

§ 59

Процедуры

Что такое процедура?

Предположим, что в нескольких местах программы требуется выводить на экран сообщение об ошибке: Ошибка программы. Это можно сделать, например, так:

вывод "Ошибка программы" write('Ошибка программы');

Конечно, можно вставить этот оператор вывода везде, где нужно вывести сообщение об ошибке. Но тут есть две сложности. Во-первых, при этом строка-сообщение будет храниться в памяти много раз. Во-вторых, если мы задумаем поменять текст сообщения, нужно будет искать эти операторы вывода по всей программе. Для таких случаев в языках программирования предусмотрены **процедуры — вспомогательные алгоритмы**, которые выполняют некоторые действия.

```

алг С процедурой
нач
  цел п
  ввод п
  если n<0 то Error все
  ...
кон
алг Error
нач
  вывод 'Ошибка программы'
кон

```

```

program withProc;
var n: integer;
procedure Error;
begin
  writeln('Ошибка программы')
end;
begin
  read(n);
  if n<0 then Error;
  ...
end.

```

Обратим внимание на разницу оформления программы с процедурой в школьном алгоритмическом языке и в Паскале. В школьном алгоритмическом языке процедура оформляется точно так же, как и основной алгоритм, но размещается после основной программы.

В Паскале процедура начинается с ключевого слова **procedure**, тело процедуры начинается с ключевого слова **begin** и заканчивается ключевым словом **end** с точкой с запятой. Процедура располагается после блока объявления переменных, но выше основной программы.

Фактически мы ввели в язык программирования новую команду **Error**, которая была расшифрована прямо в теле программы. Для того чтобы процедура заработала, в основной программе (или в другой процедуре) необходимо ее **вызвать** по имени.

Как мы видели, использование процедур сокращает код, если какие-то операции выполняются несколько раз в разных местах программы. Кроме того, иногда большую программу разбивают на несколько процедур для удобства, оформляя в виде процедур отдельные этапы сложного алгоритма. Такой подход делает всю программу более понятной.

Процедура с параметрами

Процедура **Error** при каждом вызове делает одно и то же. Более интересны процедуры, которым можно передавать **параметры** (аргументы) — дополнительные данные, которые изменяют выполняемые действия.

Предположим, что в программе требуется многократно выводить на экран запись целого числа (0..255) в 8-битном двоичном коде. Старшая цифра в такой записи — это частное от деления числа на 128. Далее возьмём остаток от этого деления и разделим на 64 — получается вторая цифра и т. д. Алгоритм, решающий эту задачу для переменной *n*, можно записать так:

```

k:=128
нц пока k>0
    вывод div(n,k)
    n:= mod(n,k)
    k:= div(k,2)
кц
k:=128;
while k>0 do begin
    write(n div k);
    n:= n mod k;
    k:= k div 2
end;

```

Писать такой цикл каждый раз, когда нужно вывести двоичное число, очень утомительно. Кроме того, легко сделать ошибку или опечатку, которую будет сложно найти. Поэтому лучше оформить этот вспомогательный алгоритм в виде процедуры. Но этой

процедуре нужно передать значение параметра — число для перевода в двоичную систему. Программа получается такой:

```

алг Двоичный код
нач
    printBin(99)
кон

алг printBin(цел n0)
нач
    цел n, k
    n:= n0
    k:= 128
    нц пока k>0
        вывод div(n, k)
        n:= mod(n, k)
        k:= div(k, 2)
    кц
кон

program binCode;
procedure printBin(n: integer);
var k: integer;
begin
    k:= 128;
    while k>0 do begin
        write(n div k);
        n:= n mod k;
        k:= k div 2
    end;
end;
begin
    printBin(99)
end.
```

Основная программа содержит всего одну команду — вызов процедуры `printBin` для значения 99. В заголовке процедуры в скобках записывают тип и внутреннее имя параметра (т. е. имя, по которому к нему можно обращаться в процедуре). Значение параметра, переданное из вызывающей программы в процедуру (в этом примере — число 99), обычно называют **аргументом**.

В процедуре объявлена **локальная** (внутренняя) переменная *k* — она известна только внутри этой процедуры. Обратите внимание, что в школьном алгоритмическом языке локальные переменные объявляются после ключевого слова `нач`, а в языке Паскаль — до слова `begin`.

В школьном алгоритмическом языке запрещено изменять параметры процедуры внутри процедуры, поэтому мы назвали параметр *n0*, а в начале процедуры скопировали его значение в переменную *n*.

