ΤΕΧΝΙΚΕΣ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΟΥΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Εισαγωγή στη Java III

Ισότητα Strings

Τι θα τυπώσουν τα παρακάτω?

```
class StringTest{
  public static void main(String args[])
  {
        String x1 = "java";
        String y1 = "java";
        System.out.println("1. " + (x1==y1));

        String x2 = new String("java");
        String y2 = new String("java");
        System.out.println("2. " + (x2==y2));

        System.out.println("3. " + (x1 == "java"));
        System.out.println("4. " + (x2 == "java"));

        System.out.println("5. " + x1.equals(x2));
        System.out.println("6. " + x2.equals("java"));
  }
}
```

1. true
2. false
3. true
4. false
5. true
6. true

Για την σύγκριση Strings **ΠΑΝΤΑ** χρησιμοποιούμε την μέθοδο equals.

String Interning

- Γιατί συμβαίνει αυτό?
  - Ο τελεστής == μεταξύ δύο αντικειμένων εξετάζει αν πρόκειται για την ίδια θέση μνήμης.
  - Το JVM για κάθε string value που εμφανίζεται δημιουργείται ένα αντικείμενο, το οποίο ονομάζεται intern string, και το οποίο κρατάει αυτή την τιμή.
  - Η εντολή `String x1 = "java";` κάνει το `x1` να δείχνει στη θέση που είναι αποθηκευμένη η τιμή `"java"`
  - Γι αυτό (`x1 == x2`) επιστρέφει true.
  - Όλα αυτά θα είναι πιο ξεκάθαρα όταν θα μιλήσουμε για αναφορές.

Wrapper classes

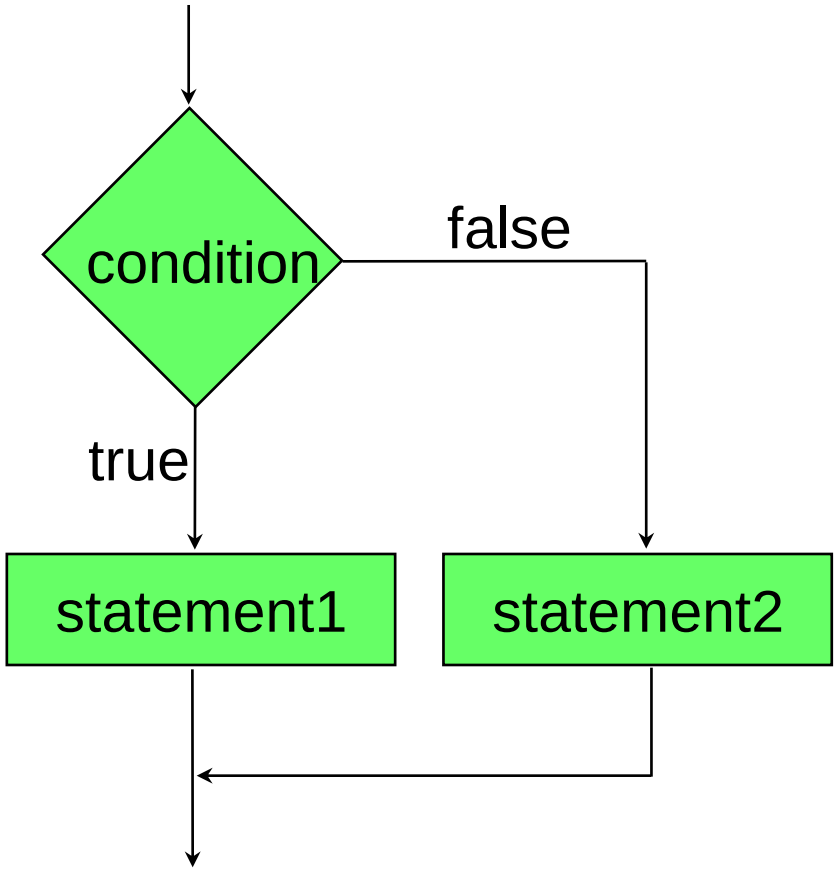
- Για κάθε βασικό τύπο η Java έχει και μία wrapper class:
  - Integer class
  - Double class
  - Boolean class
- Οι κλάσεις αυτές έχουν κάποιες μεθόδους και πεδία που μπορεί να μας είναι χρήσιμα
  - Κατά κύριο λόγο μετατροπή από και προς string
  - Τη μέγιστη και την ελάχιστη τιμή κάθε τύπου

