Алгоритмизация и программирование

История

Исторически термин *«алгоритм»* произошел от фамилии арабского (узбекского) математика IX века **аль-Хорезми (полное имя — Абу Абдулла (или Абу Джафар) Мухаммед ибн Муса аль-Хорезми).**

Сведений о жизни учёного сохранилось крайне мало. Имя аль-Хорезми указывает на его родину — среднеазиатское государство Хорезм.

Им было написано первое руководство по арифметике, основанное на позиционном

Алгоритм

- это понятное и точное предписание исполнителю выполнить конечную последовательность команд, приводящих от исходных данных к искомому результату.
- это точно определенная последовательность простых действий, обеспечивающих решение любой задачи из некоторого класса.

Свойства алгоритма

- Дискретность – пошаговый характер выполнения алгоритма. Переход к следующему шагу происходит только тогда, когда закончен предыдущий.
- Точность – команды, которые описывает алгоритм должны быть однозначно интерпретированы исполнителем.

Свойства алгоритма

- Понятность – алгоритм должен быть описан в доступных действиях для исполнителя.
- Конечность – алгоритм должен приводить к какому – либо результату.
- Массовость – алгоритм распределяется на конечную последовательность исходных данных.

Исполнитель алгоритма

- это субъект или устройство, способное правильно интерпретировать описание алгоритма и выполнить содержащийся в нем перечень действий.

Этапы решения задач на ЭВМ

1. Техническое задание (постановка задачи).
2. Формализация (математическая постановка задачи).
3. Выбор (или разработка) метода решения.
4. Разработка алгоритма (алгоритмизация).
5. Выбор языка программирования.
6. Структура данных.
7. Оптимизация.
8. Подготовка отладки.
9. Тесты и методы «ручной» проверки.
10. Запись программы на языке программирования.
11. Тестирование и отладка программы.
12. Вычисление и обработка результатов.
13. Документирование.

Способы записи алгоритма

- Словесно – строчный
- Блок – схема
- Язык программирования

Словесно – строчная запись

- *Пример суммирование двух чисел*

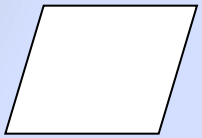
Начало.

1. Ввести A, B.
2. C:=A+B.
3. Печать C.

Конец.

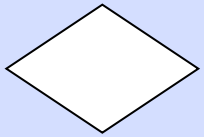
Блок схема – это схема, состоящая из древнегреческих фигур, каждая из которых отвечает за определенное действие алгоритма

Пуск – остановка (начало-конец)



Ввод – вывод данных

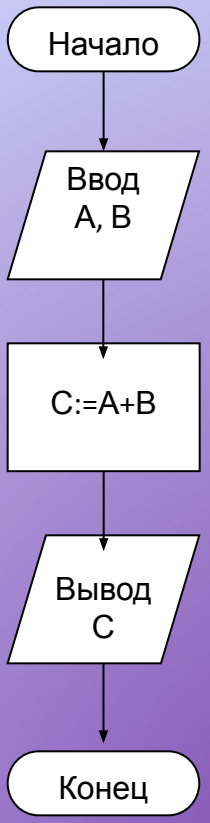
Процесс (Вычислительный блок)



Решение (Логический блок)

Модификация (заголовок цикла)

Пример

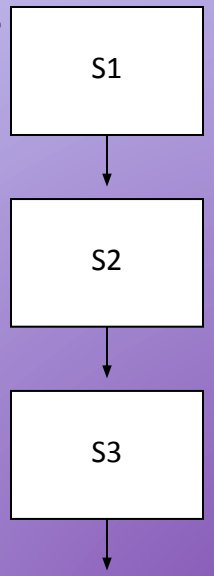


Язык программирования (Паскаль)

```
program massiv;
uses crt;
var
  X : array [1..100] of real;
  i: integer;
  summa: real;
begin
  randomize;
  for i:=1 to 100 do
  begin
    X[i]:=random;
    writeln(X[i]);
    summa:=summa+X[i];
  end;
  writeln;
  writeln(summa);
end.
```

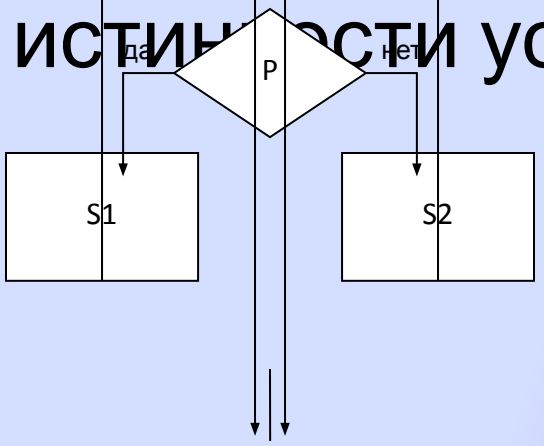
Виды алгоритмов

- Линейный – обеспечивает последовательность действий, при котором команды выполняются в порядке следования.

