ΤΕΧΝΙΚΕΣ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΟΥΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Υπάρχουσες κλάσεις και αντικείμενα στην Java

Strings

Wrapper Classes

Δομές

ΔΟΜΕΣ

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.
- Συντακτικό:
  - `Τύπος[] myArray;`
  - Ο τύπος είναι οποιοσδήποτε μια οποιαδήποτε κλάση. Είναι ο τύπος των δεδομένων που αποθηκεύει ο πίνακας μας.
- Παραδείγματα:
  - `int[] integerArray;` // πίνακας από ακεραίους
  - `String[] stringArray;` // πίνακας από String

Πίνακες

- Ορισμός πίνακα:

```
int[] myArray1  = {10,20};
int[] myArray2 = new int[2];
int[] myArray3;
```

Ορίζει ένα πίνακα ακεραίων δύο θέσεων με αρχικές τιμές 10,20

Ορίζει ένα πίνακα ακεραίων δύο θέσεων χωρίς αρχικές τιμές (αρχικοποιούνται αυτόματα στο μηδέν)

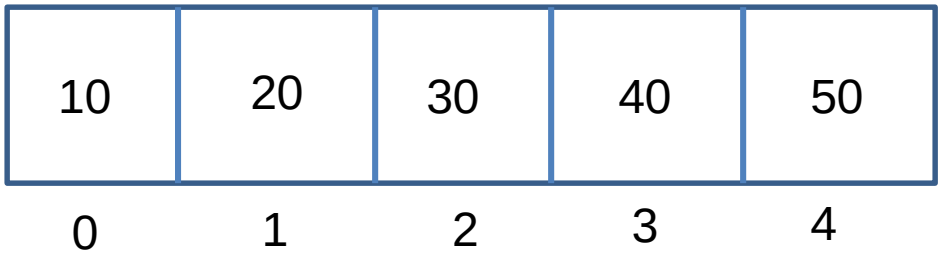
Ορίζει μια μεταβλητή που είναι πίνακας από ακεραίους αλλά δεν δεσμεύει χώρο για τον πίνακα

- Οι πίνακες ορίζονται με ένα μέγεθος (length) και αυτό δεν αλλάζει εκτός αν ορίσουμε ξανά τον πίνακα.

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

Πρόσβαση 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]);`

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])
}
```

```
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];
        arr0 = arr2;
        for (int i = 0; i < arr2.length; i ++){
              arr2[i] = i+1;
        }
        for (int x: arr0){
              System.out.println(x);
        }
    }
}
```

Εναλλακτικό συντακτικό

Ορισμός πίνακα χωρίς χώρο

Ο πίνακας arr0 δείχνει στον ίδιο χώρο μνήμης όπως και ο arr2. Όποια αλλαγή γίνει στον ένα θα εμφανιστεί και στον άλλο

Παράδειγμα

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

```
class ScanArray
{
    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]);
            }
        }
    }
}
```

Σε αυτή την περίπτωση δεν μπορούμε να διατρέξουμε τα στοιχεία, πρέπει να διατρέξουμε τις θέσεις

Πολυδιάστατοι πίνακες

- Μπορούμε να ορίσουμε και πολυδιάστατους πίνακες
  - `int[][] myArray1 = {{10,20,30},{3,4,5}};`

10	20	30
3	4	5

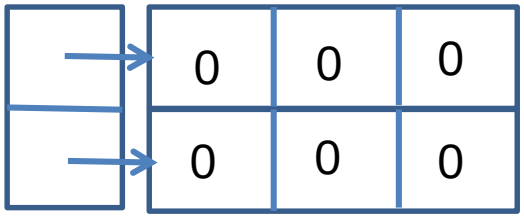
  - `int[][] myArray2 = new int[2][3];`

10	20	30
3	4	5

Πολυδιάστατοι πίνακες

- Ένας δισδιάστατος πίνακας είναι ένας πίνακας από αντικείμενα-πίνακες.
  - `int[][] myArray3;`

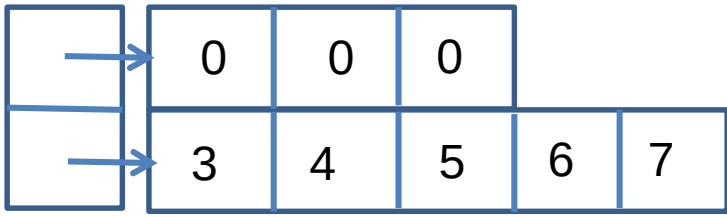
Ο τύπος του πίνακα είναι int[], δηλαδή πίνακας από int



  - `int[][] myArray3 = new int[2][];`
- Η κάθε γραμμή είναι ένας πίνακας και πρέπει να αρχικοποιηθεί
  - `myArray3[0] = new int[3];`
  - `myArray3[1] = new int[3];`

Πολυδιάστατοι πίνακες

- Ένας δισδιάστατος πίνακας είναι ένας πίνακας από αντικείμενα-πίνακες.
  - `int[][] myArray3;`
  - `int[][] myArray3 = new int[2][];`
  - `myArray3[0] = new int[3];`
  - `myArray3[1] = new int[3];`
- Ο πίνακας μπορεί να είναι ασύμμετρος
  - `myArray3[1] = {3,4,5,6,7}`



- Τι παίρνω για τα παρακάτω?
  - `System.out.println(myArray3.length);`
  - `System.out.println(myArray3[1].length);`

