

Αντικειμενοστρεφής Προγραμματισμός

Εξετάσεις Ιουλίου 2023

Θέμα 1.

Ένας τουριστικός προορισμός (**Destination**) περιλαμβάνει το όνομα του προορισμού (**name**), το όνομα του δήμου στον οποίο ανήκει (**municipality**), τις γεωγραφικές συντεταγμένες του εν λόγω προορισμού υπό τη μορφή πίνακα, `double`, δύο στοιχείων (**coordinates**), καθώς και μια ενδεικτική, μέση τιμή κόστους ημέρας στον εν λόγω προορισμό (**cost**) υπό τη μορφή ακεραίου. Ο τουριστικός προορισμός επίσης διαθέτει τη δική του υλοποίηση για τη μέθοδο **toString()**, σύμφωνα με την οποία επιστρέφει το όνομα του προορισμού.

Ο προορισμός βουνό (**Mountain**) και ο προορισμός θάλασσα (**Sea**), αποτελούν εξειδικεύσεις ενός τουριστικού προορισμού. Στον προορισμό βουνό, μπορούμε να βρούμε ένα σύνολο από δραστηριότητες βουνού (π.χ. αναρρίχηση, πεζοπορία), υπό τη μορφή λίστας αλφαριθμητικών (`ArrayList<String>`) (**activities**). Στον προορισμό βουνό, υπάρχει επιπλέον και το πεδίο υψόμετρο (**altitude**), ως `ακέραιος`. Στη θάλασσα έχουμε ένα σύνολο από αθλήματα που διεξάγονται στην παραλία (π.χ. βόλεϊ, ρακέτες), πάλι υπό τη μορφή λίστας αλφαριθμητικών (`ArrayList<String>`) (**sports**).

Τόσο ο προορισμός βουνό, όσο και ο προορισμός θάλασσα υλοποιούν τη διεπαφή **IPrint**, η οποία διαθέτει τη μοναδική αφηρημένη μέθοδο: **void print()**. Για τον προορισμό βουνό η υλοποίηση της μεθόδου `print()` εκτυπώνει στην οθόνη το σύνολο των διαθέσιμων δραστηριοτήτων βουνού, εφόσον υπάρχουν, ενώ για τον προορισμό θάλασσα η εν λόγω μέθοδος εκτυπώνει στην οθόνη το σύνολο των διαθέσιμων αθλημάτων παραλίας, εφόσον υπάρχουν.

Θέμα 1, ερώτημα Α. (4 μονάδες)

Να υλοποιήσετε τον κώδικα Java (κλάσεις, διεπαφή) που αντιστοιχεί στην παραπάνω περιγραφή.

Θέμα 1, ερώτημα Β. (3 μονάδες)

Να υλοποιήσετε την κλάση **Demo**, στην οποία:

- Θα φτιάξετε τη μέθοδο **main** για την εκτέλεση του προγράμματος. Σε αυτήν θα δημιουργήσετε 2 αντικείμενα τύπου **Mountain** και 2 αντικείμενα τύπου **Sea**, με τις πληροφορίες που θα διαθέτουν (δικής σας επιλογής), τα οποία θα τοποθετήσετε σε αντίστοιχες λίστες `ArrayList<Mountain>` (**mountainList**) και `ArrayList<Sea>` (**seaList**). Από τη λίστα `mountainList` θέλουμε να εκτυπώσετε το όνομα και τις συντεταγμένες του προορισμού με το υψηλότερο υψόμετρο.
- Θα φτιάξετε τη μέθοδο:
List<Destination> findByMunicipality(ArrayList<Destination> input, String municipality). Η μέθοδος αυτή θα δέχεται ως παραμέτρους μια λίστα προορισμών και ένα όνομα δήμου και θα επιστρέφει μια λίστα προορισμών που βρίσκονται στον εν λόγω δήμο.

Θέμα 2. (2 μονάδες). Απαντήστε τις παρακάτω ερωτήσεις πολλαπλής επιλογής.