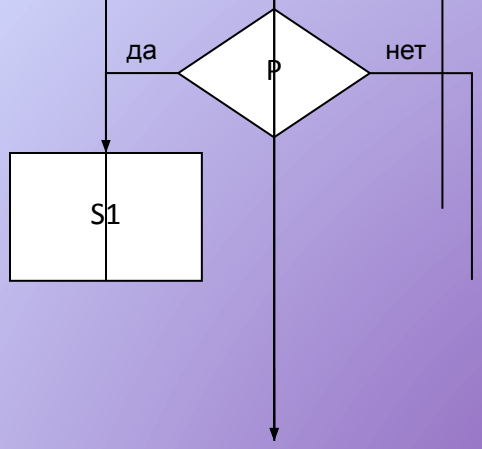


Виды алгоритмов

- Разветвляющийся – обеспечивает выполнение серии команд в зависимости от результата проверки истинности условия.



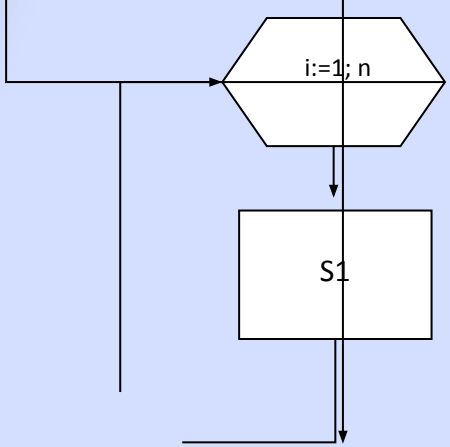
Условие (полное)



Условие (неполное)

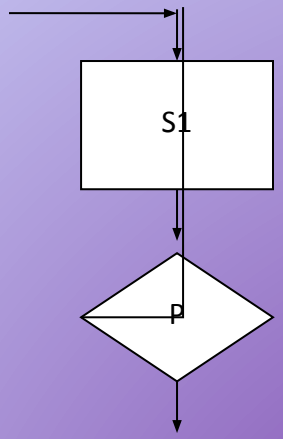
Виды алгоритмов

- Циклический – обеспечивает многократное выполнение некоторой совокупности действий (тела цикла).



Цикл с параметрами (пошаговый)

Цикл с предусловием



Цикл с постусловием

Язык программирования Pascal

В широком смысле язык программирования – это фиксированная система обозначений и правил для алгоритмов и структур данных. Основой для языка программирования Pascal является латинский алфавит, десятичные цифры и правила, используемые для написания программ.

Основные понятия языка программирования

- **Идентификатор** – имя объекта, устанавливающее соответствие объекта некоторому набору символов.

Например: f gh k1 ~~(1k)~~ d_r ~~(d r d,r)~~

- **Константы** – элементы данных, значения которых установлены в описательной части программы и в процессе выполнения программы не изменяются. Описание констант осуществляется в разделе const.

Const k=8; m=19;

Основные понятия языка программирования

- **Переменные** – величины, которые могут менять свои значения в процессе выполнения программы. Описание переменных осуществляется в разделе var.

  Var g: integer;

  st: string;

- **Тип данных** – множество величин, объединенных определенной совокупностью допустимых операций.

Например: 5.9 и -7.456 относятся к вещественному типу (real), их можно складывать, вычитать, умножать, делить и т.д.

Типы данных

- Типы данных делятся на *стандартные* и *пользовательские*.

Стандартные типы предложены разработчиками системы программирования Turbo Pascal, например, *Boolean, real, byte, string, char.*

Типы данных

- **Пользовательские типы данных** разрабатывается пользователями системы, для обеспечения семантического контроля вводимых данных, значительного улучшения наглядности программы, более легкого поиска ошибок и экономии памяти. К пользовательским относят *перечисляемый и интервальный* типы.

Type color=(red, blue, white, black, brown);

month=1..12;

Основные понятия языка программирования

- **Выражение** – конструкция языка, задающая порядок выполнения действий над элементами данных.
- **Операнд** – элемент данных, над которыми производиться операция.
- **Операция** – действия, которые необходимо выполнить над операндом.

Основные понятия языка программирования

- **Оператор** – предложение языка Pascal, задающее полное описание некоторого действия, которое необходимо выполнить.

Операторы в Pascal разделяются точкой с запятой (;).

Оператор присваивания (:=) *A:=78;*

Оператор вызова процедуры *ClrScr;*
{вызов стандартной процедуры очистки экрана}

Операторы ввода-вывода

- *Read* обеспечивает ввод данных в память ЭВМ, при этом курсор остается на месте. *Readln* обеспечивает ввод данных в память ЭВМ, при этом курсор осуществляется перевод курсора на новую строку.
- *Write* предназначена для вывода значений на экран. *Writeln* выводит значения данных на экран и перводит курсор в начало следующей строки.

