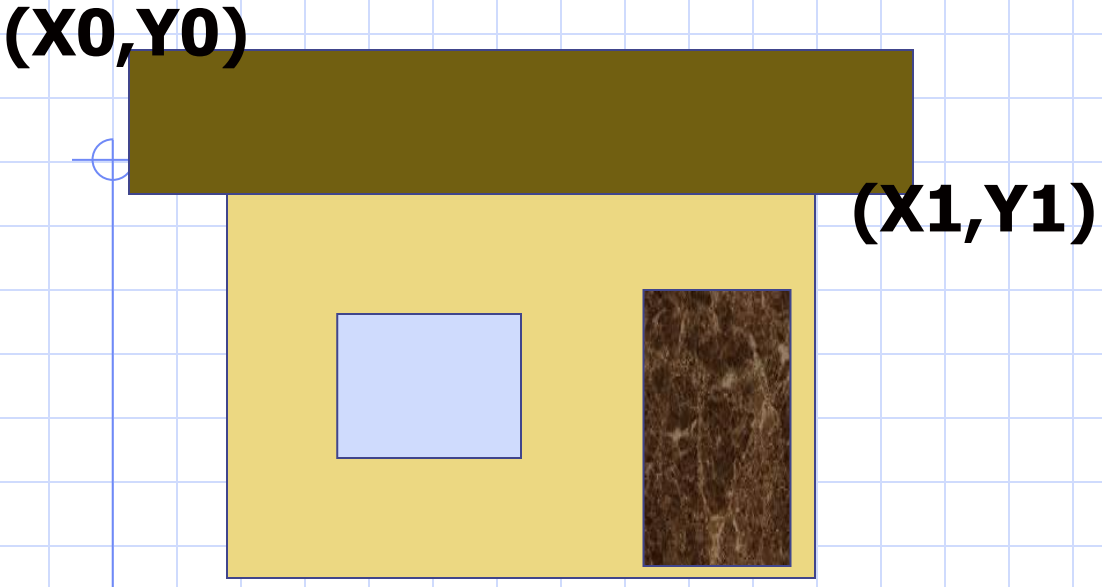
Вспомогательный алгоритм

Процедуры и функции



АЛГОРИТМ:

1 часть рисует дом

2 часть рисует крышу

3 часть рисует окно

4 часть рисует дверь

АЛГОРИТМ

1 часть задать координаты и цвет, выполнить алгоритм «прямоугольник»

2 часть задать координаты и цвет, выполнить алгоритм «прямоугольник»

3 ……

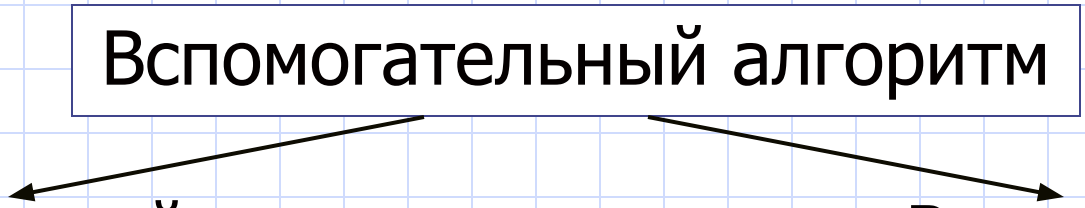
Вспомогательным алгоритмом называется набор действий, выделяемый в качестве самостоятельного алгоритма.

Для его вызова есть специальная команда. Таким образом, вспомогательный алгоритм – это часть программы, оформленная в виде отдельной синтаксической конструкции и снабженная именем. Кроме задания последовательности действий, алгоритм может содержать описание констант, переменных, типов и т.д. Все это предназначено для организации действий только внутри данного вспомогательного алгоритма.

Такая «настройка» алгоритма реализуется с помощью понятия *параметров*

Переменные, которые являются аргументами, необходимыми для работы вспомогательного алгоритма называются параметрами.

При вызове вспомогательного алгоритма важен порядок передачи и приема параметров.



Находится в теле данной программы

Помещается в библиотеку, оформленную соответствующим образом.

Имя библиотеки указывается после служебного слова USES.

Процедуры и функции

```
Procedure имя (параметры);
  var
  объявление переменных;
   BEGIN
    ……….
    ……….
   имя:=выражение;
   END;
```

```
Function имя (параметры):тип данных;
    var
     объявление переменных;
     BEGIN
      ……….
      ……….
END;
```

Основное различие между ними в том, что функция возвращает значение величины и может использоваться в выражениях.

