Практическая работа № 6 Часть 1

Задание: Вывести на экран сообщение «Hello World!»

Эта задача включает в себя лишь демонстрацию использования оператора вывода `write` (или `writeln`), который будет единственным в теле нашей маленькой программы.

Напомню, что при использовании функции `write` курсор останется в той же строке, в которой осуществлялся вывод.

Откройте редактор `Pascal` (онлайн или `Pascal ABC`). Напишите программу вывода строки на экран и запустите ее.

Решение задачи:

```
program HelloWorld;
begin
  write('Hello World!');
end.
```

Этот пример не нужно отправлять преподавателю.

Задание: Объявление и инициализация переменных

Рассмотрим объявление переменной на простых примерах.

Объявим новую переменную целого типа `integer` с именем `first` после служебного слова `var`. Далее, между операторными скобками `begin .. end` присвоим ей любое числовое значение.

В разделе описания `var` создадите вторую переменную целого типа с именем `second`. Эта переменная равна переменной `first`, увеличенной в 3 раза.

У вас должен получиться следующий код:

```
program MyProgram;
var
  first : integer;
  second : integer;
begin
  first := 3;
  second := first * 3;
end.
```

Обратите внимание, что при объявлении переменной обязательно указывают тип после двоеточия, а при инициализации - только значение переменной.

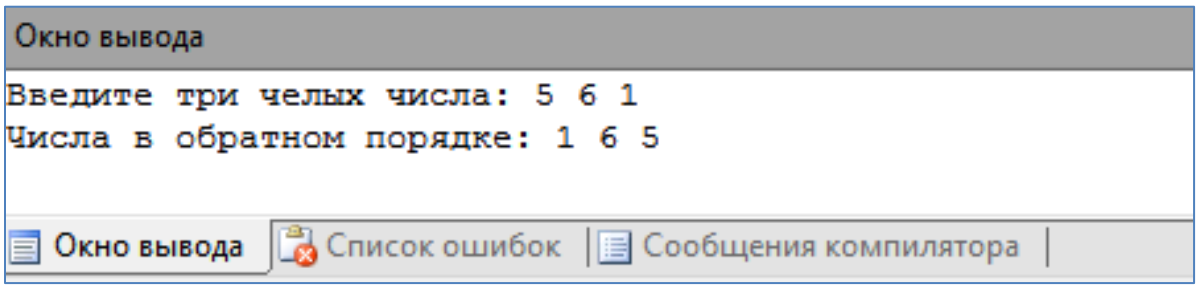
Выполнить самостоятельно: добавьте переменную `third`, которая равна сумме двух первых переменных.

Думаю, что в объявлении и присваивании значения (инициализации) переменных ничего сложного нет.

Сохраните решение задания.

Задание: Вывести на экран три введенных с клавиатуры числа в порядке, обратном их вводу

Ваша задача написать простое приложение, которое спрашивает у пользователя три целых числа и выводит их на экран в обратном порядке. Пример:



Для решения этой задачи откройте редактор. Придумать название программы вам нужно самостоятельно, но с учетом того, что эта программа будет делать, например можно назвать ее `revers` – обратный.

```
program revers;
```

Для решения задачи потребуется две функции:

- первая будет выводить информацию - `write(P1, P2,…, Pn);`

- вторая будет считывать значения в переменные - `read(P1, P2,…, Pn);`

Алгоритм решения задачи будет следующим:

1) объявить три переменных (пусть они будут целого типа - `integer`);

2) вывести пользователю подсказку, чтобы он ввел три целых числа - функция `write();`

3) считать введенные значения с помощью функции `read();`

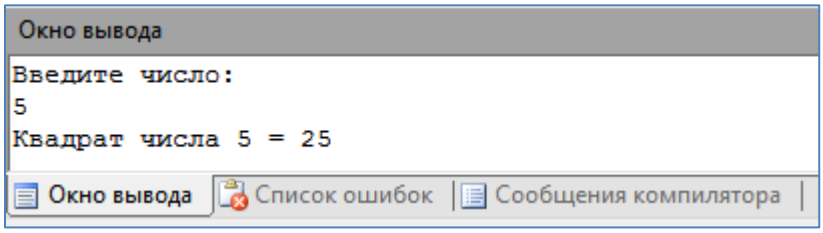
4) и, наконец, вывести числа в обратном порядке: `c, b, a.`

Сохраните решение задания «revers».

Задание: Вычислить квадрат введенного числа

Выражения - это действия, выполняемые программой над определенными в ней объектами и дающее единственное значение на выходе. Любое выражение может состоять из: переменных, констант и операций.