Решение задач с помощью линейного алгоритма

Вычисление площади треугольника по трем сторонам.

```
Program zadacha (input, output);
var
 a, b, c, p, s: real ;
begin
writeln (‘вв. знач. Сторон треугольника’);
readln (a, b, c);
p:=(a+b+c)/2;
s:=sqrt(p*(p-a)*(p-b)*(p-c));
writeln (‘площадь треугольника=’,s);
end
```

Решение задач с помощью линейного алгоритма

Найти среднее геометрическое и среднее арифметическое двух чисел.

```
Program zadacha (input, output);
Var
a, b, sa, sq: real;
Begin
Writeln (‘введите значение двух чисел’);
Readln (a,b);
Sa:=(a+b)/2;
Sq:=sqrt(a*b);
Writeln (‘среднее арифметическое =’,sa, ‘среднее геометрическое =’,sq)
End.
```

Решение задач с помощью линейного алгоритма

Вычисление гипотенузы и площади треугольника

Паскаль

```
program zadacha (input, output)
Var
a,d,s,g:real;
begin
 writeln ( `введите значения двух катетов `);
readln (a,b);
s:=a*b/2;
g:= sqrt(a*a+b*b)
writeln (‘площадь = ‘,s, ’гипотенуза =’,g);
end.
```

Решение задач с помощью линейного алгоритма

Вычислить длины медиан треугольника, заданного длинами сторон.

```
Program zadacha (input,output);
var
a,b,c,m1,m2,m3:real;
begin
writeln (‘вв.знач. трез сторон треугольника’);
readln (a,b,c);
m1:=sqrt(2*a*a+2*b*b-c*c)/2;
m2:=sqrt(2*a*a+2*c*c-b*b)/2;
m3:=sqrt(2*b*b+2*c*c-a*a)/2
writeln (‘медианы= ’, m1,m2,m3);
end.
```

Решение задач с помощью линейного алгоритма

Треугольник задан величинами своих углов и радиусом описанной окружности. Вычислить длины сторон треугольника.

```
Program zadacha (input, output);
var
a1,b1,c1,r,a,b,c:real;
begin
writeln (‘введите значения a1,b1,c1,r’);
readln (a1,b1,c1,r);
a:=2*r*sin(a1);
b:=2*r*sin(b1);
c:=2*r*sin(c1);
writeln (‘длины сторон= ’, a, b, c);
end.
```

Решение задач с помощью линейного алгоритма

Смешали V1 литров воды с температурой T1 с V2 литрами с t T2. Вычислить t и объем полученной смеси.

```
Program zadacha (input, output);
Var
v1, t1, v2, t2, v, t:real;
begin
Writeln (‘введите значения v1, t1, v2,t2’);
Readln (v1, t1, v2, t2);
v:-v1+v2
t: = (v1*t1+v2*t2)/(v1+v2);
Writeln (‘V смеси = ’, v, ‘T смеси = ’, t);
end.
```

Решение задач с помощью линейного алгоритма

Выделение цифр из заданного числа

```
Program zadacha (input, output);
var
a, a1, c , d: integer;
begin
writeln (‘введите трехзначное число a’);
readln (a);
a1: = a mod 10;
d:=a div 10 mod 10;
c: = a div 100;
writeln (‘число единиц = ’, a1 , ‘ число десятков =’, d);
writeln (‘  число сотен = ‘ , c) ;
end.
```

Решение задач с помощью линейного алгоритма

Вычислить сумму членов арифметической прогрессии, зная её первый член, разность прогрессии и число членов.

```
program zadacha (input, output);
    var
       n: integer;
       a1, d, s:real;
    begin
    writeln (‘введите первый член, разность и число членов ариф. прогрессии’);
    readln (a1,d,n);
    s:=(2*a1 +d*(n-1))/2*n;
    writlen (‘сумма членов арифметической прогрессии’,s)
End.
```

Решение задач с помощью линейного алгоритма

Определить давление на грунт опоры, имеющей форму основания в виде круга радиусом R с вырезанный из центра квадратом со стороной A. Масса опоры M.

```
program zadacha (input, output);
var
m,r,a,d :real;
begin
writeln (‘вв.  массу, R осн., сторону опоры’);
readln  (m,r,a);
d:=9.8*m/(pi*r*r-a*a);
Writeln (‘давление на грунт=’,d);
end.
```

Решение задач с помощью линейного алгоритма

Дана, длина ребра куба. Найти площадь грани, площадь полной поверхности и объем этого куба.

```
Program zadacha (input output);
Var
a, sg, sp, v, real;
begin
Writeln (`введите длину ребра куда);
Readln (a);
Sg: =a*a;
Sp: = 6*sg;
V: = a*a*a;
Writeln ( `S гр.= `, sg, `Sпов.= `, sp,  `V куба = `, v) ;
End.
```

Ветвление

Оператор условия *If <условие> then <оператор - да> else <оператор - нет>;* предназначен для реализации простого ветвления алгоритма.