Механизм параметров.

Примеры.

```
Function Smax (a, b: integer, s: char):integer;
Procedure Sum (a,b: real; var c: real);
```

2 способа задачи параметров:

- Параметры, перед которыми отсутствует служебное слово VAR;
- Параметры, перед которыми указано служебное слово VAR.

Параметры-значения.

В данном случае параметр считается обычной переменной, которая действует только в пределах данного вспомогательного алгоритма. По окончании алгоритма она буде уничтожена и ее значение будет потеряно. При вызове вспомогательного алгоритма этому параметру устанавливается начальное значение равное значению соответствующего фактического параметра.

Внутри вспомогательного алгоритма возможны произвольные действия с данным формальным параметром, но любые изменения его значения никак не отражаются на значении переменных вне программы.

Все это объясняется тем, что каждая программа работает в специально отведенном участке памяти компьютера. Вспомогательному алгоритму отводится свой участок памяти.

Пример.

```
Program task;
  var   a,b,c:integer;
   procedure sum(a,b:integer);
      var sum :integer;
   begin
       sum:=a+b;
   writeln('Сумма равна',sum);
   end;
BEGIN
 clrscr;
  write('Введи слагаемые');
   read(a,b);
  sum(a,b);
END.
```

В начале работы программы в рабочем поле отведено место для хранения значений для переменных a,b.

Вспомогательный алгоритм Sum1 после вызова будет активизирован на другом участке памяти. Там же будет отведено и место для переменных a,b,sum. В момент вызова переменным a,b из процедуры будет присвоено значение указанное в скобках. Переменные a,b программы не имеют никакого отношения к переменным a,b из программы. Результат работы процедуры Sum мы увидим, а получить значение sum мы не сможем. После окончания работы участок памяти, где работала процедура, будет освобожден с потерей всех данных, созданных этой программой.

Параметры-переменные.

Параметры, передаваемые по ссылке, указываются заданием служебного слова VAR перед их идентификатором в списке параметров. Этот способ используется тогда, когда необходимо вернуть из процедуры некоторое значение.

Для того, чтобы изменение в теле процедуры значение формального параметра приводило к аналогичному изменению соответствующего фактического параметра, необходимо использовать передачу параметров по ссылке.

Пример.

```
Program task1;
   var   a,b,c:integer;
procedure sum1(a,b:integer; var sum:integer );
      var sum :integer;
   begin
       sum:=a+b;
       writeln('Сумма равна',sum);
   end;
BEGIN
 clrscr;
  write('Введи слагаемые');
   read(a,b);
  sum(a,b,c);
END.
```

Переменной c после работы вспомогательного алгоритма будет присвоено значение переменной Sum

Процедуры и функции могут использовать переменные, объявленные в заголовке основной программы. Такие переменные называются глобальными.

Переменные, описанные в заголовке вспомогательного алгоритма называются локальными и работать с ними можно только в пределах этого алгоритма.

Задача. Сократить дробь a/b.

Процедура

```
Program DROB1;
  Uses CRT;
    var   A,B,C:integer;
 Procedure EVCLID (X,Y:integer; var D:integer);
      begin
          while X<>Y do
         if X>Y then X:=X-Y;
              else Y:=Y-X;
       D:=X;
       end;
BEGIN
 clrscr;
 textcolor(3); writeln('Сокращаем дробь');
 write ('Введи числитель A и знаменатель B'); readln(A,B);
  EVCLID(A,B,D);
  A:=A div D;
  B:=B div D;
writeln ('Сокращенная дробь ' , A , ' / ' B);    readln;
END.
```

Функция

```
Program DROB2;
  Uses CRT;
    var   A,B,C:integer;
Fuction EVCLID (X,Y:integer):integer;
      begin
          while X<>Y do
         if X>Y then X:=X-Y;
              else Y:=Y-X;
       EVCLID:=X;
       end;
BEGIN
 clrscr;
 textcolor(3); writeln('Сокращаем дробь');
 write ('Введи числитель А и знаменатель В'); readln(A,B);
  A:=A div EVCLID (A,B);
  B:=B div EVCLID (A,B);
writeln ('Сокращенная дробь ' , A , ' / ' B);    readln;
END.
```

Домашнее задание:

Учить конспект, составить программы с использованием функции и процедуры для нахождения НОД 3 чисел.

Желаем удачи!