Информатика

СТРУКТУРНОЕ ПРОГРАММИРОВАНИЕ

ОСНОВНЫЕ СВЕДЕНИЯ ОБ АЛГОРИТМАХ

11 класс

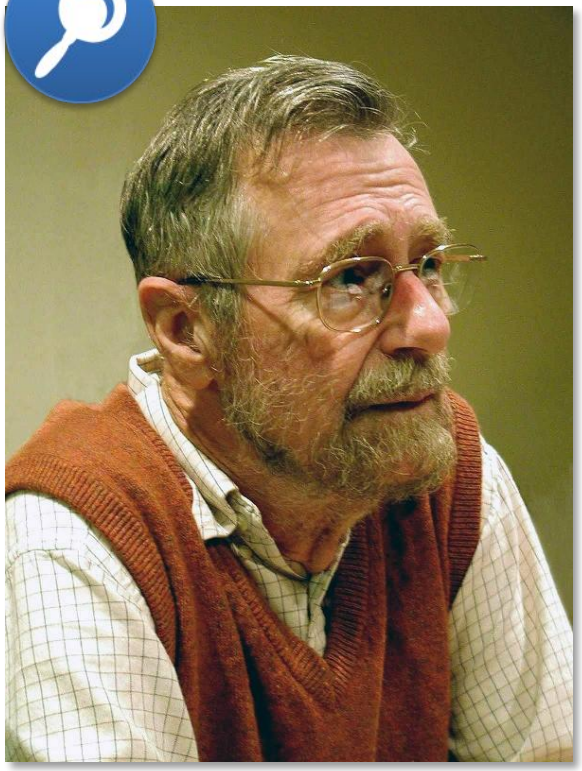
ИЗДАТЕЛЬСТВО БИНОМ

Ключевые слова

- структурное программирование
- вспомогательный алгоритм
- рекурсия
- подпрограммы: процедуры и функции
- фактические и формальные параметры
- параметры-значения и параметры-переменные

Структурное программирование

Структурное программирование – технология разработки программного обеспечения, в основе которой лежит представление программы в виде иерархической структуры логически целостных фрагментов (блоков).

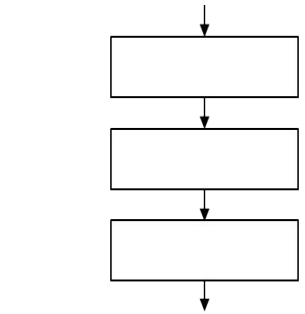


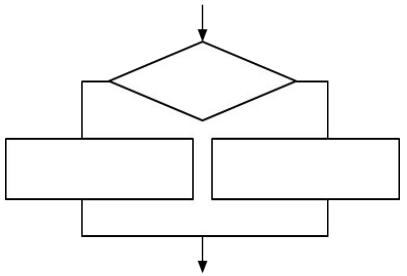
Э́дсгер Ви́бе Де́йкстра (11.05.1930–6.08.2002) – нидерландский учёный, труды которого оказали влияние на развитие информатики и информационных технологий; один из разработчиков концепции структурного программирования, исследователь формальной верификации и распределенных вычислений. Автор нескольких книг и множества статей, самые известные публикации – книги «Дисциплина программирования», «Заметки по структурному программированию», статья

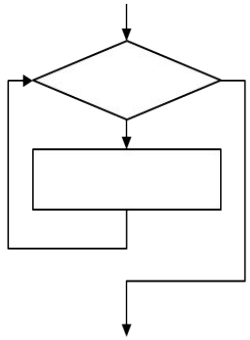
Принципы структурного программирования

Некоторые принципы структурного программирования

1. Любая программа строится из трёх базовых управляющих конструкций: последовательность, ветвление, цикл.
2. В программе базовые управляющие конструкции могут быть вложены друг в друга произвольным образом.
3. Повторяющиеся фрагменты программы можно оформить в виде подпрограмм (процедур и функций). В виде подпрограмм можно оформить логически целостные фрагменты программы, даже если они не повторяются.
4. Все перечисленные конструкции должны иметь один вход и один выход.
5. Разработка программы ведётся пошагово, методом «сверху вниз» (метод последовательной детализации).







