ΤΕΧΝΙΚΕΣ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΟΥΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Πίνακες

Κλάσεις και Αντικείμενα

Μαθήματα από το πρώτο εργαστήριο

- Έλεγχος ισότητας για Strings:
  - Διαβάζουμε το String `option` και θέλουμε ένα loop να συνεχίσει όσο το option δεν παίρνει την τιμή "EXIT"
  - `while (!option.equals("EXIT"))`

NOT equals

Έλεγχος ισότητας **μόνο** με τη μέθοδο **equals**
Όχι `option == "EXIT"` είναι **ΛΑΘΟΣ!**

- Διάβασμα από την είσοδο:
  - Θελουμε να διαβάσουμε ένα πραγματικό αριθμό ακολουθούμενο από ένα string.

ΣΩΣΤΟ!

```
Scanner in = new Scanner(System.in);
double d = in.nextDouble();
String s = in.next();
```

ΛΑΘΟΣ!

```
Scanner in = new Scanner(System.in);
double d = in.nextDouble();
String s = in.nextLine();
```

Το nextLine() δεν μας κάνει γιατί διαβάζει ότι ακολουθεί τον αριθμό μέχρι να βρει "\n"

Πίνακες

- Πολλές μεταβλητές του ίδιου τύπου μαζί.
  - 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

Διατρέχοντας ένα πίνακα

- Στην Java έχουμε δύο τρόπους να διατρέχουμε ένα πίνακα

Διατρέχουμε τα στοιχεία

```
for (<array type> element: array)
{
   … do something with element…
}
```

```
int array[] = {1,3,5,7};
for (int element: array)
{
  System.out.println(element)
}
```

Διατρέχουμε τις θέσεις του πίνακα

```
for (int i = 0; i < array.length; i ++)
{
   … do something with array[i]…
}
```

```
int array[] = {1,3,5,7};
for (int i = 0; i < array.length; i ++)
{
  System.out.println(array[i])
}
```

Παράδειγμα 1

- Τυπώστε όλα τα στοιχεία του πίνακα και όλα τα ζεύγη από στοιχεία στον πίνακα

```
class TestArray
{
    public static void main(String [] args)
    {
        double [] array = {5.3, 3.4, 2.3, 1.2, 0.1};

        // Print all elements
        for (double element: array){
            System.out.println(element);
        }

        // Print all pairs of elements
        for (int i = 0; i < array.length; i ++){
            for (int j = i+1; j < array.length; j ++){
                System.out.println(array[i] + " " + array[j]);
            }
        }
    }
}
```

Παράδειγμα 2

- Έχουμε ένα πίνακα από ακέραιους και θέλουμε να τους ταξινομήσουμε (sort) από τον μικρότερο προς τον μεγαλύτερο (αύξουσα σειρά).
  - Πολύ χρήσιμη και πολύ συχνή λειτουργία
    - Τι πλεονέκτημα έχει ένας ταξινομημένος πίνακας?
  - Τι αλγόριθμο να χρησιμοποιήσουμε?

- Insertion Sort: Επαναληπτικά, βρες το μικρότερο από τα εναπομείναντα στοιχεία και φέρε το στην αρχή.

```
class InsertionSort
{
   public static void main(String [] args)
   {
       double [] array = {5.3, 3.4, 2.3, 1.2, 0.1};

       for (int i = 0; i < array.length; i ++){
           double min = array[i];
           for (int j = i+1; j < array.length; j ++){
              if (array[j] < min){
                     min = array[j];
                     array[j] = array[i];
                     array[i] = min;
              }
           }
       }

       for (double element: array){
           System.out.print(element + " ");
       }
       System.out.println();
   }
}
```

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 κλάση Person.
  - Τα αντικείμενα αυτής της κλάσης θα μπορούν να μιλήσουν

Hello World Revisited

```
class Person
{
        private String name = "Alice";

        public void sayHello()
        {
                System.out.println(name+": Hello World");
        }
}

class HelloWorld2
{
        public static void main(String[] args)
        {
                Person someone = new Person();
                someone.sayHello();
        }
}
```

Ορισμός κλάσης

Ορισμός (και αρχικοποίηση) πεδίου

Ορισμός μεθόδου

Χρήση πεδίου

Ορισμός αντικειμένου

Κλήση μεθόδου

Public/Private

- Τι σημαίνουν οι δεσμευμένες λέξεις (keywords) public και private?
  - Ότι είναι ορισμένο ως public σε μία κλάση είναι προσβάσιμο από μία άλλη κλάση που ορίζει ένα αντικείμενο τύπου Person
    - Π.χ., η μέθοδος sayHello() είναι προσβάσιμη από την κλάση HelloWorld2 μέσω του αντικειμένου someone.
  - Ότι είναι ορισμένο ως private σε μία κλάση δεν είναι προσβάσιμο από μία άλλη κλάση
    - Π.χ., το πεδίο name δεν είναι προσβάσιμο από την κλάση HelloWorld2 μέσω του αντικειμένου someone.
  - Μπορούμε να έχουμε public και private πεδία και μεθόδους.
    - Κανόνας: Τα πεδία τα ορίζουμε (σχεδόν) **ΠΑΝΤΑ** private.
    - Οι κλάσεις που χρειάζονται να καλούνται από αντικείμενα είναι public αυτές που είναι βοηθητικές είναι private.
- Τα πεδία και οι μέθοδοι μίας κλάσης, ανεξάρτητα αν είναι public ή private, είναι προσβάσιμα από όλες τις μεθόδους της ίδιας κλάσης
  - Π.χ., το πεδίο name είναι προσβάσιμο παντού μέσα στην κλάση Person.