```
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(arr4.length + " "
                                        + arr4[0].length);


                int arr5[][] = new int[2][];
                arr5[0] = new int[3];
                arr5[1] = new int[5];
                System.out.println(arr5.length + " "
                                        + arr5[0].length + " "
                                        + arr5[1].length);
        }
}
```

Πίνακας με αρχικές τιμές

Πίνακας 10x20

Ο πίνακας arr4 γίνεται ίδιος με τον arr3. Δηλαδή δείχνει στον ίδιο χώρο μνήμης. Ο προηγούμενος χώρος χάνεται

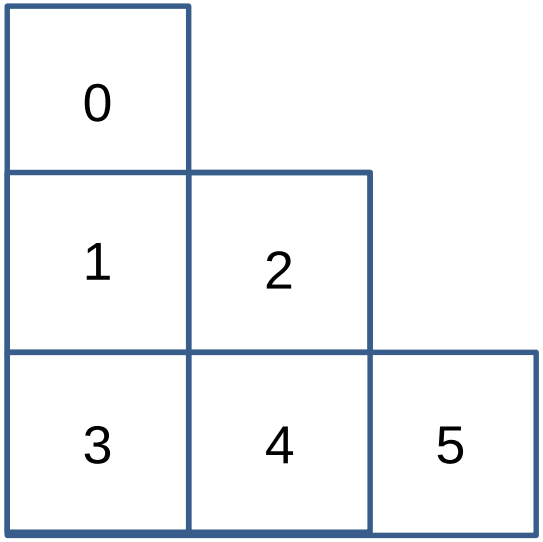
Τυπώνει “2 3”

Ασύμμετρος πίνακας

Τυπώνει “2 3 5”

Αρχικοποίηση πινάκων

- Δημιουργήστε τους παρακάτω πίνακες:
  - Ένα μονοδιάστατο πίνακα με n θέσεις με τις τιμές 0..n-1
  - Ένα δισδιάστατο πίνακα με nxn θέσεις με τις τιμές 0…n^2-1
  - Ένα κάτω διαγώνιο πίνακα nxn (π.χ. παρακάτω 3x3)
- Το μέγεθος του πίνακα θα το δίνει ο χρήστης



```
import java.util.Scanner;

class ArrayInitialization
{
	public static void main(String[] args){
		Scanner input = new Scanner(System.in);
		int n = input.nextInt();

		int[] array1d = new int[n];
		for (int i = 0; i < n; i ++){
			array1d[i] = i;
		}
		for (int i = 0; i < n; i ++){
			System.out.print(array1d[i] + " ");
		}
		System.out.println();
	}
}
```

```
import java.util.Scanner;

class ArrayInitialization
{
	public static void main(String[] args){
		Scanner input = new Scanner(System.in);
		int n = input.nextInt();

		int[][] array2d = new int[n][n];
		for (int i = 0; i < n; i ++){
			for (int j = 0; j < n; j ++){
				array2d[i][j] = i*n+j;
			}
		}
		for (int i = 0; i < n; i ++){
			for (int j = 0; j < n; j ++){
				System.out.print(array2d[i][j] + " " );
			}
		System.out.println();
		}
	}
}
```