Оператор выбора *Case <селектор> of*

<значение1>:<оператор1>;

<значение2>:<оператор2>;

……;

<значениеN>:<операторN>

Else <оператор - нет>;

Ветвление

Оператор *Case* используется при множественном разветвлении алгоритма, например, если необходимо по числовому значению месяца указать время года (1, 2, 12 – зима; 3, 4, 5 – весна и т.д.).

```
program sezon;
var n: 1..12;  {перечисляемый тип, здесь допускает только значения от 1 до 12 }
begin
writeln('Введите числовое значение месяца');
 readln(n);
case n of
1, 2, 12: writeln('зима');
3..5: writeln('весна');
6..8: writeln('лето')
else writeln('осень');
end.
```

Решение задач с помощью ветвления

Вычислить y=1/x

```
Program zadacha (input,output);
Var
x,y:real;
Begin
Writeln (‘введите значение переменной x’);
Readln (x);
If x=0 then writeln (‘при x=0 решения нет’)
Else writeln (‘при x= ’,x,’ y=’, 1/x)
End.
```

Решение задач с помощью ветвления

Даны два числа. Выдать на печать наибольшее из них.

```
Program zadacha (input,output);
Var
a,b,s:real;
Begin
Writeln (‘вв. значение двух переменных а, в’);
Readln (a,b);
If a>b then s:= a else s:=b;
writeln (‘большее число = ‘ , s)
end.
```

Решение задач с помощью ветвления

Составить алгоритм выбора наименьшего из трех различных чисел.

```
Program zadacha (input,output);
Var
a, b, c, min :real
Begin
Writeln(`вв.знач. трех перемен a, b, c`)
Readln(a, b, c);
Min:=a;
If d <min then min: =b;
If c< min then min:= c;
Writeln( ` наименование число =`, min)
End.
```

Решение задач с помощью ветвления

Составить алгоритм, который для заданного числа x определяет, принадлежит ли x отрезку [A,B]

```
program zadacha (input, output);
var
a, b, x:real;
begin
writeln (‘  введите значение переменной x‘);
readln (x);
writeln (‘вв. Нач. и кон. Точки отрезка , a, в’);
readln (a,d);
If (x<a) or (x>b)
 then writeln (‘точка не прин. отрезку’)
else writeln (‘точка прин. отрезку’)
end.
```

Решение задач с помощью ветвления

РЕШЕНИЕ КВАДРАТНОГО УРАВНЕНИЯ

```
Program zadacha (input,output);
VAR
A,B,C,D,X1,X2,X:REAL;
BEGIN
Writeln('ВВ. КОЭФ. КВАДР.  УРАВНЕНИЯ a,b,c');
Readln (a,b,c);
D:=b*b-4*a*c;
If d>0 then begin
X1:=(-b+sqrt(d))/(2*a);
X2:=(-b-sqrt(d))/(2*a);
Writeln ('x1=',x1, 'x2=', x2)
End
Else if d=0 then begin x:=(-b)/(2*a);
Writeln ('x=',x)
End
Else Writeln ('уравнение не имеет корней')
End.
```

Решение задач с помощью ветвления

Определение номера четверти плоскости

```
Program zadacha (input, output);
Var
x,y:real;
Begin
Writeln (‘введите координаты точки x,y ‘ );
Readln (x,y);
If (x>0) and (y>0)
Then k:=1
Else if (x<0) and (y>0) then k:=2
Else if (x<0) and (y<0)
Then k:=3
Else k:=4;
Writeln (‘точка находится в ‘ ,k, ‘ четверти ‘)
End.
```

Решение задач с помощью ветвления

Для данного N составить алгоритм вычисления факториала N!=1*2*3*….*N.

```
Program zadacha
Var
F,I,n:integer;
begin
Writeln (‘введите натуральное число N’);
Readln (n);
F:=1;
For i:=1 to n do f:=f*I;
Writeln (n,’!=’,f)
End.
```

Решение задач с помощью ветвления

Среди чисел 1 <= N <= 100. Найти все пары чисел, для которых их сумма равнялась бы их произведению.

```
Program zadacha (input, output);
Var
k, a, b: integer;
begin
k:=0
for a:=1 to 100 do
for b:= 1 to 100 do begin
if a+b=a*b then begin
k:= k+1;
Writeln (‘числа’, a, b)
end;
end
if k=0 then Writeln (‘таких чисел нет’)
end.
```