Параметров может быть несколько, в этом случае они перечисляются в заголовке процедуры через запятую (в школьном алгоритмическом языке) или точку с запятой (в Паскале). Например, процедуру, которая выводит экран среднее арифметическое двух целых чисел, можно записать так:

```

алг printSred(цел a,
               цел b)
нач
  вывод (a+b)/2
кон

```

```

procedure printSred
(a: integer;
b: integer);
begin
  write((a+b)/2);
end.

```

Если несколько параметров одного типа стоят в списке один за другим, их можно определить списком:

```

алг printSred(цел a, b) procedure printSred(a, b: integer);
нач
  вывод (a+b)/2
кон

```

```

begin
  write((a+b)/2);
end.

```

Изменяемые параметры

Напишем процедуру, которая меняет местами значения двух переменных. Проще всего для этого использовать третью переменную (пока напишем программу только на Паскале):

```

program Exchange;
var x, y: integer;
procedure Swap(a, b: integer);
var c: integer;
begin
  c:=a; a:=b; b:=c
end;

begin
  x:=2; y:=3;
  Swap(x, y);
  write(x, ' ', y)
end.

```

После запуска этой программы обнаружится, что значения переменных x и y остались прежними: на экран будет выведено: 2 3. Дело в том, что эта процедура работает с *копиями* переданных ей параметров. Это значит, что процедура Swap создаёт в памяти временные локальные переменные с именами a и b и копирует в них переданные значения переменных x и y основной программы. Поэтому и все перестановки в нашей программе были

сделаны именно с копиями, а значения переменных x и y не изменились. Такая передача параметров называется **передачей по значению**.

Чтобы решить проблему, нужно явно сказать, чтобы процедура работала с теми же ячейками памяти, что и основная программа. Для этого в Паскале в заголовке процедуры перед именем изменяемого параметра пишут ключевое слово **var**:

```
procedure Swap(var a, b: integer);
```

Теперь процедура решает поставленную задачу: на выходе мы увидим: 3 2, что и требовалось. В подобных случаях говорят, что параметры **передаются по ссылке**, а не по значению. Это означает, что фактически в процедуру передаётся адрес переменной и можно изменять значение этой переменной, записывая новые данные по этому адресу.

При вызове процедуре Swap можно передавать только переменные, но не константы (постоянны) и не арифметические выражения. Например, вызовы Swap(2, 3) и Swap(x, y+3) противоречат правилам языка программирования, и программа выполниться не будет.

В школьном алгоритмическом языке все параметры делятся на **аргументы** (исходные данные, обозначаются **arg**) и **результаты** (ключевое слово **рез**, эти значения процедура передаёт вызывающей программе). По умолчанию (если не указано иначе) все параметры считаются аргументами. Поэтому два следующих заголовка равносильны:

```
алг Swap (цел a, b)  
и  
алг Swap (арг цел a, b)
```

В нашем случае параметры процедуры одновременно являются и аргументами, и результатами, поэтому их нужно объявлять с помощью ключевого слова **аргрез**. Приведём полную программу:

```
алг Обмен  
нач  
  цел x=2, y=3  
  Swap(x, y)  
  вывод x, ' ', y  
кон
```

```

алг Swap(арггрез цел a, b)
нач
    цел c
    c:=a; a:=b; b:=c
кон

```

Вопросы и задания

- Что такое процедуры? В чём смысл их использования?
- Как оформляются процедуры в школьном алгоритмическом языке и в Паскале?
- Достаточно ли включить процедуру в текст программы, чтобы она «сработала»?
- Что такое параметры? Зачем они используются?
- Какие переменные называются локальными? Где они объявляются?
- Как оформляются процедуры, имеющие несколько параметров?
- Что такое изменяемые параметры? Зачем они используются?
- Как в заголовке процедуры отличить изменяемый параметр от неизменяемого? Приведите примеры.
- Какие типы параметров выделяются в школьном алгоритмическом языке? Как они объявляются?

Подготовьте сообщение

- «Процедуры в языке Си»
- «Процедуры в языке Python»

Задачи

- Напишите процедуру, которая выводит на экран запись числа, меньшего, чем 8^{10} , в виде 10 знаков в восьмеричной системе счисления.
- Напишите процедуру, которая выводит на экран запись числа, меньшего, чем $16^4 = 65\ 536$, в виде 4 знаков в шестнадцатеричной системе счисления.
- Напишите процедуру, которая принимает параметр — натуральное число N и выводит на экран квадрат из звёздочек со стороной N .
- Напишите процедуру, которая принимает числовой параметр — возраст человека в годах и выводит этот возраст со словом «год», «года» или «лет». Например, 21 год, 22 года, 12 лет.