```
import java.util.Scanner;

class ArrayInitialization
{
	public static void main(String[] args){
		Scanner input = new Scanner(System.in);
		int n = input.nextInt();

		int[][] lowerDiagonal = new int[n][];
		for (int i = 0; i < n; i ++){
			lowerDiagonal[i] = new int[i+1];
			for (int j = 0; j < i+1; j ++){
				lowerDiagonal[i][j] = i*(i+1)/2 + j;
			}
		}
		for (int i = 0; i < n; i ++){
			for (int j = 0; j < i+1; j ++){
				System.out.print(lowerDiagonal[i][j] + " ");
				}
			System.out.println();
		}
	}
}
```

Πίνακες από Strings

- Φτιάξετε ένα πρόγραμμα που να διαβάζει από την είσοδο το όνομα και το επώνυμο n ατόμων και τα αποθηκεύει. Το n δίνεται στην είσοδο

```
import java.util.Scanner;

class Names
{
	public static void main(String[] args)
	{
		Scanner input = new Scanner(System.in);
		System.out.println("Give number of persons:");
		int n = input.nextInt();
		String[][] nameArray = new String[n][2];
		for (int i = 0; i < n; i ++){
			nameArray[i][0] = input.next();
			nameArray[i][1] = input.next();
		}

		for (int i = 0; i < n; i ++){
			System.out.println(nameArray[i][0]
					+ " " + nameArray[i][1]);
		}
	}
}
```

Παράδειγμα με strings και πίνακες

- Φτιάξτε ένα πρόγραμμα που να διαβάζει μία γραμμή από κείμενο και να ψάχνει μία λέξη που δίνουμε σαν όρισμα μέσα σε αυτή τη γραμμή.

➢ **`java LookFor hello`**

  - Περιμένει να διαβάσει μια γραμμή από κείμενο και ψάχνει τη λέξη hello 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);
           }
       }
    }
}
```

Τα command-line ορίσματα του προγράμματος αποθηκεύονται στον πίνακα από Strings που είναι όρισμα στην main()

Η μέθοδος split της κλάσης String με όρισμα ένα delimiter string σπάει το String με βάση το delimiter και επιστρέφει ένα πίνακα από Strings

Στην περίπτωση αυτή σπάμε το line με βάση το κενό και παίρνουμε τις λέξεις.

Η κλάση ArrayList

- Η κλάση ArrayList ορίζει ένας δυναμικό πίνακα με μεταβλητό μέγεθος ανάλογα με τον αριθμό των στοιχείων που τοποθετούμε
  - Παρόμοιο με το list στην Python
  - Το ArrayList κρατάει αντικείμενα συγκεκριμένου τύπου.

- Συντακτικό:
  - `import java.util.ArrayList;`
  - `ArrayList<Τύπος> myList;`

- Ο τύπος είναι οποιοσδήποτε μια οποιαδήποτε κλάση.
  - Αυτός είναι ο τύπος των δεδομένων που αποθηκεύει ο πίνακας μας.
  - Για να αποθηκεύσουμε βασικούς τύπους χρειαζόμαστε την wrapper class.

- Παραδείγματα:
  - `ArrayList<Integer> myList;` // λίστα από ακεραίους
  - `ArrayList<String> myList;` // λίστα από String

ArrayList

- Δημιουργία αντικειμένου
  - `ArrayList<T> myList = new ArrayList<T>();`
- Μέθοδοι
  - `size()`: ο αριθμός των στοιχείων του πίνακα.
  - `add(T x)`: προσθέτει το στοιχειό `x` στο τέλος του πίνακα.
  - `add(int i, T x)`: προσθέτει το στοιχειό `x` στη θέση `i` και μετατοπίζει τα υπόλοιπα στοιχεία κατά μια θέση.
  - `set(int i, T x)`: θέτει στην θέση `i` την τιμή `x` αλλάζοντας την προηγούμενη
  - `get(int i)`: επιστρέφει το αντικείμενο τύπου `T` στη θέση `i`.
  - `remove(int i)`: αφαιρεί το στοιχείο στη θέση `i`
  - `remove(T x)`: αφαιρεί την πρώτη εμφάνιση του `x`
  - `contains(T x)`: επιστρέφει `true/false` αν το `x` περιέρχεται στην λίστα
- Διατρέχοντας τον πίνακα:
  - `ArrayList<T> myList = new ArrayList<T>();`
  - `for(T x: myList){…}`

