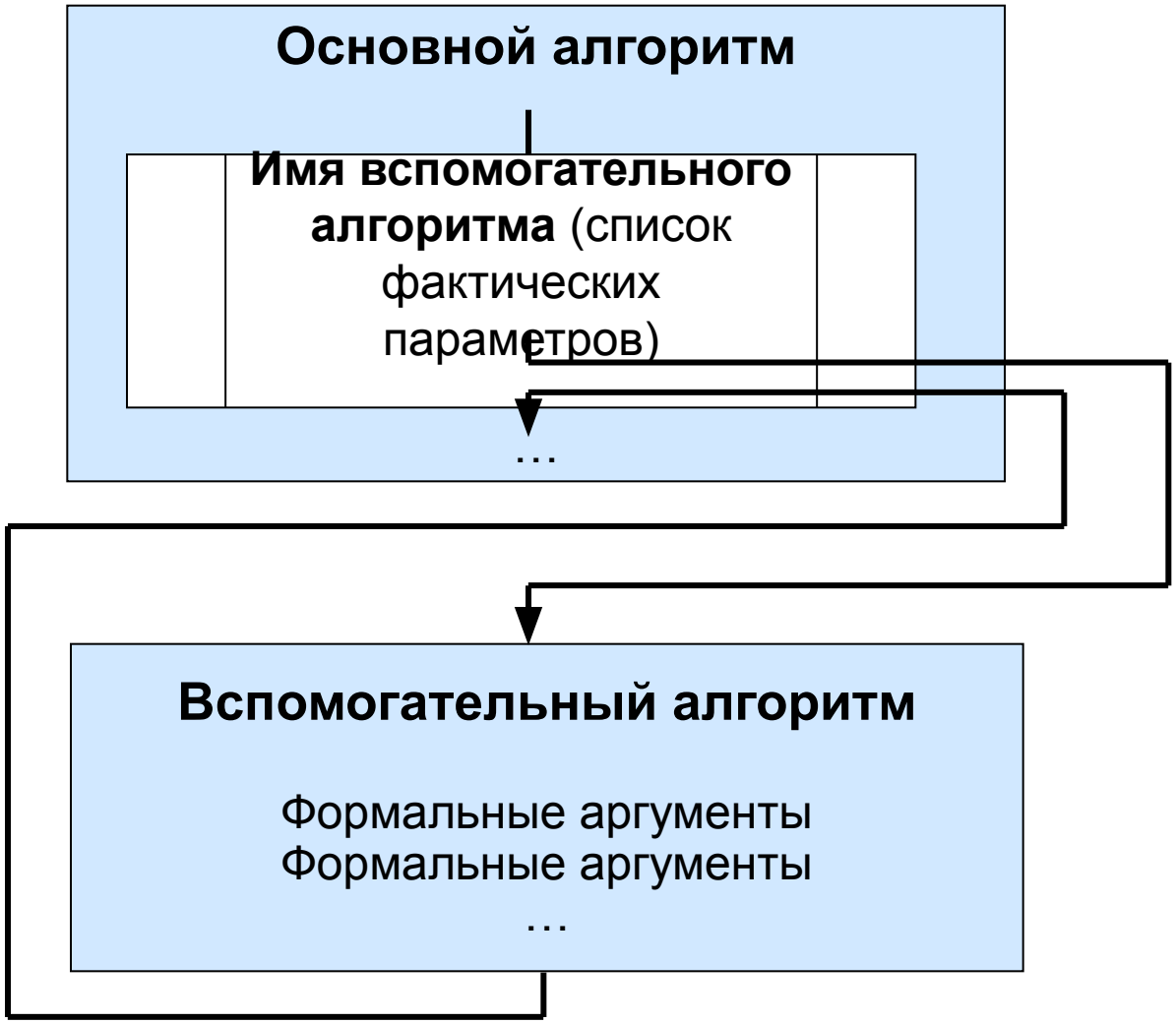
Формальные и фактические параметры

Формальные параметры используются при описании алгоритма.

Фактические параметры **-** те величины, для которых будет исполнен вспомогательный алгоритм.

Типы, количество и порядок следования формальных и фактических параметров должны совпадать.

