ΤΕΧΝΙΚΕΣ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΟΥΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Χρήση Δομών

String processing

Στατικές μέθοδοι και μεταβλητές

ΜΑΘΗΜΑΤΑ ΑΠΟ ΤΟ ΕΡΓΑΣΤΗΡΙΟ

Περίπλοκες δομές

- Έχουμε μάθει τρεις βασικές δομές
  - `ArrayList`
  - `HashSet`
  - `HashMap`
- Μπορούμε να δημιουργήσουμε αντικείμενα που συνδιάζουν αυτές τις δομές
  - `HashMap<String,ArrayList<String>>`
  - `ArrayList<HashSet<String>>`
  - `HashMap<Integer,HashMap<String,String>>`

Εργαστήριο

- Για την άσκηση του εργαστηρίου έπρεπε να ορίσετε ένα

```
HashMap<String,ArrayList<String>> myMap =
        new HashMap<String,ArrayList<String>>();
```

  - Το κλειδί είναι το όνομα του ανθρώπου
  - Η τιμή είναι η λίστα με τις πόλεις
- ΔΕΝ μας κάνει ένα HashSet (ούτε ένα ArrayList) γιατί με κάθε όνομα θέλουμε να συσχετίσουμε κάτι (μια λίστα με ονόματα)
- ΔΕΝ μας κάνει ένα HashMap<String,String> γιατί τότε με κάθε όνομα θα συσχετίζαμε μόνο ένα όνομα, ενώ εμείς θέλουμε μια λίστα.

HashMap

- Την πρώτη φορά που βρίσκουμε ένα καινούριο όνομα (name) θα πρέπει να το προσθέτουμε στο HashMap και να δημιουργούμε ένα καινούριο ArrayList γι αυτό το όνομα
  - Μπορούμε να χρησιμοποιήσουμε και ένα ανώνυμο αντικείμενο

```
myMap.put(name, new ArrayList<String>());
```

  - Η λίστα αυτή αρχικά είναι άδεια, σε αυτή θα προσθέσουμε τις πόλεις
- Αν το όνομα υπάρχει ήδη, τότε μπορούμε να πάρουμε το ArrayList που αντιστοιχεί στο όνομα name από το HashMap και να προσθέσουμε το όνομα της πόλης (city)

```
myMap.get(name).add(city)
```

- Προσέξτε ότι τελικά το HashMap θα έχει το σύνολο των διακριτών ονομάτων

Χρήση δομών

- ArrayList: όταν θέλουμε να διατρέχουμε τα αντικείμενα και δεν θα χρειαστούμε random access σε κάποιο αντικείμενο
  - Π.χ., μια κλάση Exam περιέχει μια λίστα από αντικείμενα τύπου Questions
    - Για κάθε ερώτηση την τυπώνουμε, παίρνουμε την απάντηση, τη βαθμολογούμε, κλπ.

- HashSet: όταν θέλουμε να έχουμε μια συλλογή από μοναδικά αντικείμενα και θέλουμε γρήγορη αναζήτηση για να μάθουμε αν κάποιο αντικείμενο ανήκει σε αυτή
  - Π.χ., να βρούμε τα μοναδικά ονόματα από μια λίστα με ονόματα με επαναλήψεις
  - Π.χ., να βρούμε αν ένα όνομα ανήκει σε μια λίστα ονομάτων με διασημότητες

- HashMap: Ίδια λειτουργικότητα με το HashSet αλλά μας επιτρέπει να συσχετίσουμε μια τιμή με κάθε στοιχείο του συνόλου
  - Π.χ. θέλω να ανακαλέσω γρήγορα τις πληροφορίες για ένα φοιτητή χρησιμοποιώντας το ΑΜ του
  - Το HashMap είναι πιο χρήσιμο απ ότι ίσως θα περιμένατε

Παραδείγματα

- Στο παράδειγμα του εργαστηρίου έχουμε εκατομμύρια ανθρώπους και χιλιάδες πόλεις
  - Τι δομή θα πρέπει να χρησιμοποιήσουμε για να βρούμε γρήγορα αν ένας άνθρωπος επισκέφτηκε μια πόλη?
  - Αν μας ενδιαφέρει να κρατάμε και πόσες φορές την επισκέφτηκε?
- Στο πρόγραμμα της γραμματείας ενός πανεπιστημίου που κρατάει πληροφορία για τους φοιτητές, θέλω γρήγορα με το ΑΜ του φοιτητή να μπορώ να βρω το βαθμό για ένα μάθημα χρησιμοποιώντας τον κωδικό του μαθήματος. Τι δομή πρέπει να χρησιμοποιήσω?
  - Τι γίνεται αν θέλω στο πρόγραμμα μου να έχω μια κλάση Student που κρατάει τις πληροφορίες για ένα φοιτητή?