Παράδειγμα

- Ζητάμε από την είσοδο θετικούς ακεραίους αριθμούς μέχρι ο χρήστης να δώσει το -1. Αποθηκεύστε τους αριθμούς σε ένα πίνακα και τυπώστε τους

- Δεν ξέρουμε εκ των προτέρων πόσους αριθμούς θα πρέπει να αποθηκεύσουμε.
  - Θα χρησιμοποιήσουμε ArrayList αντί για πίνακα.

```
import java.util.ArrayList;
import java.util.Scanner;

class ArrayListTest
{
    public static void main(String[] args){
       ArrayList<Integer> numbers = new ArrayList<Integer>();
       Scanner input = new Scanner(System.in);
       int x = input.nextInt();
       while (x != -1){
           numbers.add(x);
           x = input.nextInt();
       }

       for (Integer y: numbers){
           System.out.print(y + " ");
       }
       System.out.println();
    }
}
```

Πίνακες από Strings II

- Φτιάξετε ένα πρόγραμμα που να διαβάζει από την είσοδο όνομα και επώνυμο μέχρι να τερματίσουμε την είσοδο.

```
import java.util.Scanner;
import java.util.ArrayList;

class Names2
{
	public static void main(String[] args)
	{
		Scanner input = new Scanner(System.in);
		ArrayList<String[]> nameList =
					new ArrayList<String[]>();
		while (input.hasNext()){
			String[] nameArray = new String[2];
			nameArray[0] = input.next();
			nameArray[1] = input.next();
			nameList.add(nameArray);
		}

		for (String[] nameArray:nameList){
			System.out.println(nameArray[0]
					+ " " + nameArray[1]);
		}
	}
}
```

Ορισμός λίστας από πίνακες String

Σύνολα

- Οι πίνακες και η κλάση ArrayList μας επιτρέπουν να κρατάμε μια διατεταγμένη σειρά από αντικείμενα
- Αν δεν μας ενδιαφέρει η διάταξη, αλλά μόνο η συλλογή των αντικειμένων μπορούμε να τα αποθηκεύσουμε σε ένα σύνολο.
- Στην Java μια κλάση για να ορίσουμε ένα σύνολο από αντικείμενα είναι η κλάση HashSet
- Τα σύνολα μας επιτρέπουν γρήγορη αναζήτηση μιας τιμής μέσα στην συλλογή.
- Συντακτικό:
  - `import java.util.HashSet;`
  - `HashSet<Τύπος> mySet;`

HashSet

- Δημιουργία αντικειμένου
  - `HashSet<T> mySet = new HashSet<T>();`
- Μέθοδοι
  - `size()`: ο αριθμός των στοιχείων στο σύνολο.
  - `add(T x)`: προσθέτει το στοιχειό `x` αν δεν υπάρχει ήδη στο σύνολο.
  - `remove(T x)`: αφαιρεί το στοιχείο `x`.
  - `contains(T x)`: επιστρέφει `true/false` αν το σύνολο περιέχει το στοιχείο `x`.
  - `isEmpty()`: boolean αν έχει στοιχεία το σύνολο ή όχι.
- Διατρέχοντας τα στοιχεία του συνόλου:
  - `HashSet<T> mySet = new HashSet<T>();`
  - `for(T x: mySet){…}`

Παράδειγμα Ι

- Διαβάζουμε ένα String που περιέχει ένα κείμενο και θέλουμε να βρούμε όλες τις μοναδικές λέξεις στο κείμενο
  - Π.χ. να φτιάξουμε το λεξικό ενός βιβλίου

```
import java.util.HashSet;
import java.util.Scanner;

public class HashSetExample
{
    public static void main(String[] args){
       HashSet<String> mySet = new HashSet<String>();
       Scanner input = new Scanner(System.in);

       String line = input.nextLine();
       String words[] = line.split(" ");
       for (String word: words){
           if (!mySet.contains(word)){
              mySet.add(word);
           }

       for(String word: mySet){
           System.out.println(word);
       }
    }
}
```

Δήλωση μιας μεταβλητής HashSet από Strings.

