Άσκηση 4η

Περιγραφή του προβλήματος

Ο σκοπός της 4ης άσκησης είναι η απόκτηση εμπειρικής γνώσης σχετικά με τη χρήση της κληρονομικότητας (inheritance), του πολυμορφισμού και τη χρήση *super, final, enumeration* .

Περιγραφή του προβλήματος: Ένα πανεπιστήμιο έχει ακαδημαϊκό προσωπικό το οποίο διακρίνεται σε μόνιμους καθηγητές και ωρομίσθιους εκπαιδευτικούς (με μισθό ανά ώρα διδασκαλίας).

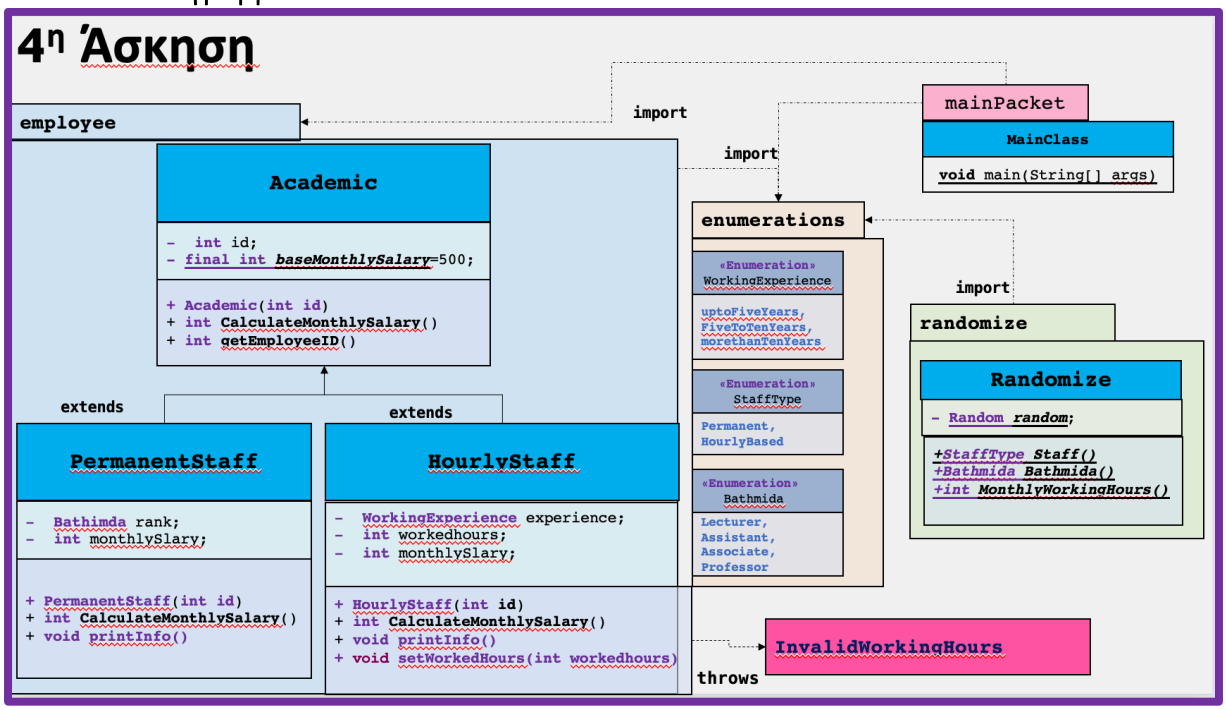
- Όλοι οι υπάλληλοι του πανεπιστημίου ξεκινούν την μηνιαία τους μισθοδοσία με ένα σταθερό ποσό των 500 ευρώ.
- Οι ωρομίσθιοι εκπαιδευτικοί ανά ώρα διαθέτουν επιπλέον ωριαία αμοιβή η οποία εξαρτάτε από το σύνολο των ωρών που δίδαξαν σε ένα μήνα και από την μισθολογική τους κλίμακα που προσδιορίζετε από τον χρόνο προϋπηρεσίας τους στο πανεπιστήμιο.
  - Με μέχρι πενταετή προϋπηρεσία είναι 10 Ευρώ ανά ώρα, από πέντε μέχρι δέκα χρόνια προϋπηρεσίας είναι 20 Ευρώ ανά ώρα, ενώ πάνω από δέκα χρόνια προϋπηρεσία η αμοιβή είναι 30 Ευρώ ανά ώρα. Επίσης, κάθε ωρομίσθιος εκπαιδευτικός επιτρέπεται να εργαστεί μέγιστο 40 ώρες το μήνα.
- Για το μόνιμο εκπαιδευτικό προσωπικό η επιπλέον αμοιβή τους, ανεξάρτητα από τις ώρες διδασκαλίας, είναι ανάλογα με τη βαθμίδα (Λέκτορας 1000 Ευρώ/μήνα, Επίκουρος Καθηγητής 1200 Ευρώ/μήνα, Αναπληρωτής Καθηγητής 1400 Ευρώ/μήνα, Καθηγητής 1500 Ευρώ/μήνα).

