Условные конструкции

В первую очередь вспомним тип данных boolean. Как вы помните, тип boolean может хранить внутри себя либо true (истина), либо false (ложь).

```
boolean b = false;

boolean c = !b;
```

! - означает отрицание. Т.о. в переменной с будет лежать значение true.

Помимо явного указания значения, можно использовать логические выражения. Общи вид логического выражения:

```
boolean d = a *ло* b;
```

где *ло* - логический оператор.

Логические операторы:

- & - логическое и
- | - логическое или
- ! - логическое отрицание
- && - укороченное логическое и
- || - укороченное логическое или

Укороченное логическое «и» работает следующим образом: сначала проверяется первый аргумент, если он равен false, то результат логического «и» в любом случае будет равен false. Поэтому при первом аргументе, равным false, укороченное «и» не проверяет второй аргумент, а выдаёт сразу результат false.

По той же логике работает укороченное логическое «или». Только наоборот: при первом аргументе, равном true, второй аргумент не проверяется и автоматически возвращает true.

Условия могут более сложными:

```
(a>6&&a<10||a==4)
```

Есть несколько базовых операций для написания условных выражений:

- a > b – a больше b
- a < b – a меньше b
- a>=b – a меньше или равно b
- a<=b – a меньше или равно b
- a==b – a равно b

Рассмотрим задачу: с клавиатуру вводится число a, требуется вывести true, если a<5 и false в остальных случаях.

Т.к. boolean имеет встроенное преобразование к строке, то удобно просто положить значение выражения в переменную этого типа.

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      int a = sc.nextInt();
      boolean b = a<5;
      System.out.println(b);
   }
}
```

Обратите внимание, что в первой строке делает импорт Scanner из библиотеки java.utils, т. к. он лежит в другой библиотеке.

Условная конструкция if...else.

Общий вид конструкции if. В скобках можно указывать как логическое выражение любой сложности, так и непосредственно переменную типа boolean. Если условие или переменная равны true, то выполняется всё, что находится внутри первого структурного блока({делать_если_да}), если false, то второго ({делать_если_нет}). В каждом из структурных блоков может быть сколько угодно команд.

```
if(условие){
   делать_если_да
}else{
   делать_если нет
}
```

Попробуем решить задачу: на вход подаётся целое число. Требуется вывести a, если оно меньше, чем 5 и число 5 в остальных случаях.

```
import java.util.Scanner;
public class Main {
  public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      int a =sc.nextInt();
      if (a<5){
            System.out.println(a);
      }else{
         System.out.println(5);
      }
  }
}
```

Заметим, что для такой простой задачи строится довольно громоздкая конструкция. Для такого рода зада создана укороченная конструкция, называемая «тернарная операция» (в переводе с греческого тройная).

Общий вид тернарной операции:

```
условие?если_да:если_нет;
```

Тернарную операцию стоит воспринимать, как особый инструмент, который вместо себя в том месте кода, где он указан, в зависимости подставляет инструкцию «если_да» или «если_нет».

При помощи тернарной операции нашу задачу можно решить двумя способами.

Первый через промежуточную переменную:

```
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a =sc.nextInt();
        int b = a<5?a:0;
        System.out.println(b);
    }
}
```

Второй без промежуточной переменной:

```
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a =sc.nextInt();
        System.out.println(a<5?a:0);
    }
}
```

Оператор множественного выбора.

Представьте, что перед нами стоит задача считать с клавиатуры натуральное число и в случае, если число является цифрой, вывести ей название (например, при a, равном числу 5 надо вывести слово «пять»), и сообщение о том, что число не является цифрой.

При помощи if-конструкции придётся построить очень сложную конструкцию, состоящую из 10 вложенных if. Приведём только общую идею построения такой конструкции без конкретной реализации, т. к. это нерационально.

Если a равно 0, вывести «Ноль», иначе если a равно 1, вывести «Один», иначе…

Это абсолютно нерационально, поэтому в Java реализован оператор множественного выбора switch...case.

Общий вид:

```
switch(имя_переменной){
    case значение1:
      делать_если_переменная_равна_значению1;
   break;
   case значение1:
      делать_если_переменная_равна_значению2;
   break;
   ...
   case значениеN:
      делать_если_переменная_равна_значению3;
   break;
   default:
      делать_если_переменная_не_равна_ни_одному_из_значений;
}
```

Обратите внимание, что структурный блок команд, соответствующие каждому из значений не выделяется фигурными скобками {}. Это скорее исключение из правил. Здесь выполняются все команды, записанные между «case значение:» и «break».

После default break ставить не надо, т. к. конец этого блока совпадает с концом всего блока оператора множественного выбора.

Вернёмся к решению нашей задачи:

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      int a =sc.nextInt();
      switch(a){
         case 0:
            System.out.println("Ноль");
         break;
         case 1:
            System.out.println("Один");
            break;
```

```
            case 2:
                System.out.println("Два");
                break;
            case 3:
                System.out.println("Три");
                break;
            case 4:
                System.out.println("Четыре");
                break;
            case 5:
                System.out.println("Пять");
                break;
            case 6:
                System.out.println("Шесть");
                break;
            case 7:
                System.out.println("Семь");
                break;
            case 8:
                System.out.println("Восемь");
                break;
            case 9:
                System.out.println("Девять");
                break;
            default:
                System.out.println("a не является цифрой");
        }
    }
}
```

А что если для нескольких значений нам надо выполнить одно и тоже действие? Например по номеру месяца определить время года, а если число не является номером месяца вывести сообщение об этом?

Такую задачу можно также решить с помощью оператора множественного выбора.

```java
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        switch(a){
            case 12:
            case 1:
            case 2:
                System.out.println("Зима");
                break;
            case 3:
            case 4:
            case 5:
                System.out.println("Весна");
                break;
            case 6:
            case 7:
            case 8:
                System.out.println("Лето");
                break;
            case 9:
            case 10:
            case 11:
                System.out.println("Осень");
                break;
            default:
                System.out.println("a не является номером месяца");
        }
    }
}
```

Заметим, что для выполнения одного набора действий для нескольких значений, нужно указать все case подряд.

На самом деле, switch case работает следующим образом: он идёт по всем значениям, указанным в case. Если находит совпадение, выполняет все инструкции до первого break, игнорируя на своём пути все попавшиеся case.