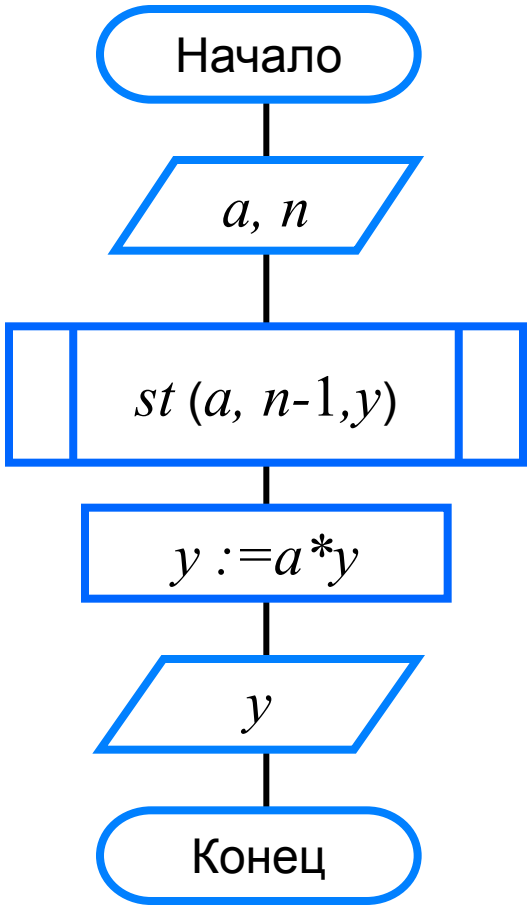
Схема вызова вспомогательного алгоритма



Рекурсивный алгоритм

Алгоритм, в котором прямо или косвенно содержится ссылка на него же как на вспомогательный алгоритм, называют ***рекурсивным***.

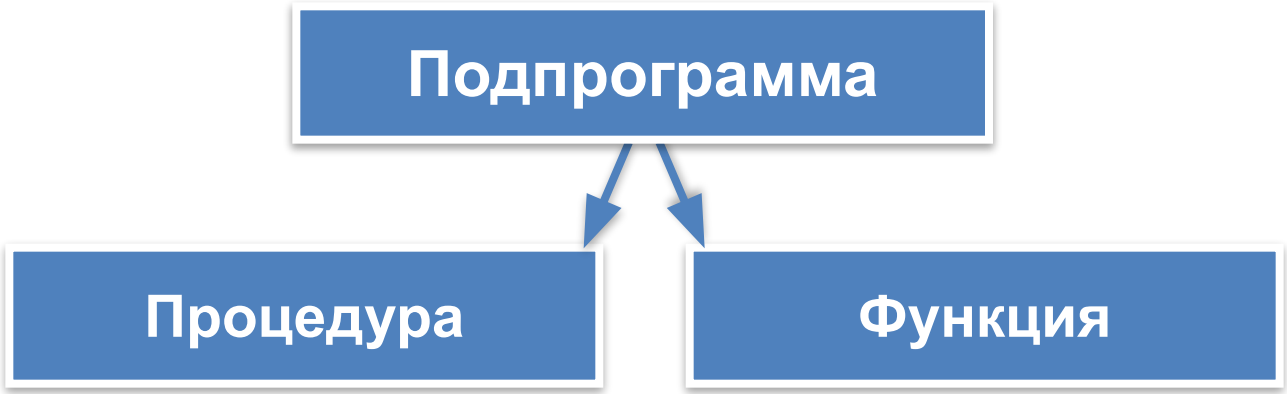
Пример. Алгоритм вычисления степени с натуральным показателем n для любого вещественного числа a, представленный в виде рекурсивного алгоритма



Подпрограммы

Запись вспомогательных алгоритмов в языках программирования осуществляется с помощью ***подпрограмм***.

Структура описания подпрограммы аналогична структуре главной программы. Описание подпрограммы начинается с заголовка и заканчивается оператором **end**



Процедуры

Процедура - подпрограмма, имеющая произвольное количество входных и выходных данных.

Входные параметры: переменные, константы, выражения

procedure <имя_процедуры> (<описание параметров-значений>; **var**: <описание параметров-переменных>);

begin

<операторы>

end;

Выходные параметры

Для вызова процедуры достаточно указать её имя со списком фактических параметров.

Алгоритм Евклида

Процедура для нахождения НОД

```
procedure nod (a, b: integer; var c:integer);
begin
  while a<>b do
    if a>b
        then a:=a-b
        else b:=b-a;
  c:=a
end;
```

Варианты вызова процедуры

nod (36, 15, z)	в качестве параметров-значений использованы константы
nod (x, y, z)	в качестве параметров-значений использованы имена переменных
nod (x+ y, 15, z)	в качестве параметров-значений использованы выражение и константа

! Между фактическими и формальными параметрами должно быть полное соответствие по количеству, порядку следования и типу.