STRING PROCESSING

Κανονικές Εκφράσεις στη Java

- Μπορείτε να διαβάσετε μια περίληψη στη σελίδα της Oracle
- Οι κανονικές εκφράσεις μπορούν να περιγράψουν πολλά πράγματα. Εμείς θα χρησιμοποιήσουμε κάποιες απλές εκφράσεις.
- Παραδείγματα:
  - [abc]: a ή b ή c
  - ^a : Ξεκινάει με a
  - a$: τελειώνει με a
  - \s ή \p{Space}: white space (κενό, tab, αλλαγή γραμμής)
  - \p{Punct}: όλα τα σημεία στίξης

- Για να χρησιμοποιήσουμε τις κανονικές εκφράσεις τις μετατρέπουμε σε ένα string που δίνεται ως όρισμα στην split η την replaceAll.
  - Π.χ. `"[abc]"`, `"^a"`, `"a$"`, `"\\s"`, `"\\p{Space}"`, `"\\p{Punct}"`
    - Χρειαζόμαστε το "\\" ώστε να βάλουμε το \ μέσα στο string.

Παρένθεση

- Ο χαρακτήρας **** λέγεται escape character
  - Όταν τον συνδυάζουμε με άλλους χαρακτήρες παίρνει διαφορετικό νόημα όταν είμαστε μέσα σε String
    - **\n**: αλλαγή γραμμής
    - **\t**: tab
    - **\“**: ο χαρακτήρας **“**
    - ****: ο χαρακτήρας ****

Παράδειγμα

```
class SplitTest2
{
    public static void main(String args[]){
        String s1 = "sentense 1\tsentence 2";
        String[] tokens = s1.split("[\t ]");
        for (String t: tokens){
            System.out.println(t);
        }
        tokens = s1.split("\\s");
        for (String t: tokens){
            System.out.println(t);
        }

        String s2 =  "To be or not to be? This is the question. The
question we must face";
        String[] sentences = s2.split("[?.]");
        for (String s: sentences){
            System.out.println(s.trim());
        }
    }
}
```

Split στο tab και το κενό

Split σε οποιοδήποτε white space

Split στο ερωτηματικό και την τελεία

Παράδειγμα

Για να χρησιμοποιήσουμε την κανονική έκφραση χρειαζόμαστε την εντολή replaceAll

```
class ReplaceTest2
{
    public static void main(String args[]){
        String s = "The cost is 99.99 dollars.";
        System.out.println(s);
        s = s.replaceAll("[.]$","");
        System.out.println(s);

        s = "\"Quoted (\"quote\") text\"";
        System.out.println(s);
        s = s.replaceAll("^\"","");
        s = s.replaceAll("\"$","");
        System.out.println(s);

        s = "What?Yes!No...";
        System.out.println(s);
        s = s.replaceAll("[.!?]"," ");
        System.out.println(s);

        s = "Space: Tab:\t:End";
        System.out.println(s);
        s = s.replaceAll("\\p{Space}","");
        System.out.println(s);
    }
}
```

Σβήνει την τελεία στο τέλος του String

Σβήνει το “ στην αρχή του String

Σβήνει το ” στο τέλος του String

Αντικαθιστά τελεία, θαυμαστικό και ερωτηματικό με κενό.

Σβήνει τους whitespace χαρακτήρες

FILE

Η κλάση File

- Η κλάση File μας δίνει πληροφορίες για ένα αρχείο που θα μπορούσαμε να πάρουμε από το λειτουργικό σύστημα
- Constructor:
  - File fileObject = new File(<αφηρημένο όνομα>);
  - Το αφηρημένο όνομα συνήθως θα είναι ένα όνομα αρχείου, αλλά μπορεί να είναι και directory.
- Μέθοδοι:
  - exists(): επιστρέφει boolean αν υπάρχει ή όχι το αρχείο/path
  - getName(): επιστρέφει το όνομα του αρχείου από το full path name
  - getPath(): επιστρέφει το path μέχρι το αρχείο από το full path name
  - isFile(): boolean που μας λέει αν το όνομα είναι αρχείο η όχι
  - isDirectory(): boolean που μας λέει αν το όνομα είναι directory η όχι
  - mkdir(): δημιουργεί το directory στο path που δώσαμε ως όρισμα.