Вспомогательный алгоритм

Пример 1. Найти периметр треугольника ABC, заданного координатами своих вершин – (X_A, Y_A), (X_B, Y_B), (X_C, Y_C).

Решение:

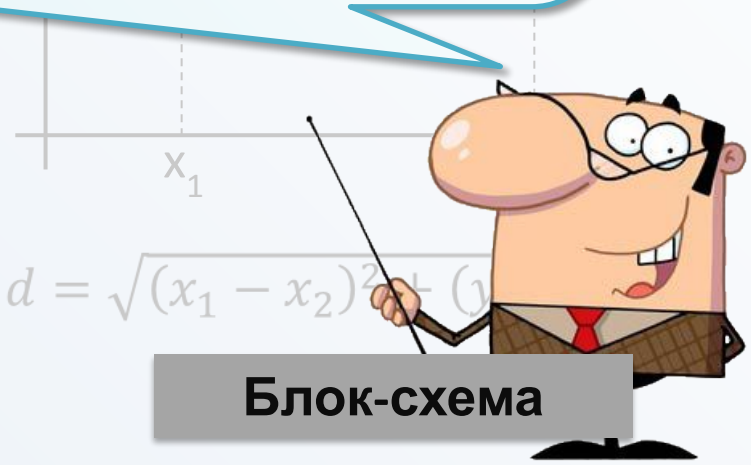
Чтобы найти периметр тр… Для … ко… фо… Де… отр… логически целостный фрагмент, который можно оформить в виде вспомогательного алгоритма.

Вызывая вспомогательный алгоритм с разными исходными данными, вычислим длины всех сторон.