Программа с процедурой

```
program n_6;
const m: array [1..6] of integer =(16, 32, 40, 64, 80, 128);
var I, x, y, z: integer;
procedure nod (a, b: integer; var c: integer);
begin
  while a<>b do
    if a>b then a:=a-b else b:=b-a;
  c:=a
end;
begin
  x:=m[1];
  for i:=2 to 6 do
   begin
    y:=m[i];
    nod (x, y, z);
    x:=z
   end;
writeln ('НОД=', x)
end.
```

Заголовок главной программы

Описание констант

Раздел описания переменных

Раздел описания подпрограммы

Раздел описания операторов главной программы

Функции

Функция - подпрограмма, имеющая единственный результат, записываемый в ячейку памяти, имя которой совпадает с именем функции.

Перечень формальных параметров и их типов

function <имя_функции> (<описание входных данных>): <тип_функции>;
begin
<операторы>;
<имя_функции> := <результат>
end;

Тип результата

! В блоке функции обязательно должен присутствовать оператор **<имя_функции>:=<результат>.**

Для вызова функции достаточно указать её имя со списком фактических параметров в любом выражении, в условиях, (после слов if, while, until) или в операторе **write** главной программы.

Функция поиска максимального из 2-х

```
program n_7;
  var a, b, c, d, f: integer;
function max (x, y: integer): integer;
begin
  if x>y then max:=x else max:=y;
end;
begin
  readln (a, b, c, d);
  f:= max(max(a, b), max(c, d));
  writeln ('f=', f);
end.
```

Заголовок главной программы

Описание переменных

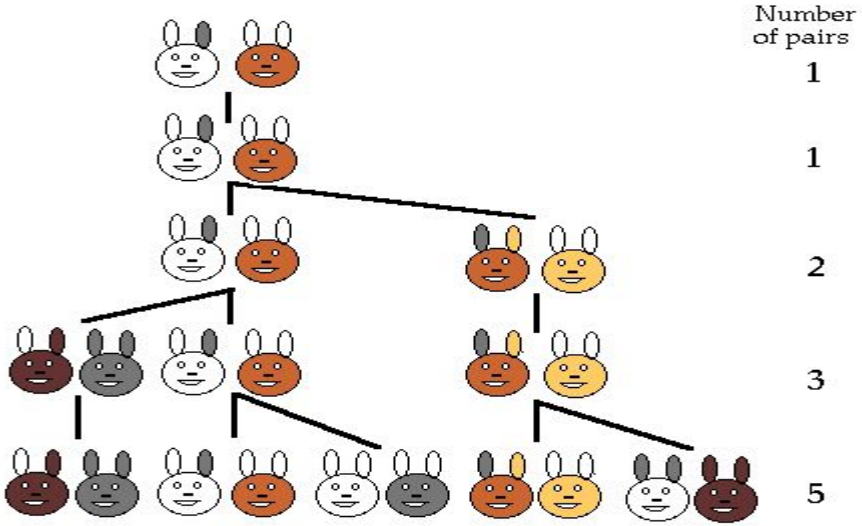
Раздел описания подпрограммы

Раздел операторов главной программы (поиск максимального из 4-х чисел)

Последовательность Фибоначчи

В январе Саше подарили пару новорождённых кроликов. Через два месяца они дали первый приплод - новую пару кроликов, а затем давали приплод по паре кроликов каждый месяц.

Каждая новая пара также даёт первый приплод (пару кроликов) через два месяца, а затем - по паре кроликов каждый месяц. Сколько пар кроликов будет у Саши в декабре?



Числа 1, 1, 2, 3, 5, 8, … образуют так называемую **последовательность Фибоначчи**, названную в честь итальянского математика, впервые решившего соответствующую задачу ещё в начале XIII века.

Математическая модель

Пусть $f(n)$ количество пар кроликов в месяце с номером n.

По условию задачи:

$f(1) = 1$,

$f(2) = 1$,

$f(3) = 2$.