Ваша задача написать простое приложение, которое спрашивает у пользователя целое число и выводит квадрат этого числа. Пример:



Решение задачи:

Введем две переменных: A - введенное число и B - квадрат числа A.

```
var
 A, B : integer;
begin
 writeln('Введите число: ');
 read(A);

 B := A * A;

 writeln('Квадрат числа ', A, ' = ', B);
end.
```

Можно обойтись только переменной A, тогда вывод квадрата будет выглядеть вот так:

```
writeln('Квадрат числа ', A, ' = ', A * A);
```

Этот пример не нужно отправлять преподавателю.

Задание: Вычислить выражение

Новую программу назовите `expression`.

Вычислить простое выражение:

$$\frac{x+y}{x-\frac{1}{2}} - \frac{x-z}{xy}$$

Выражения записываются в одну строку. Не забывайте про скобки, обозначающие действия выражения.

Для вычисления выражения нам нужны три переменные: `x, y, z`. Пусть они будут вещественными (`real`). А также нужна переменная, которая получит результат выражения. Назовем ее `res`.

Вам необходимо:

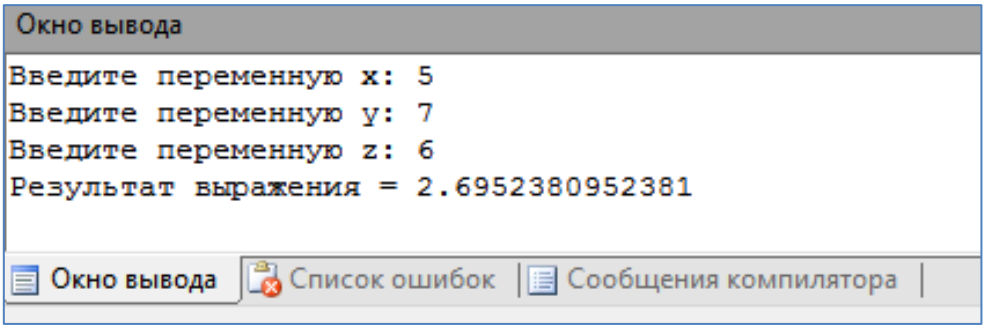
1) запросить у пользователя ввести 3 переменные (либо в одном запросе, либо сделать их 3) - функция `write`.

2) считать все введенные значения в переменные (функция read)

3) вычислить выражение и результат записать в переменную `res`;

4) вывести результат вычисления пользователю на экран.

На рисунке приведен пример выполнения программы:



В этом выражении были использованы самые простые арифметические операции, известные еще со школы.

Сохраните решение задачи.

Задание: Получить реверсную запись трехзначного числа

Рассмотрим операции `div` и `mod` - целочисленное деление и остаток от деления.

Давайте разберемся с заданием. В нашем случае пользователь с помощью клавиатуры вводит некоторое трехзначное число. Нам необходимо получить в некоторой переменной число, которое будет представлять собой реверсную запись введенного числа. Другими словами, нам нужно перевернуть введенное число

«задом наперед», представить результат в некоторой переменной и вывести его на экран.

Алгоритм решение задачи:

1) Объявить переменную n, которая будет хранить введенное пользователем значение.

```
var
 N : integer;
```

2) Так как у нас трехзначное число, значит 3 разряда (сотые, десятые и единицы). Объявите переменные A, B и C для хранения значения каждого разряда.

3) Теперь необходимо разделить число. Это можно сделать с помощью функций div и mod.

Переменная a будет хранить разряд единиц, т.е. если N = 542, то A = 2. Как известно, последний разряд любого числа - это остаток от деления этого числа на 10. На языке Pascal это означает, что нам нужно переменной A присвоить результат операции `N mod 10`.

```
A := N mod 10;
```

4) Очевидно, что при выполнении операции: `N div 100`, мы можем получить разряд сотен и записать его в переменную C.

```
C := N div 100;
```

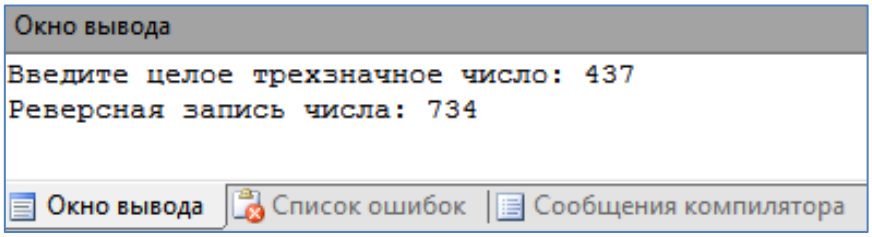
5) Выша задача подумать, как получить разряд десятых.