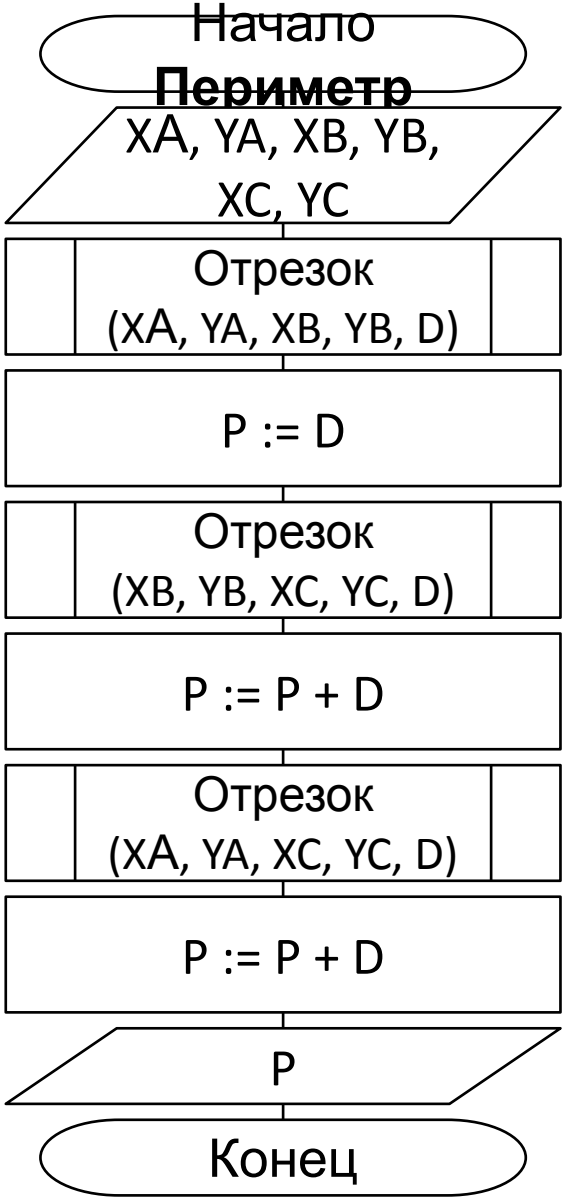
А затем Найдем периметр

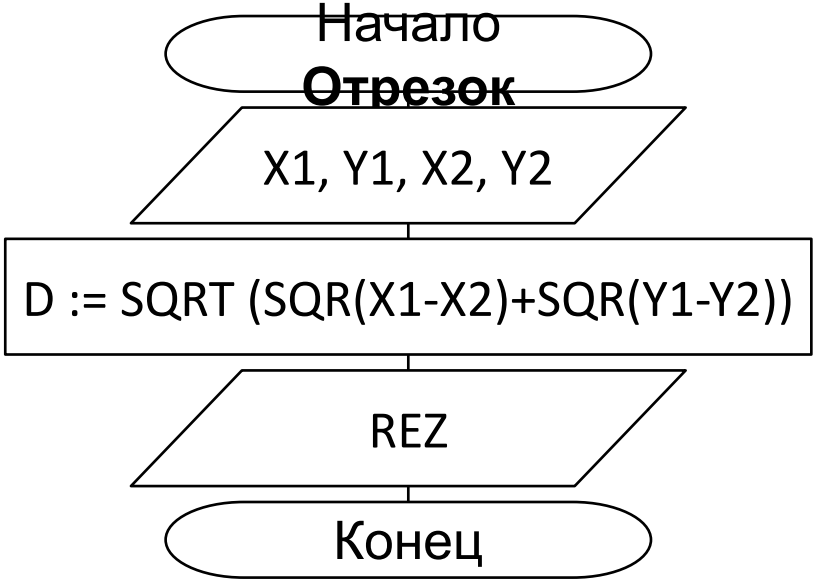
Вспомогательный алгоритм – это алгоритм, целиком используемый в составе другого алгоритма.

При вызове вспомогательного алгоритма указываются его параметры (входные данные и результаты).



Пример программирования сверху вниз





?

Каким будет результат работы алгоритма при следующих исходных данных:

XA = 1, YA = 1,
XB = 1, YB = 5,
XC = 4, YC = 1?

Рекурсивные алгоритмы

Рекурсивные алгоритмы

Алгоритм называется **рекурсивным**, если на каком-либо шаге он прямо или косвенно обращается сам к себе.
В рекурсивном определении должно присутствовать ограничение (граничное условие), при выходе на которое дальнейшая инициация рекурсивных обращений прекращается.

Ночь, улица, фонарь, аптека,
Бессмысленный и тусклый свет.
Живи еще хоть четверть века –
Все будет так. Исхода нет.
Умрешь – начнешь опять сначала
И повторится все, как встарь:
Ночь, ледяная рябь канала,
Аптека, улица, фонарь.

А. Блок

Приведите примеры рекурсии, встречающиеся в жизни, природе или литературных произведениях.

Примеры рекурсивных алгоритмов

Пример 2. Числа Фибоначчи – элементы последовательности 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, … , в которой первые два числа равны 1, а каждое следующее число равно сумме двух предыдущих чисел. Запишите рекуррентное определение чисел Фибоначчи.

Ответ:

$F(n) = 1$ при $n \le 2$;

$F(n) = F(n-1) + F(n-2)$ при $n > 2$.

Пример 3. Запишите рекуррентное определение функции, вычисляющей количество цифр в натуральном числе n.

Ответ:

$K(n) = 1$ при $n < 10$;

$K(n) = K(n \text{ div } 10) + 1$ при $n \ge 10$.

Примеры рекурсивных алгоритмов

Пример 4. Алгоритм вычисления значения функции $F(n)$, где n – натуральное число, задан следующими соотношениями:

$F(1) = 2$;

$F(n) = n \cdot F(n-1)$ при $n > 1$.