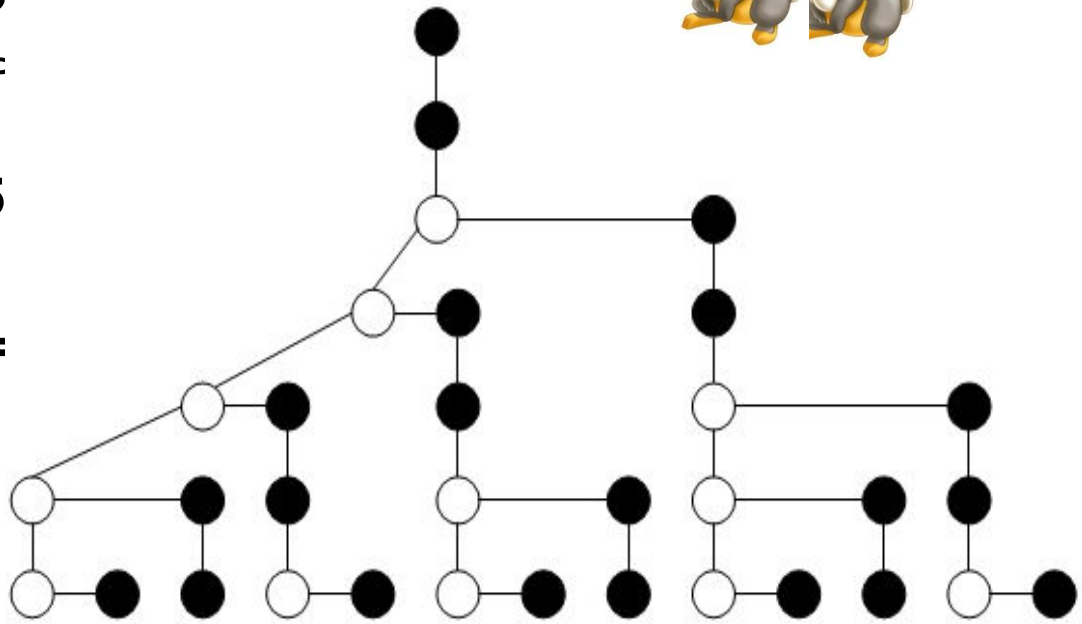
Из двух пар, имеющихся в марте, дать приплод в апреле сможет только одна: $f(4) = 3$.

Из пар, имеющихся в апр… дать приплод в мае смогут толь… родившиеся в марте и ранее:

$f(5) = f(4) + f(3) = 3 + 2 = 5$

В общем случае:

$f(n) = f(n-1) + f(n-2)$, $n >$



Функция

```
function f (n: integer): integer;
begin
  if (n=1) or (n=2) then f:=1
  else f:=f(n-1)+f(n-2)
end;
```

Полученная функция ***рекурсивная*** - в ней реализован способ вычисления очередного значения функции через вычисление её предшествующих значений.

