Класс 8 А

### Тема: "Программирование разветвляющихся алгоритмов. Простой и составной условные операторы"

**Ссылка на видео урок для дополнительного просмотра** [**https://youtu.be/iMp\_f517VPE**](https://youtu.be/iMp_f517VPE)

**Вопросы рассматриваемые на уроке:**

·     Определение разветвляющегося алгоритма.

·     Операторы, используемые для записи разветвляющихся алгоритмов в языке Pascal.

·     Простой и составной условные операторы.

Далеко не все алгоритмы можно свести к простой последовательности действий. Иногда действия изменяются в зависимости от некоторых условий. Например, если будет солнечная погода – Миша пойдёт играть в футбол, в противном случае – он останется дома и будет придумывать полезные программы, которые он мог бы написать.

Для описания подобных ситуаций используются **разветвляющиеся алгоритмы**. Они несколько сложнее линейных. **В разветвляющихся алгоритмах** помимо следования используется конструкция ветвления. **Ветвлением** называется конструкция, в которой в зависимости от некоторого условия происходит исполнение одной из двух последовательностей команд, которые называются **ветвями**.

Так как разветвляющиеся алгоритмы сложнее линейных, для того чтобы облегчить их составление при решении задач, прежде чем приступить к программированию алгоритмов будем составлять их блок-схемы. При записи блок-схемы такого алгоритма видно, что после условия – блок которого изображается в виде ромба – следует две ветви операторов. Одна из них будет выполняться в том случае, если будет выполняться условие, в противном случае будет выполняться другая ветвь.

Для записи конструкции ветвления в языке Pascal используется **условный оператор**.

У условного оператора в языке программирования Паскаль есть две **формы записи**: полная и сокращённая.

1. **В сокращённой форме** для записи условного оператора используется служебное слово **if**. После него через пробел следует условие. В роли условия выступает логическое выражение, простое или сложное. После условия на следующей строке записывается служебное слово **then**. После него следует оператор, который будет выполнен, если будет выполняться указанное условие. После оператора следует точка с запятой.
2. **В полной форме**записи условного оператора следует всё тоже самое, однако после оператора, следующего после слова **then**, точка с запятой не ставится. Вместо этого знака на следующей строке записывается служебное слово **else**. После него следует оператор, который будет выполнен, если указанное условие выполнятся не будет. После него следует точка с запятой.

**Задача:** Написать программу для определения наибольшего из двух чисел, введённых пользователем. Если числа равны между собой – вывести любое из них.

Обозначим введённые пользователем числа **a** и **b**. Составим блок-схему алгоритма решения задачи. Как мы помним, любая блок-схема начинается с блока «Начало». Сначала программа будет принимать на ввод два числа: **a** и **b**. Далее будет следовать ветвление. Его условием будет, что **a** больше **b**. В случае, если условие выполняется программа должна вывести **a**, если же условие выполняться не будет, то есть **b** будет больше **a** или же они будут равны между собой, то программа должна вывести **b**. Любая блок-схема заканчивается блоком «Конец».

Напишем программу по составленной блок-схеме. Назовём нашу программу max. В разделе описания переменных укажем две переменные: **a** и **b**. Так как в тексте задачи не указано, что они целые – объявим их вещественного типа **real**. Запишем логические скобки. Сначала запишем оператор **writeln**, который будет выводить на экран поясняющее сообщение о том, что это программа определения наибольшего из двух чисел, и запрос на ввод двух чисел. Дальше будет следовать оператор **readln**, который будет считывать значения переменных **a** и **b**.

Теперь запишем условный оператор. Как мы узнали, он начинается со служебного слова **if**. Через пробел поле него запишем условие **a>b**. Cтрокой ниже запишем служебное слово **then**, после которого будет следовать оператор **write**, который будет выводить на экран значение переменной **a** и поясняющее сообщение, о том, что это наибольшее из двух введённых чисел. После этого оператора точка с запятой ставиться не будет. На следующей строке запишем служебное слово **else**, после которого будет следовать оператор **write**. Он будет выводить на экран значение переменной **b** и поясняющее сообщение о том, что это наибольшее число. После этого оператора поставим точку с запятой.

Пример:

**program**max;

**var**

a, b: real;

**begin**

writeln ('Программа нахождения большего из 2 чисел. Введите 2 числа.');

 readln (a, b);

**if**a>b

**then**write (a, ' - наиболешее из 2 чисел.')

**else**write (b, ' - наибольшее из 2 чисел.');

**end**.

Есть ситуации, когда от выполнения условия в программе зависит исполнение ни одного действия, а целой их серии. В этом случае используется **составной условный оператор**. В этом операторе после служебных слов **then** и **else** следуют целые последовательности операторов, заключённые между логическими скобками. Важно при этом помнить, что перед служебным словом **else** точка с запятой не ставится.

**Домашняя работа: составить блок-схему для следующей задачи в тетради**

**Задача:** 3 отрезка заданы своими длинами. Определить, образуют ли эти отрезки треугольник, и если образуют, то какой: остроугольный, прямоугольный или тупоугольный.

Срок сдачи работы не позднее 23.04.2020