Η κάθε απάντηση θα πρέπει να συνοδεύεται από αιτιολόγηση για να ληφθεί ως σωστή

1. Θέλουμε να ορίσουμε ένα ανώνυμο αντικείμενο μιας ανώνυμης κλάσης η οποία υλοποιεί τη διεπαφή (interface) `IListen`, χωρίς να μας ενδιαφέρουν οι αφηρημένες μέθοδοι που τυχόν περιέχει. Ποια από τις παρακάτω προτάσεις είναι σωστή;

- a) `IListen listen = new IListen();`
- b) `class NewClass implements IListen{...implemented methods here...}`
- c) `new IListen(){...implemented methods here...};`
- d) Καμία από τις παραπάνω

2. Έστω ο κώδικας της κλάσης `Main`

```
public class Main {  
    static String department = "Unipi";  
    public static void main(String[] args) {  
        handleString(department);  
        System.out.println(department);  
    }  
    static void handleString(String s) {  
        s.toUpperCase();  
        s = s.replace('U', 'O');  
    }  
}
```

Τι θα εκτυπώσει η εκτέλεσή της;

- a) Unipi
- b) UNIPi
- c) Onipi
- d) ONIPi

Απαντήσεις
Θέμα 1, Ερώτημα Α.

```
public interface IPrint {  
    void print();  
}
```

```
public class Destination {  
    String name;  
    String municipality;  
    double[] coordinates = new double[2];  
    int cost;  
  
    @Override  
    public String toString() {  
        return name;  
    }  
}
```

```
public class Mountain extends Destination implements IPrint{  
    ArrayList<String> activities = new ArrayList<>();  
    int altitude;  
  
    @Override  
    public void print() {  
        for (String s: activities)  
            System.out.println(s);  
    }  
}
```

```
public class Sea extends Destination implements IPrint{  
    ArrayList<String> sports = new ArrayList<>();  
    @Override  
    public void print() {  
        for (String s: sports)  
            System.out.println(s);  
    }  
}
```

Θέμα 1, Ερώτημα Β.

```
public class Demo {
    public static void main(String[] args) {
        //Mountains
        Mountain m1 = new Mountain();
        m1.name = "Olympus";
        m1.municipality = "Elassona";
        m1.coordinates[0] = 40.085567;
        m1.coordinates[1] = 22.358521;
        m1.cost = 100;
        m1.altitude = 2918;
        m1.activities.add("Hiking");
        Mountain m2 = new Mountain();
        m2.name = "Taygetus";
        m2.municipality = "Sparta";
        m2.coordinates[0] = 36.95281;
        m2.coordinates[1] = 22.34996;
        m2.cost = 80;
        m2.altitude = 2405;
        m2.activities.add("Climbing");
        //Seas
        Sea s1 = new Sea();
        s1.name = "Voidokilia";
        s1.municipality = "Messini";
        s1.coordinates[0] = 36.9654321671;
        s1.coordinates[1] = 21.6623461246;
        s1.cost = 200;
        s1.sports.add("Beach Volley");
        s1.sports.add("Jet ski");
        Sea s2 = new Sea();
        s2.name = "Myrtos";
        s2.municipality = "Kefalonia";
        s2.coordinates[0] = 38.343052;
        s2.coordinates[1] = 20.535814;
        s2.cost = 180;
        s2.sports.add("Rackets");
        ArrayList<Mountain> mountainList = new ArrayList<>();
        mountainList.add(m1);
        mountainList.add(m2);
        ArrayList<Sea> seaList = new ArrayList<>();
        seaList.add(s1);
        seaList.add(s2);
        //Προορισμός Βουνού με το μεγαλύτερο υψόμετρο
        Mountain max = findMaxAltitude(mountainList);
        if (max!=null)
            System.out.println(max+", "+ Arrays.toString(max.coordinates));
        //Προαιρετικά χρήση της findByMunicipality
        System.out.println(findByMunicipality(seaList, "Messini"));
    }
    static Mountain findMaxAltitude(List<Mountain> mountainList){
        if (mountainList.isEmpty())
            return null;
        int max = mountainList.get(0).altitude;
        Mountain temp = mountainList.get(0);
        for (Mountain m: mountainList){
            if (m.altitude>max){
```

```
        max = m.altitude;
        temp = m;
    }
}
return temp;
}
static List<Destination> findByMunicipality(ArrayList<Sea> input, String
municipality){
    List<Destination> tempList = new ArrayList<>();
    for (Destination d : input){
        if (d.municipality.equals(municipality))
            tempList.add(d);
    }
    return tempList;
}
}
```

Θέμα 2.

1. C
2. A