$$\sqrt{3+\sqrt{3+\sqrt{3+\sqrt{3+\sqrt{3+\ldots}}}}}$$

Самое главное

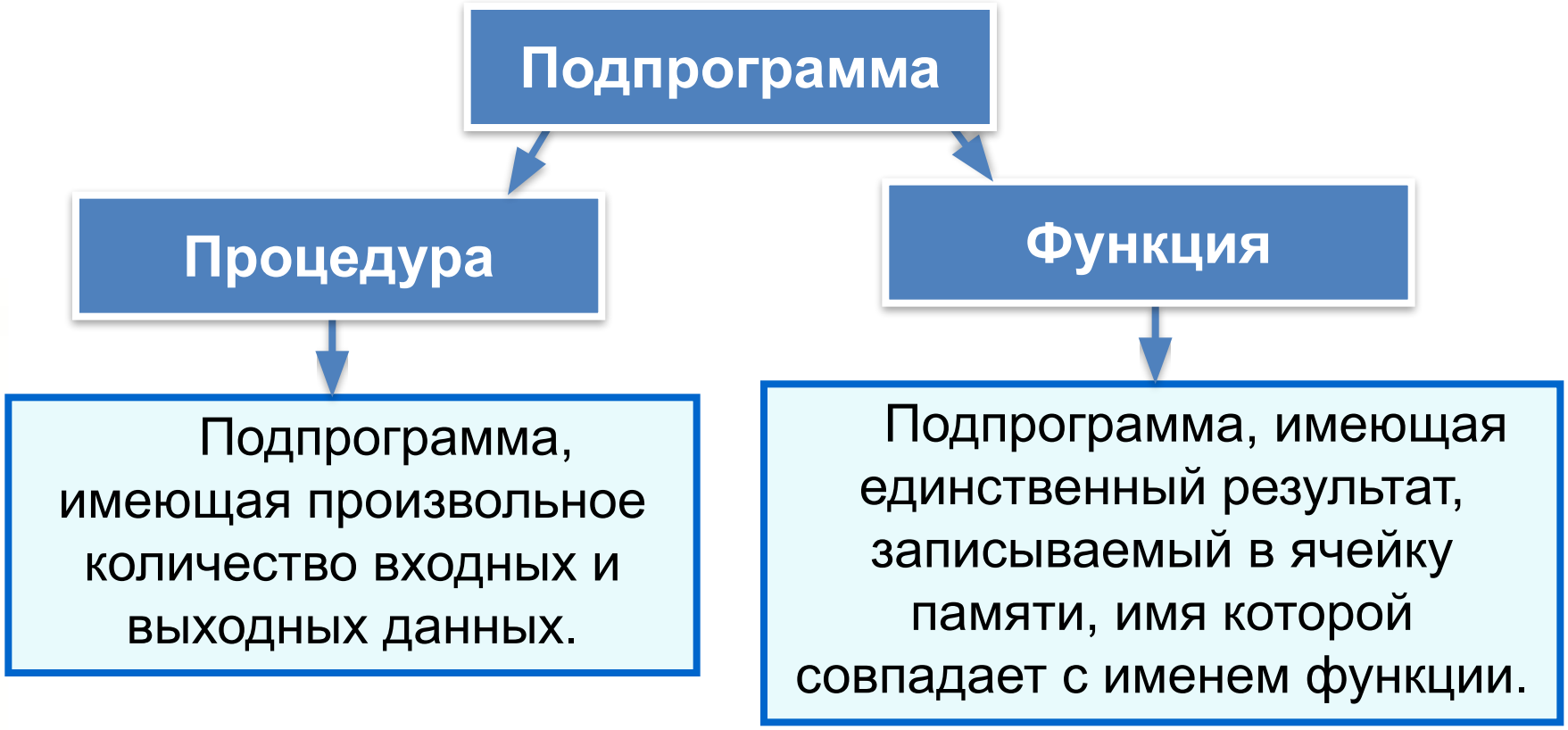
Запись вспомогательных алгоритмов в языках программирования осуществляется с помощью ***подпрограмм***. В Паскале различают два вида подпрограмм: процедуры и функции.

Процедура - подпрограмма, имеющая произвольное количество входных и выходных данных.

Функция - подпрограмма, имеющая единственный результат, записываемый в ячейку памяти, имя которой совпадает с именем функции.

Опорный конспект

Запись вспомогательных алгоритмов в языках программирования осуществляется с помощью ***подпрограмм***.



Работаем в РТ

92. Выделите в программе и подпишите справа от нее заголовок головной программы; раздел описания переменных; раздел описания подпрограммы с указанием имени функции, входных данных, типа результата и операторов функции; раздел операторов головной программы.

```
program n92;
var x, y: integer;
p: longint;
function f (n: integer):
                longint;
var i: integer; p: longint;
begin
  p:=1;
  for i:=1 to x do p:=p*i;
  f:=p
end;
begin
  write ('x='); readln (x);
  write ('y='); readln (y);
  p:= f(x)+f(y)+f(x+y);
  writeln ('p=', p)
end.
```

Что будет выведено на экран в результате выполнения этой программы?

90. Выделите в программе и подпишите справа от нее заголовок головной программы; раздел описания переменных; раздел описания подпрограммы с указанием имени подпрограммы, параметров-значений, параметров переменных и операторов подпрограммы; раздел операторов головной программы.

```
program n90;
var x, y: integer;
procedure tr(a: integer;
        var b: integer);
begin
  writeln (a, '', b);
  a:=a+10;
  b:=b+a;
  writeln (a, '', b);
end;
begin
  x:=5;
  y:=5;
  writeln (x, '', y);
  tr(x, y);
  writeln (x, '', y);
end.
```

Вопросы и задания

Задача 46_1 Напишите программу перестановки значений переменных *a, b, c* в порядке возрастания, т. е. так, чтобы *a* < *b* < *c*. Используйте процедуру **swap**.

```
procedure swap (var x, y: integer);
    var m: integer;
begin
     m:=x;
     x:=y;
     y:=m
end;
```

Исходные данные вводятся с клавиатуры.

Задача 46_2 Напишите функцию, вычисляющую длину отрезка по координатам его концов. С помощью этой функции напишите программу, вычисляющую периметр треугольника по координатам его вершин.