Определите значение функции $F(6)$.

Решение:

$F(1) = 2$

$F(2) = 2 \cdot F(1) = 2 \cdot 2 = 4$

$F(3) = 3 \cdot F(2) = 3 \cdot 4 = 12$

$F(4) = 4 \cdot F(3) = 4 \cdot 12 = 48$

$F(5) = 5 \cdot F(4) = 5 \cdot 48 = 240$

$F(6) = 6 \cdot F(5) = 6 \cdot 240 = 1440$

Подобные вычисления можно проводить в уме, а их результаты фиксировать в таблице:

n	1	2	3	4	5	6
$F(n)$	2	4	12	48	240	1440

Ответ: 1440

Подпрограммы в Паскале

Запись вспомогательных алгоритмов в языках программирования осуществляется с помощью подпрограмм.

Подпрограмма – относительно независимая часть программы, оформленная специальным образом и имеющая оригинальное имя, по которому ее можно вызывать в тексте программы.

Процедура

Pascal

Процедура – подпрограмма, имеющая произволь-ное количество входных и выходных данных.

Описание процедуры:

procedure <имя>(<описание параметров-значений>;
var <описание параметров-переменных>);
begin
<операторы>
end;

В заголовке процедуры после её имени приводится перечень *формальных параметров* и их типов.

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

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

Функция

Pascal

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

Описание функции:

function <имя>(<описание параметров>):
<тип_функции>;
begin
  <операторы>
end;

В заголовке функции после её имени приводится описание входных данных – перечень *формальных* параметров и их типов. Там же указывается тип самой функции, т. е. тип результата. В блоке функции должен присутствовать оператор:

<имя_функции> := <результат>;

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

Типы формальных параметров

Формальные параметры

Параметры-значения

- определяют исходные данные, которые нежелательно изменять в ходе выполнения подпрограммы
- играют роль входных параметров
- описываются в заголовке:
  Имя_переменной: тип
- соответствующие им фактические параметры могут быть константами, переменными, выражениями

Параметры-переменные

- используют, если необходимо передать значения переменных в подпрограмму, а затем возвратить их изменившиеся значения в место вызова подпрограммы
- играют роль как входных, так и выходных параметров
- описываются в заголовке:
  Var *Имя_переменной: тип*
- соответствующие им фактические параметры могут быть только

Самое главное

Структурное программирование – технология разработки программного обеспечения, в основе которой лежит представление программы в виде иерархической структуры логически целостных фрагментов (блоков).

Основные принципы структурного программирования:

1) любая программа строится из трёх базовых управляющих конструкций: последовательность, ветвление, цикл;
2) в программе базовые управляющие конструкции могут быть вложены друг в друга произвольным образом;
3) повторяющиеся или логически целостные фрагменты программы можно оформить в виде подпрограмм (процедур и функций)
4) все перечисленные конструкции должны иметь один вход и один выход;
5) разработка программы ведётся пошагово, методом «сверху вниз».

Самое главное

Вспомогательный алгоритм – это алгоритм, целиком используемый в составе другого алгоритма.