Παράδειγμα

```
class WrapperTest{
  public static void main(String argsp[])
  {
    int i = Integer.valueOf("2");
    double d = Double.parseDouble("2.5");
    System.out.println(i*d);
    Integer x = 5;
    Double y = 2.5;
    String s = x.toString() + " " + y.toString();
    System.out.println(s);
    System.out.println(Integer.MAX_VALUE);
  }
}
```

Το if-else statement

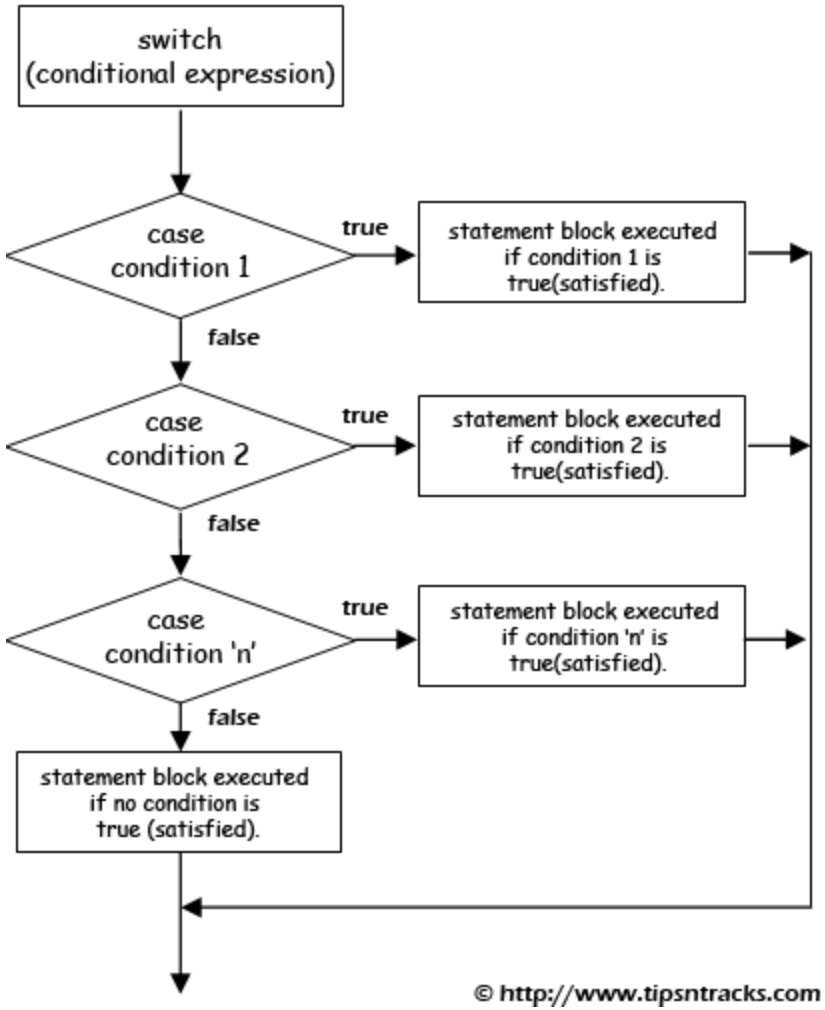
- Το if-else statement δουλεύει καλά όταν στο condition θέλουμε να περιγράψουμε μια επιλογή με δύο πιθανά ενδεχόμενα.
- Τι γίνεται αν η συνθήκη μας έχει πολλά ενδεχόμενα?