STATIC

Στατικές μέθοδοι

- Τι σημαίνει το keyword static στον ορισμό της main μεθόδου? Τι είναι μια στατική μέθοδος?
- Μια στατική μέθοδος μπορεί να κληθεί χωρίς αντικείμενο της κλάσης, χρησιμοποιώντας κατευθείαν το όνομα της κλάσης
  - Η μέθοδος ανήκει στην κλάση και όχι σε κάποιο συγκεκριμένο αντικείμενο.
  - Όταν καλούμε την συνάρτηση main κατά την εκτέλεση του προγράμματος δεν δημιουργούμε κάποιο αντικείμενο της κλάσης
  - Χρήσιμο για τον ορισμό βοηθητικών μεθόδων

Συντακτικό

- Ορισμός

```
class myClass
{
        ...

        public static ReturnType methodName(arguments)
        { … }

        ...
}
```

- Κλήση

```
myClass.methodName(arguments)
```

Παράδειγμα

Ορισμός

```
class Auxiliary
{
   public static int max(int x, int y){
       if (x > y){
           return x;
       }
       return y;
   }
}
```

Κλήση

```
int m = Auxiliary.max(6,5);
```

Η κλήση της μεθόδου max δεν χρειάζεται τον ορισμό αντικείμενου
Γίνεται χρησιμοποιώντας κατευθείαν το όνομα της κλάσης

Παρένθεση

- Ένας άλλος τρόπος να υλοποιήσετε το max τελεστή

```
public static int max(int x, int y){
      return (x>y)? x: y;
}
```

Η έκφραση:

```
condition ? value_if_true: value_if_false
```

επιστέφει μια τιμή ανάλογα με την αποτίμηση του condition και είναι ένας γρήγορος τρόπος να υλοποιήσουμε ένα if το οποίο επιστρέφει μία τιμή

Στατικές μεταβλητές

- Παρόμοια με τις στατικές μεθόδους μπορούμε να ορίσουμε και στατικές μεταβλητές
  - Οι στατικές μεταβλητές ανήκουν στην κλάση και όχι σε κάποιο συγκεκριμένο αντικείμενο και, εφόσον είναι public μπορούμε να έχουμε πρόσβαση σε αυτές χρησιμοποιώντας το όνομα της κλάσης χωρίς να έχουμε ορίσει κάποιο αντικείμενο.

Συντακτικό

- Ορισμός

```
class myClass
{
        public static Type varName;

        public static ReturnType methodName(arguments)
        { … }

        ...
}
```

- Κλήση

```
…. myClass.varName…. ;
```

Παράδειγμα

Ορισμός

```
class Auxiliary
{
    public static int factor = 2.0;

    public static int max(int x, int y){
        if (x > y){
            return x;
        }
        return y;
    }
}
```

Κλήση

```
int m =
    Auxiliary.factor * Auxiliary.max(6,5);
```

Σταθερές

- Οι στατικές μεταβλητές πολλές φορές χρησιμοποιούνται για να ορίσουμε σταθερές.
  - Τις ορίσουμε σε μία κλάση και μπορούμε να τις χρησιμοποιούμε σε διάφορα σημεία στο πρόγραμμα.

- Για να προσδιορίσουμε ότι μία μεταβλητή είναι σταθερά μπορούμε να χρησιμοποιήσουμε το keyword final.

Παράδειγμα

Ορισμός

```
class Circle
{
    public static final double PI = 3.14;

    public static double area(double r){
        return PI*r*r;
    }
}
```

Κλήση

```
int unitCircleArea = Circle.area(1);
System.out.println("PI value is" + Circle.PI);
```

Στατικές μέθοδοι

- Όταν ορίζουμε μια στατική μέθοδο μέσα σε μία κλάση, δεν μπορούμε να χρησιμοποιούμε μη στατικά πεδία, ή να καλούμε μη στατικές μεθόδους.
  - Μη στατικά πεδία και μη στατικές μέθοδοι συσχετίζονται με ένα αντικείμενο. Εφόσον μπορούμε να καλέσουμε μια στατική μέθοδο χωρίς αντικείμενο, δεν μπορούμε μέσα σε αυτή να χρησιμοποιούμε μη στατικά πεδία ή μεθόδους.
  - Σκεφτείτε ότι για κάθε χρήση μιας μεθόδου ή μιας μεταβλητής μπορούμε να βάλουμε το this μπροστά. Αν δεν υπάρχει αντικείμενο η αναφορά this δεν ορίζεται
- Αν θέλουμε να καλέσουμε μια μη στατική μέθοδο θα πρέπει να ορίσουμε ένα αντικείμενο μέσα στην στατική μέθοδο