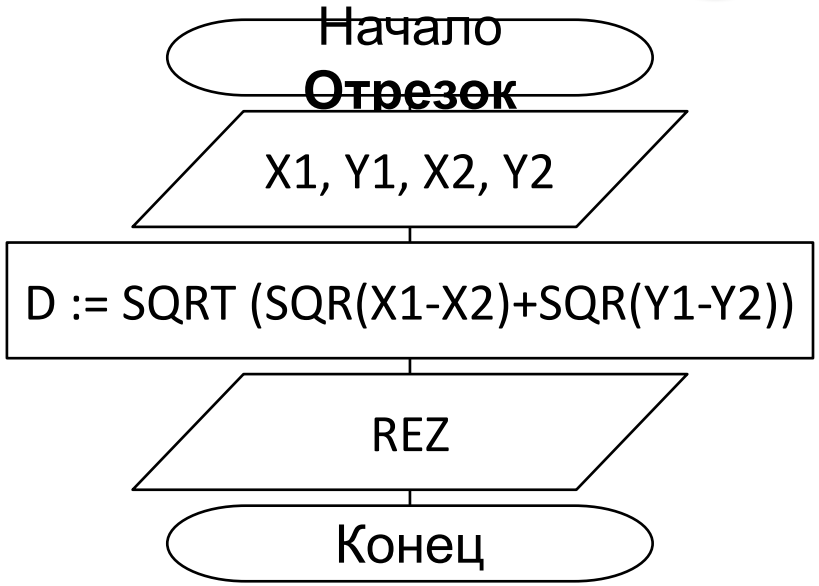
Алгоритм называется **рекурсивным**, если на каком-либо шаге он прямо или косвенно обращается сам к себе.

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

Вопросы и задания

Задание 1. Запишите на языке Pascal подпрограмму нахождения длины отрезка, заданного координатами точек:

1) с помощью функции;
2) с помощью процедуры.



Ответ (с помощью функции):

```
function d (x1,y1, x2,y2: real): real;
begin
    d := sqrt (sqr (x1-x2) + sqr (y1-y2))
end;
```

Ответ (с помощью процедуры):

```
procedure otrezok (x1,y1, x2,y2: real; var d: real);
begin
    d := sqrt (sqr (x1-x2) + sqr (y1-y2))
end;
```

Вопросы и задания

PascalABC

Задание 2. С клавиатуры вводятся n чисел (n<100, запрашивается с клавиатуры). Требуется вывести числа в обратном порядке. Массив использовать

```
var n: integer;
procedure back (n: integer);
   var x: integer;
begin
    if n > 0  then begin
                        read (x);
                        back (n-1);
                        write (x,'  ')
                      end
end;
BEGIN
   write ('Введите n = ');   readln (n);
   back (n)
END.
```

Подсказка | Программа

Вопросы и задания

PascalABC

Задание 3. С клавиатуры вводится натуральное число *X*. Требуется получить число *Y*, в котором записаны цифры числа *X* в обратном порядке. Например, для X=123 Y=321

```
var x, y: integer;
procedure reverse (x: integer; var y: integer);
begin
    if x>0 then  begin
                        y := y*10 + x mod 10;
                        reverse (x div 10, y)
                    end
end;
BEGIN
    write ('Введите число = ');
    readln (x);
    reverse (x, y);
    writeln ('Ответ: ', y)
END.
```

Программа

Вопросы и задания

Задание 4. У исполнителя *Калькулятор* есть две команды:

1. Прибавить 1 – увеличивает число на экране на 1
2. Умножить на 2 – умножает число на экране на 2

Программа для исполнителя – это последовательность команд.

Сколько существует программ, для которых при исходном числе 4 результатом является число 14?

Решение:

Количество программ, с помощью которых можно попасть в некоторое число *n* будем рассматривать как функцию *K* (*n*).

К(n) = 0 при n < 4;

К(n) = 1 при n = 4;

К(n) = К(n – 1) + К(n / 2) при n > 4.

n	4	5	6	7	8	9	10	11	12	13	14
K (*n*)											

Ответ: 5

Информационные источники

- https://commons.wikimedia.org/wiki/File:Edsger_Wybe_Dijkstra.jpg
- https://ru.wikipedia.org/wiki/Дейкстра,_Эдсгер_Вибе
- http://iq230.com/images/sampledata/1/teacher-desk.jpg
- http://cliparts.co/cliparts/Bcg/je7/Bcgje7dc8.png
- http://marshak.su/wp-content/uploads/2015/08/doll_1920_2.png
- http://school15yi.ru/wp-content/uploads/2017/04/3333.jpg