Ζητούμενα: Καλείστε να δημιουργήσετε ένα πρόγραμμα μισθοδοσίας του πανεπιστημίου με βάση την παραπάνω περιγραφή το οποίο θα προσομοιώνει τη μισθοδοσία 100 ακαδημαϊκών υπαλλήλων. Κάθε ακαδημαϊκός υπάλληλος χαρακτηρίζεται από έναν αύξοντα αριθμό υπαλλήλου και την μηνιαία μισθοδοσία του.

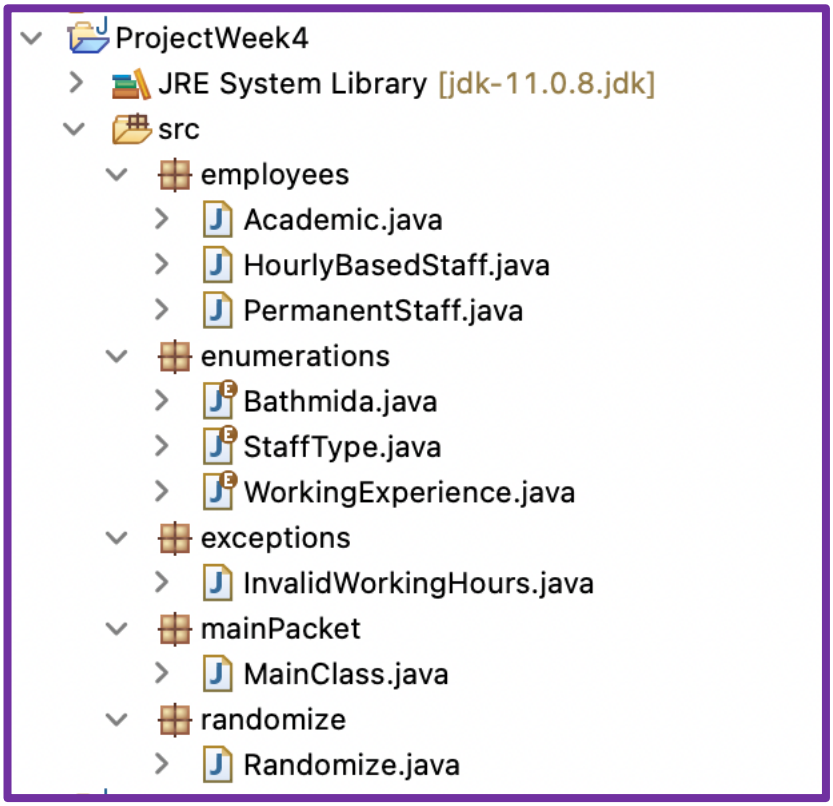
- Το πρόγραμμα θα κατατάσσει τους υπαλλήλους τυχαία (*χρησιμοποιήστε για αυτό το σκοπό την κλάση Random*) είτε σε μόνιμο είτε σε ωρομίσθιο προσωπικό. **Η υλοποίηση της κλάσης Randomize σας δίνεται έτοιμη για να την χρησιμοποιήσετε στο eclass, αλλά μπορείτε να την αγνοήσετε και αν θέλετε να δημιουργήσετε δική σας.**
- Στην συνέχεια θα πρέπει κάθε αντικείμενο στον κατασκευαστή του (*πάλι με χρήση της «τυχαίας» γεννήτριας*) να αποφασίζει την βαθμίδα του υπαλλήλου (αν είναι μόνιμο προσωπικό) ή την προϋπηρεσία του (αν είναι ωρομίσθιο προσωπικό).
- Τέλος το πρόγραμμα πρέπει να εκτυπώνει στατιστικά στοιχεία της μισθοδοσίας όπως για παράδειγμα το κόστος των μόνιμων και των μισθωτών υπαλλήλων.