Switch statement

Συντακτικό:

```
switch (<condition expression>){
  case <condition 1>:
    code statements 1
    break;
  case <condition 2>:
    code statements 2
    break;
 case <condition 3>:
    code statements 3
    break;
  default:
    default statements
    break;
}
```



Παράδειγμα

- Ένα πρόγραμμα που να εύχεται καλημέρα σε τρεις διαφορετικές γλώσσες ανάλογα με την επιλογή του χρήστη.

```
import java.util.Scanner;

class SwitchTest{
  public static void main(String args[])
  {
    Scanner input = new Scanner(System.in);
    String option = input.next();

    switch(option){
      case "GR":
      case "gr":
        System.out.println("kalimera");
        break;
      case "EN":
      case "en":
        System.out.println("good morning");
        break;
      case "FR":
      case "fr":
        System.out.println("bonjour");
        break;
      default:
        System.out.println("I do not speak this language. " +
                           "Greek, English, French only");
    }
  }
}
```

Πίνακες

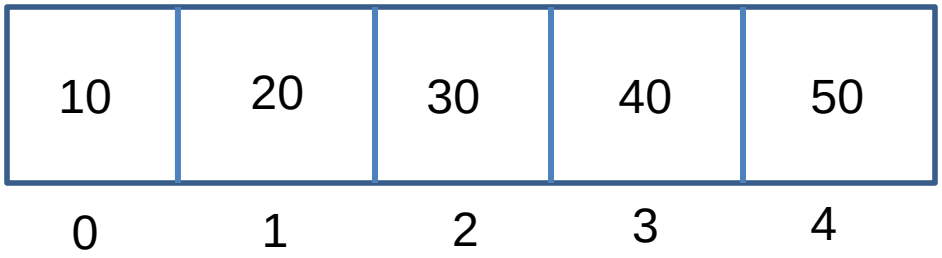
- Πολλές φορές έχουμε πολλές μεταβλητές του ίδιου τύπου που συσχετίζονται και θέλουμε να τις βάλουμε μαζί.
  - Τα ονόματα των φοιτητών σε μία τάξη
  - Οι βαθμοί ενός φοιτητή για όλα τα εργαστήρια.

- Για το σκοπό αυτό χρησιμοποιούμε τους πίνακες.
- Ορισμός πίνακα:
  - int [] myArray1 = {10,20}; // αρχικοποιημένος πίνακας
  - int myArray2[] = new int[2];
  - Δημιουργούν δύο πίνακες 2 θέσεων (length 2) που κρατάνε ακέραιους
- Οι πίνακες ορίζονται με ένα μέγεθος (length) και αυτό δεν αλλάζει

- Στη Java ένας πίνακας είναι ένα αντικείμενο και έχει properties
  - System.out.println(myArray2.length);
  - Τυπώνει το μέγεθος του πίνακα.

Προσβαση των στοιχείων του πίνακα

- Προσοχή! Τα στοιχεία του πίνακα αριθμούνται από το 0…length-1 (ΟΧΙ 1…length)
  - `int myArray[] = {10,20,30,40,50};`



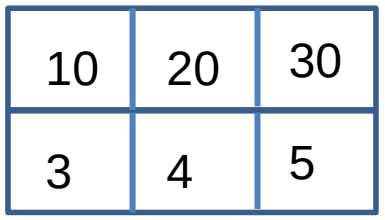
- Για να προσπελαύνουμε το δεύτερο στοιχείο του πίνακα
  - `myArray[1] += 5;`
  - `System.out.prinln(myArray[1]);`

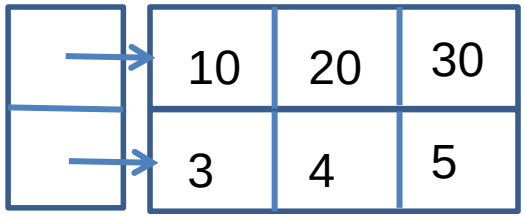
Πίνακες

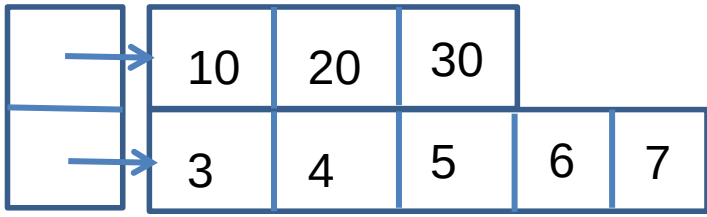
```
public class TestArrays1 {
   public static void main(String [] args){

      int arr0[];  // int[] arr0;

      int arr1 [] = {1, 2, 3, 4};
      for (int i = 0; i < arr1.length; i ++){
            System.out.println(arr1[i]);
      }

      int arr2[] = new int [10];
      for (int i = 0; i < arr2.length; i ++){
            arr2[i] = i+1;
      }
      arr0 = arr2;

   }
}
```