Τοποθετούμε στο HashSet μόνο τα Strings τα οποία δεν έχουμε ήδη δει (δεν είναι ήδη στο σύνολο)

Διατρέχουμε και να τυπώνουμε τα στοιχεία του HashSet

```
import java.util.HashSet;
import java.util.Scanner;

public class HashSetExample
{
    public static void main(String[] args){
       HashSet<String> mySet = new HashSet<String>();
       Scanner input = new Scanner(System.in);

       String text = input.nextLine();
       String words[] = text.split(" ");
       for (String word: words){
           mySet.add(word);
       }

       for(String word: mySet){
           System.out.println(word);
       }
    }
}
```

Επειδή το HashSet κρατάει μοναδικά αντικείμενα, δεν χρειάζεται να κάνουμε τον έλεγχο. Αν υπάρχει ήδη το String δεν θα το ξαναπροθέσει.

Λεξικά

- Οι πίνακες και η κλάση ArrayList μας επιτρέπουν να κρατάμε μια διατεταγμένη σειρά από αντικείμενα και τα δεικτοδοτούν με τη θέση τους.
- Αν θέλουμε να δεικτοδοτήσουμε με κάποιο κλειδί χρειαζόμαστε λεξικό.
- Τα λεξικά κρατάνε ζευγάρια της μορφής (κλειδί,τιμή) και επιτρέπουν γρήγορη αναζήτηση με το κλειδί
- Συντακτικό
  - `import java.util.HashMap;`
  - `HashMap<Τύπος Κλειδιου,Τύπος Τιμής> myMap;`

HashMap

- Δημιουργία αντικειμένου
  - `HashMap<K,V> myMap = new HashMap<K,V>();`
- Μέθοδοι
  - `size()`: ο αριθμός των στοιχείων (κλειδιών) στο map.
  - `put(K key, V value)`: προσθέτει το ζευγάρι (`key,value`) (δημιουργεί μία συσχέτιση)
  - `V get(K key)`: επιστρέφει την τιμή για το κλειδί `key`.
  - `remove(K key)`: αφαιρεί το ζευγάρι με κλειδί `key`.
  - `containsKey(K key)`: boolean αν το σύνολο περιέχει το κλειδί `key` ή όχι.
  - `containsValue(V value)`: boolean αν το σύνολο περιέχει την τιμή `value` ή όχι. (αργό)
  - `isEmpty()`: boolean αν έχει στοιχεία το map ή όχι.
  - `Set<K> keySet()`: επιστρέφει ένα Set με τα κλειδιά.

Παράδειγμα ΙΙ

- Διαβάζουμε ένα String που αναπαριστά ένα κείμενο και θέλουμε να βρούμε όλες τις μοναδικές λέξεις μέσα στο κείμενο και τον αριθμό των εμφανίσεων τους.

- Πώς θα το υλοποιήσουμε αυτό?

```
import java.util.HashMap;
import java.util.Scanner;

class HashMapExample
{
    public static void main(String[] args){
        HashMap<String, Integer> myMap =
                                new HashMap<String,Integer>();
        Scanner input = new Scanner(System.in);

        String text = input.nextLine();
        String words[] = text.split(" ");
        for (String word: words){
            if (!myMap.containsKey(word)){
                myMap.put(word,1);
            }else{
                myMap.put(word,myMap.get(word)+1);
            }
        for(String word: myMap.keySet()){
            System.out.println(word+":"+myMap.get(word));
        }
    }
}
```

Δήλωση μιας μεταβλητής HashMap που συσχετίζει Strings (κλειδια) και Integers (τιμές). Για κάθε λέξη (String) τον αριθμό ενφανίσεων (Integer)

Αν η λέξη δεν είναι ήδη στο HashMap τότε πρόσθεσε την με τιμή 1.

Αλλιώς πάρε την τιμή της λέξης, αύξησε την κατά 1, και βάλε τη πίσω σαν νέα τιμή

Διέτρεξε το σύνολο με τα κλειδιά (ονόματα) στο HashMap Για κάθε κλειδί πάρε και τύπωσε την τιμή του.