Θα πρέπει να δημιουργήσετε τις κλάσεις και να χρησιμοποιήσετε καλές πρακτικές αντικειμενοστρεφούς προγραμματισμού. Για παράδειγμα, θα μπορούσατε να δημιουργήσετε μια κλάση Randomize η οποία θα εκτελούσε όλες τις τυχαίες λειτουργίες καθώς και enumerations για εύρωστο κώδικα. Επίσης, θα μπορούσατε να υλοποιήσετε μία μέθοδο CalculateMonthlySalary στην κλάση **Academic** η οποία θα πρέπει βάση της παραπάνω περιγραφής να αναιρείται από τις μεθόδους CalculateMonthlySalary που θα υλοποιούν οι υποκλάσεις. Ένα αρχικό και ενδεικτικό σχέδιο των κλάσεων δίνεται παρακάτω

Ενδεικτικό διάγραμμα κλάσεων



Ενδεικτική δόμηση πακέτων και κλάσεων στο Eclipse



Ενδεικτικός Κώδικας της *main*:

```
public class MainClass {
    public static void main(String[] args) {
        System.out.println("HELLO PROJECT WEEK 4");
        createRandomizedEmployees(100);
    }
    public static void createRandomizedEmployees(int k) {
        System.out.println("Creating "+ k +" Randomized Employees");
        for(int i=1;i<=k;i++) {
            StaffType staff=Randomize.Staff();
            if (staff==StaffType.Permanent) {
                PermanentStaff pstaff=new PermanentStaff(i);
                pstaff.CalculateMonthlySalary();
                pstaff.printInfo();
            }

            if (staff==StaffType.HourlyBased) {
                HourlyBasedStaff hstaff=new HourlyBasedStaff(i);
```

Ενδεικτικός Κώδικας για τις κλάσεις του πακέτου Enumeration:

```
public enum StaffType {
    Permanent,
    HourlyBased
}
```

```
public enum WorkingExperience {
uptoFiveYears,
FiveToTenYears,
morethanTenYears
}
```

```
public enum Bathmida {
Lecturer,
Assistant,
Associate,
Professor
}
```

Ενδεικτικός Κώδικας της Academic:

```
public class Academic {

    static final int baseMonthlySalary=500;
    private int id;
    Academic(int id){
        this.id=id;
    }

    public int CalculateMonthlySalary() {
        return baseMonthlySalary;
    }

    public int getEmployeeID() {return id;}
}
```

Ενδεικτικός Κώδικας της *PermanentStaff*:

```
final public class PermanentStaff extends Academic{
    private Bathmida rank;
    private int monthysalary;

    public PermanentStaff(int id){
        super(id);

        rank=Randomize.Bathmida();
    }

    @Override
    public int CalculateMonthlySalary() {
        int onTopSalary=0;
        if (rank==Bathmida.Lecturer) onTopSalary=1000;
        if (rank==Bathmida.Assistant) onTopSalary=1200;
        if (rank==Bathmida.Associate) onTopSalary=1400;
        if (rank==Bathmida.Professor) onTopSalary=1500;
        monthysalary=baseMonthlySalary+onTopSalary;
        return monthysalary;
    }

    public Bathmida getBathmida() {return rank;}

    public void printInfo() {
        System.out.println("");
        System.out.println("EmployeeID:"+super.getEmployeeID()+ " is permanent. ");
        System.out.println("Academic rank:"+rank);
        System.out.println("Hourly based employee salary:"+monthysalary);

    }
}
```