Πολυδιάστατοι πίνακες

- Μπορούμε να ορίσουμε και πολυδιάστατους πίνακες
  - `int myArray1[][] = {{10,20},{2,3}};`
  - `int myArray2[][] = new int[2][3];`
- Ένας δισδιάστατος πίνακας είναι ένας πίνακας από πίνακες.
  - `int myArray3[][] = new int[2][]`
  - `myArray3[0] = new int[3]`
  - `myArray3[1] = new int[3]`
- Ο πίνακας μπορεί να είναι ασύμμετρος
  - `myArray3[1] = new int[5]`
- Τι παίρνω για τα παρακάτω?
  - `System.out.println(myArray3.length);`
  - `System.out.println(myArray3[1].length);`







Πίνακες

```
public class TestArrays2 {
      public static void main(String [] args){
            int arr3[][] = {{1, 2, 3}, {3, 4, 5}};
            int arr4[][] = new int [10][20];
            arr4 = arr3;
            System.out.println(arr3.length + " "
                  + arr3[0].length);
            int arr5[][] = new int[2][];
            arr5[0] = new int[3];
            arr5[1] = new int[5];
      }
}
```

Τυπώνει “2 3”

Ασύμμετρος πίνακας

Παράδειγμα με strings και πίνακες

- Φτιάξτε ένα πρόγραμμα που να διαβάζει γραμμές από κείμενο και να ψάχνει ένα όνομα μέσα στο κείμενο.

```
import java.util.Scanner;

class LookFor
{
    public static void main(String args[])
    {
       String name = "default";
       if (args.length == 1)
       {
              name = args[0];
       }
       Scanner input = new Scanner(System.in);
       String line = input.nextLine();
       String [] words = line.split(" ");
       for (int i =0; i < words.length; i ++)
       {
           if (name.equals(words[i])){
              System.out.println(name + " found it at " + i);
           }
       }
    }
}
```

CLASSES AND OBJECTS

Hello World

- Θα κάνουμε το ίδιο ακριβώς πρόγραμμα αλλά αυτή τη φορά θέλουμε «κάποιος» να πει το hello world.
  - Θέλουμε μια οντότητα που να μπορεί να πει κάτι
- Πως θα το κάνουμε?
  - Θα ορίσουμε μια κλάσση Person.
  - Τα αντικείμενα αυτής της κλάσης θα μπορούν να μιλήσουν

Hello World Revisited

```
class Person
{
        public void speak(String s)
        {
                System.out.println(s);
        }
}

public class HelloWorld2
{
        public static void main(String[] args)
        {
                Person alice = new Person();
                alice.speak("Hello World");
        }
}
```

Ορισμός κλάσης

Ορισμός μεθόδου

Ορισμός αντικειμένου

Κλήση μεθόδου

Πιο σύνθετο παράδειγμα

- Το αντικείμενο που ορίσαμε έχει μεθόδους αλλά όχι χαρακτηριστικά (πεδία)
- Ενας άνθρωπος έχει χαρατηριστικά, π.χ. ένα όνομα.

Hello World re-revisited

```
class Person
{
    private String name;

    public Person(String n){
        name = n;
    }

    public void speak(String s){
        System.out.println(name+":"+s);
    }
}

public class HelloWorld3
{
    public static void main(String[] args){
        Person alice = new Person("Alice");
        alice.speak("Hello World");
    }
}
```

private: Η μεταβλητή name είναι ορατή μόνο από τις μεθόδους τις κλάσης

Constructor: μια μέθοδος με το ίδιο όνομα όπως και η κλάση και χωρίς επιστρεφόμενη τιμή (ούτε void)

Constructor: καλείται όταν δημιουργείται το αντικείμενο και μόνο τότε

Παραδειγμα

- Ένα πρόγραμμα που να παίρνει το ύψος και το βάρος ενός άνθρωπου κα να τυπώνει το λόγο τους.