Урок 4. Условия. Переменные.

В предыдущих уроках мы выучили ряд **операторов** необходимых для правильной работы наших программ, таких как +, -, \*,/ , begin, end,function, := и других.

В данном уроке мы изучим *операторы* условного перехода **if** и **then** . В перыдущем уроке мы вычисляли корень из дискриминанта. Логично его вычислять только в таком случае, когда дискриминант больше или равен 0. В противном случае надо сразу выдавать сообщение, что уравнение решений не имеем. Переводя на компьютерный язык это будет звучать

**if** Discriminant(b,c)>=0 **then**

Между двумя *операторами* *if и then* располагается условие, в данном случае это то, что дискриминант больше нуля. После *оператора then* располагается то, что мы будем делать, если условие выполнено. Если наши действия занимают одну строчку , то их можно писать сразу после *then*  , а если много строчек, то действия надо помещать между *операторами*  begin и end . Если же данное условие не выполняется и при этом надо произвести какие-то действия, то далее ставиться ещё и оператор **else**.

Откройте файл lesson4\_1.pas . Интересующее нас место в программе выглядит так:

**if** Discriminant(b,c)>=0 **then**

**begin**

Writeln('Корень из дискриминанта равен: ',sqrt(Discriminant(b,c)));

**if** Discriminant(b,c)=0 **then** Writeln('Решение уравнения:');

**end**

**else** Writeln('Дискриминант отрицателен! Решений уравнения нет!');

Как видно, мы можем делать много вложенных друг в друга условий, причём как с *оператором* **else** , так и без него. Модернизируйте программу следующим образом – заставьте программу выводить одно решение уравнения, если дискриминант равен 0 , и два если больше 0. Также модернизируйте вашу функцию вычисления корня. Сделайте , чтобы функция была одна. Если на неё передаётся 0 , то она должна выводить один корень, а если что-то другое, то второй корень. Например, функция должна работать при таком вызове.

Writeln('Корни уравнения: ',Solve(sqrt(Discriminant(b,c)),0), ,', ', Solve(sqrt(Discriminant(b,c)),1));

Теперь, после выполнения, задания наша программа выводит всё правильно. Однако в ней есть недостатки. К пример функция Discriminant вычисляется много раз подряд, а могла бы вычисляться один раз. Также в нашей программе образовалось очень большое количество скобок в которых можно запутаться. Чтобы такого избежать, результат выполнения функций можно сохранять в **переменную** и далее использовать её. Переменная отличается от константы тем, что её значение задаётся во время выполнения программы, а не во время её написания.

*Переменные* задаются оператором **var** её, который ставиться после объявления констант. Присвоение значение переменной осуществляется так же как и присвоение значение результату функции – через оператор := . Ещё одно отличие от константы заключается в том, что необходимо явно определять тип переменной. Откройте файл lesson4\_2.pas и обратите внимание на секцию

**var** //Здесь объявляются переменные

Discriminant\_result : Integer;

Sqrt\_Discriminant : Double;

Здесь переменные были объявлены. А далее идёт присвоение им значений.

Discriminant\_result := Discriminant(b1,c1);

Для завершения урока создайте переменные Solve\_result1 и Solve\_result2 . Попробуйте с их помощью упростить программу, сделанную в первой части урока. Дополните программу так, чтобы она одновременно вычисляла уравнения с константами b1, с1 и с константами b2,с2.