Παράδειγμα

```
class Auxiliary2
{
        private int x;
        private int y;

        public Auxiliary2(int x, int y){
                this.x = x;
                this.y = y;
        }

        public int max(){
                return (x>y)? x: y;
        }

        public int min(){
                return (x>y)? y: x;
        }

        public static double maxToMin(int x, int y){
                Auxiliary2 aux = new Auxiliary2(x,y);
                return ((double)aux.max())/aux.min();
        }
}
```

Στατικές μεταβλητές

- Εκτός από σταθερές μπορούμε να ορίσουμε στατικές μεταβλητές όταν θέλουμε διαφορετικά αντικείμενα να επικοινωνούν μέσω μιας μεταβλητής
  - Υπάρχει μόνο ένα αντίγραφο μιας στατικής μεταβλητής, άρα όταν το αλλάζει ένα αντικείμενο την αλλαγή την βλέπουν και όλα τα άλλα αντικείμενα της κλάσης.
- Παράδειγμα: Στο πρόγραμμα TakeTurns δείχνουμε πως μπορούμε να χρησιμοποιήσουμε στατικές μεταβλητές για να επικοινωνούν μεταξύ τους τα αντικείμενα.

Στατικές μέθοδοι και μεταβλητές

- Έχετε ήδη χρησιμοποιήσει στατικές μεθόδους και μεταβλητές σε διάφορες περιπτώσεις

- Παραδείγματα
  - System.out: στατικό πεδίο της κλάσης System, το οποίο κρατάει ένα PrintStream με το οποίο μπορούμε γράψουμε στην οθόνη.
  - System.in: στατικό πεδίο της κλάσης System, το οποίο κρατάει ένα FileInputStream που συνδέεται με το πληκτρολόγιο.
  - System.exit(): στατική μέθοδος της κλάσης System

Περιβάλλουσες κλάσεις

- Οι wrapper classes Integer, Double, Boolean και Character έχουν πολλές στατικές μεθόδους και στατικά πεδία που μας βοηθάνε να χειριζόμαστε τους βασικούς τύπους.
  - Integer.parseInt(String): Μετατρέπει ένα String σε int.
    - Αντίστοιχα: Double.parseDouble(String), Boolean.parseBoolean(String)
  - Integer.MAX_VALUE, Integer.MIN_VALUE: Μέγιστη και ελάχιστη τιμή ενός ακεραίου
    - Αντίστοιχα: Double.MAX_VALUE, Double.MIN_VALUE
  - Character.IsDigit(char): επιστρέφει true αν ο χαρακτήρας είναι ένα ψηφίο
    - Παρόμοια: Character.IsLetter(char), Character.IsLetterOrDigit(), Character.IsWhiteSpace(char)
- Οι κλάσεις αυτές έχουν και μη στατικές μεθόδους.

Η κλάση Math

- Μία κλάση με πολλές στατικές μεθόδους και στατικά πεδία για μαθηματικούς υπολογισμούς
- Παραδείγματα
  - min: επιστρέφει το ελάχιστο δύο αριθμών
  - max: επιστρέφει το μέγιστο δύο αριθμών
  - abs: επιστρέφει την απόλυτη τιμή
  - pow(x,y): υψώνει το x στην y δυναμη
  - floor/ceil: επιστρέφει τον μεγαλύτερο/μικρότερο ακέραιο που είναι μικρότερος/μεγαλυτερος από το όρισμα
  - sqrt: επιστρέφει την τετραγωνική ρίζα ενός αριθμού
  - PI: ο αριθμός π
  - E: Η βάση των φυσικών λογαρίθμων

Συμπερασματικά

- Στατικές μεθόδους και πεδία συνήθως ορίζουμε όταν θέλουμε μια βοηθητική συλλογή από σταθερές και μεθόδους (παρόμοια με την κλάση Math της Java).
- Μια στατική μέθοδο που μπορείτε να ορίσετε για κάθε κλάση είναι η main, ώστε να τεστάρετε μια συγκεκριμένη κλάση.

ΕΣΩΤΕΡΙΚΕΣ ΚΛΑΣΕΙΣ

Εσωτερικές κλάσεις

- Μπορούμε να ορίσουμε μια κλάση μέσα στον ορισμό μιας άλλης κλάσης

```
class Shape
{
        private class Point
        {

                <Code for Point>
        }

        <Code for Shape>
}
```

Γιατί να το κάνουμε αυτό?

- Η κλάση Point μπορεί να είναι χρήσιμη μόνο για την Shape
- Μας επιτρέπει να ορίσουμε άλλη Point σε άλλο σημείο

- Η Point και η Shape έχουν η μία πρόσβαση στα ιδιωτικά πεδία και μεθόδους της άλλης