6) Когда переменные A, B и C имеют значения разрядов введенного числа, создаем переменную R - реверсная запись числа. Не пытайтесь обмануть пользователя простым последовательным выводом переменных на экран. Вы должны «создать» число из трех простых чисел.

Вывод числа на экран будет следующим:

```
writeln('Реверсная запись числа: ', R);
```

Пример работы программы:



Сохраните решение задачи.

В итоге: у вас должно было получиться 4 простых программы. Их необходимо отправить преподавателю (можно скопировать все программы и вставить их в один файл последовательно).

Практическая работа № 6. Часть 2

Условный оператор if

Задание: Написать программу вычисления квадратного корня уравнения.

Информация для тех, кто не помнит, как решать квадратные уравнения:

Условие	D > 0	D = 0	D < 0
Число корней	два корня	один корень	корней нет
Формула	$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$	$x_1 = x_2 = -\frac{b}{2a}$	

Для написания программы порядок действий (алгоритм) должен быть следующий:

1. Спросить у пользователя параметры уравнения: A, B, C;
2. Вычислить дискриминант по формуле D = B * B – 4 * A * C;
3. В зависимости от значения дискриминанта вычислить корни или вывести «решения нет».

Для начала, необходимо объявить шесть переменных: a, b, c, d, x1 и x2 что делается следующим образом:

```
var
  a, b, c : real; // параметры уравнения
  d : real; // дискриминант
  x1, x2 : real; // корни уравнения
begin
end.
```

Переменные можно было объявить в одну строку. В данном примере переменные объявлены так только для того, что бы написать комментарии.

Далее необходимо ввести значения с клавиатуры параметров уравнения. Это делается с помощью функций write и read. ***Самостоятельно!*** Пример работы программы:

```
параметр a: 1
параметр b: 2
параметр c: 1
```

Следующее действие - это вывод квадратного уравнения с подставленными параметрами пользователю на экран:

```
writeln('y = ', a, 'x^2 + ', b, 'x + ', c);
```

После этого, необходимо вычислить дискриминант по формуле, представленной в начале документа. ***Самостоятельно!***

В зависимости от дискриминанта вычисляются корни уравнения:

1) если дискриминант меньше нуля, то выводится соответствующее сообщение и значение дискриминанта

2) иначе, вычисляются корни уравнения и выводятся на экран.

Например, вот так:

```
if d < 0 then begin
  writeln('дискриминант = ', d);
  writeln('корней нет');
end
else begin
  x1 := (-b + sqrt(d)) / (2 * a);
  x2 := (-b - sqrt(d)) / (2 * a);
  writeln('дискриминант = ', d);
  writeln('первый корень = ', x1);
  writeln('второй корень = ', x2);
end;
```

Самостоятельно реализуйте вычисление корня в случае: d = 0 по формуле $x_1 = \frac{-b}{2a}$

Результат работы программы:

```
параметр a: 1
параметр b: 2
параметр c: 1
y = 1*x^2 + 2x + 1
один корень = -1
```

Проверьте работоспособность программы для следующего случая:

```
параметр a: 3
параметр b: 4
параметр c: 1
y = 3*x^2 + 4x + 1
дискриминант = 4
первый корень = -0.333333333333333
второй корень = -1
```

Обратите внимание на вывод чисел на экран. После запятой слишком много символов. Округлим значения до 2 знаков после запятой:

```
writeln('второй корень = ', x2:5:2);
```

Вывод корня уравнения встречается 3 раза. ***Поправьте его вывод во всех.***

Итак, начало программы положено, вспомните, сколько версий решения задачи существует.

Необходимо самостоятельно реализовать:

1) рассмотреть случай превращения квадратного уравнения в линейное (параметр a = 0).

2) выводить сообщение пользователю «Уравнение составлено НЕ правильно!» в случае, когда параметры a и b равны нулю (`a == 0 && b == 0`).

Параметр уравнения задан не правильно (a = 0) - линейное уравнение один корень: $x = -\frac{c}{b}$
Уравнение составлено неверно (a = 0 и b = 0) - ошибка

Оба этих случая поместить после считывания параметров уравнения с клавиатуры и до вывода квадратного уравнения на экран.

У вас должна получиться следующая конструкция:

```
if (a = 0) and (b = 0) then begin
  // в этом месте написать вывод ошибки на экран
  Exit
end;

if a = 0 then begin
  // в этом месте написать формулу вычисления корня
  writeln('линейное уравнение');
  writeln('x = ', x1);
  Exit
end;
```

Оператор **`Exit`** завершает работу программы.

Комментарии переписывать не нужно! На их место нужно писать программный код!

Перед сдачей задания, удостоверьтесь, что программа работает и не возникает ошибок.