Παράδειγμα

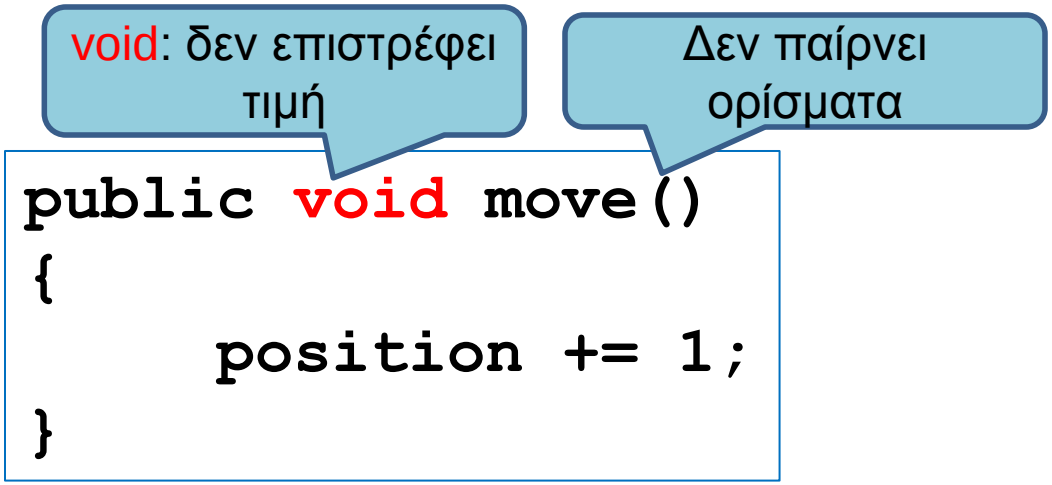
- Θέλουμε ένα πρόγραμμα που να προσομοιώνει την κίνηση ενός αυτοκινήτου, το οποίο κινείται και τυπώνει τη θέση του.

MovingCar

```
class Car
{
    private int position = 0;

    public void move(){
        position += 1;
    }

    public void printPosition(){
        System.out.println("Car at position "+position);
    }
}

class MovingCar
{
    public static void main(String args[]){
        Car myCar = new Car();
        myCar.move();
        myCar.printPosition();
    }
}
```

Μέθοδοι

- Οι μέθοδοι που έχουμε δει μέχρι τώρα είναι πολύ απλές
  - Δεν έχουν παραμέτρους (δεν παίρνουν ορίσματα)
  - Δεν επιστρέφουν τιμή



Παράδειγμα 2

- Εκτός από την κίνηση κατά μία θέση θέλουμε να μπορούμε να κινούμε το όχημα όσες θέσεις θέλουμε είτε προς τα δεξιά (+) είτε προς τα αριστερά (-). Θα τυπώνεται η θέση σε κάθε κίνηση.

```
class Car
{
    private int position = 0;

    public void move(){
            position += 1;
    }

    public void moveManySteps(int steps)
    {
            int delta = 1;
            if (steps < 0){
                steps = -steps; delta = -1;
            }
            for (int i = 0; i < steps; i ++){
                position += delta;
                System.out.println("Car at position "+position);
            }
    }

    public void printPosition(){
            System.out.println("Car at position "+position);
    }
}

class MovingCar2
{
     public static void main(String args[]){
            Car myCar = new Car();
            int steps = -10;
            myCar.moveManySteps(steps);
            System.out.println("--: " + steps);
     }
}
```

Παράμετρος της μεθόδου

Το πέρασμα των παραμέτρων γίνεται κατά τιμή (pass by value)

Η παράμετρος λειτουργεί ως τοπική μεταβλητή της συνάρτησης και χάνεται μετά την κλήση της μεθόδου. Η τιμή του ορίσματος δεν μεταβάλλεται

Όρισμα της μεθόδου

Τυπώνει `--:-10`

```
class Car
{
    private int position = 0;

    public void move(){
            position += 1;
    }

    public void moveManySteps(int steps)
    {
            int delta = 1;
            if (steps < 0){
                steps = -steps; delta = -1;
            }
            for (int i = 0; i < steps; i ++){
                position += delta;
                printPosition();
            }
    }

    public void printPosition(){
            System.out.println("Car at position "+position);
      }
}

class MovingCar2
{
    public static void main(String args[]){
            Car myCar = new Car();
            int steps = -10;
            myCar.moveManySteps(steps);
            System.out.println("--: " + steps);
    }
}
```

Μπορούμε να κάνουμε την εκτύπωση καλώντας την printPosition()