ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Αντικειμενοστρεφής Προγραμματισμός

Εργαστήριο 1

Εισαγωγή στον Αντικειμενοστρεφή Προγραμματισμό

1. Φοιτητές - Κλάση Student

Έστω ότι θέλουμε να περιγράψουμε έναν φοιτητή. Για να το πετύχουμε αποφασίζουμε να καταγράψουμε τα παρακάτω χαρακτηριστικά: *Αριθμός Μητρώου, όνομα, επώνυμο, πλήθος απουσιών, βαθμός στο εργαστήριο, βαθμός στη θεωρία.*

- Να δημιουργηθεί η κλάση Student, η οποία θα περιγράφει έναν φοιτητή, όπως παρουσιάστηκε παραπάνω. Εκτός από δομητές, μεθόδους set και get και τη μέθοδο toString(), η κλάση θα περιλαμβάνει την μέθοδο computeFinal(), η οποία υπολογίζει και επιτρέφει τον τελικό του βαθμό. Ο τελικός βαθμός του φοιτητή προκύπτει ως εξής:

  τελικός Βαθμός= 30%* βαθμός στο εργαστήριο + 70%* βαθμός στη θεωρία.
- Να δημιουργηθεί μια κλάση TestStudent η οποία θα περιέχει τη main() και με την οποία θα ελέγξετε την λειτουργία της κλάσης Student. Δημιουργείστε δύο αντικείμενα τύπου φοιτητή και εμφανίστε τα στοιχεία τους, συμπεριλαμβανομένου και του τελικού βαθμού τους.

Υπόδειξη: για την εισαγωγή των στοιχείων των φοιτητών μπορείτε να χρησιμοποιήσετε την κλάση UserInput.

2. Επαύξηση της άσκησης 1

Στην κλάση TestStudent προσθέστε την στατική μέθοδο compareThreeStudentMarks η οποία δέχεται ως παράμετρο τρεις φοιτητές και συγκρίνει τον τελικό βαθμό τους. Η μέθοδος εμφανίζει μία αριθμημένη λίστα με τα ονόματα και τον τελικό βαθμό των τριών φοιτητών κατά φθίνουσα σειρά.

3. Εργαζόμενοι - κλάση Employee

Έστω ότι θέλουμε να περιγράψουμε έναν εργαζόμενο μιας εταιρείας. Για να το πετύχουμε αποφασίζουμε να καταγράψουμε τα παρακάτω στοιχεία: *Αριθμός Μητρώου, όνομα, επώνυμο, βασικός μισθός, έτη απασχόλησης, bonus.*

- Να δημιουργηθεί η κλάση Employee, η οποία θα περιγράφει έναν εργαζόμενο, όπως παρουσιάστηκε παραπάνω. Εκτός από δομητές, μεθόδους πρόσβασης (get) και

μεταβολής (set) και τη μέθοδο toString(), η κλάση θα περιλαμβάνει την μέθοδο salary(), η οποία υπολογίζει τον τελικό του μισθό. Θεωρείστε ότι ο τελικός μισθός προκύπτει ως εξής: τελικός μισθός = βασικός μισθός + έτη απασχόλησης*10+bonus.

- Να δημιουργηθεί μια κλάση TestEmployee η οποία θα περιέχει τη main() και με την οποία θα ελέγξετε την λειτουργία της κλάσης Employee. Δημιουργείστε δύο αντικείμενα τύπου εργαζόμενος και εμφανίστε τα στοιχεία τους, συμπεριλαμβανομένου και του τελικού μισθού τους.

Υπόδειξη: για την εισαγωγή των στοιχείων των φοιτητών μπορείτε να χρησιμοποιήσετε την κλάση UserInput.

4. Επαύξηση της άσκησης 3

Στην κλάση TestEmployee προσθέστε την στατική μέθοδο FindLowSalary η οποία δέχεται ως παράμετρο δύο εργαζόμενους και ελέγχει αν κάποιος έχει βασικό μισθό λιγότερο από 1200 ευρώ και έτη απασχόλησης περισσότερα ή ίσα του 5. Αν οι παραπάνω συνθήκες ισχύουν για κάποιον εργαζόμενο, η μέθοδος εμφανίζει ένα μήνυμα που πληροφορεί ότι ο συγκεκριμένος εργαζόμενος είναι χαμηλόμισθος. Στη συνέχεια η μέθοδος αυξάνει τον βασικό μισθό του χαμηλόμισθου κατά 100 ευρώ και εμφανίζει τα νέα του στοιχεία, συμπεριλαμβανομένου και του τελικού μισθού του.

Βοήθεια:

1. Χρησιμοποιείστε την παρακάτω κλάση (ή εμπλουτίστε την υπάρχουσα) για την εισαγωγή Strings από το πληκτρολόγιο, από την οποία θα χρησιμοποιηθεί η μέθοδος getString() για την εισαγωγή του προς αναζήτηση επωνύμου.

```
import java.io.*;

class UserInput {                    //Class gia eisagogi dedomenwn apo to pliktrologio

    public static String getString() {     //Methodos gia eisagogi String
        String line;
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(isr);
        try {
            line=br.readLine();
            return line;
        }
        catch(Exception e) {
            return "Exception - Lathos";
        }
    }

    public static int getInteger() {                    //Methodos gia eisagogi Integer
        String line;
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(isr);
        try {
            line=br.readLine();
            int i=Integer.parseInt(line);
            return i;
        }
         catch(Exception e) {
            return -1;
        }
    }

    public static short getShort() {                    //Methodos gia eisagogi short
        String line;
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(isr);
        try {
            line = br.readLine();
            short i = Short.parseShort(line);
```

```
                return i;
        }
         catch(Exception e) {
                return (short)-1;
        }
}

public static long getLong() {                  //Methodos gia eisagogi long
        String line;
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(isr);
        try {
                line = br.readLine();
                long i = Long.parseLong(line);
                return i;
        }
         catch(Exception e) {
                return -1L;
        }
}

public static float getFloat() {                //Methodos gia eisagogi Float
        String line;
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(isr);
        try {
                line = br.readLine();
                float f = Float.parseFloat(line);
                return f;
        }
         catch(Exception e) {
                return -1f;
        }
}

public static double getDouble() {              //Methodos gia eisagogi Double
        String line;
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(isr);
        try {
                line = br.readLine();
                double d = Double.parseDouble(line);
                return d;
        }
         catch(Exception e) {
                return -1.0;
```

```
        }
    }

}
```