ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Αντικειμενοστρεφής Προγραμματισμός

Εργαστήριο 12

Μέρος Α: Εμφωλευμένες/Ένθετες κλάσεις (Nested classes)

Εσωτερικές κλάσεις (Inner classes)

Μέρος Β: - Γενικεύσεις (Generics)

Μέρος Α: Εμφωλευμένες/Ένθετες κλάσεις (Nested classes) Εσωτερικές κλάσεις (Inner classes)

12.Α.1. ΕΙΣΑΓΩΓΗ

Η Java επιτρέπει τον ορισμό κλάσης μέσα σε μία άλλη κλάση. Μία τέτοια κλάση ονομάζεται εμφωλευμένη (nested class).

```
class OuterClass {
    ...
    class NestedClass {
        ...
    }
}
```

Η εμφωλευμένη κλάση είναι μέλος της κλάσης που την περιέχει (περιέχουσα/εξωτερική) και ως τέτοιο μέλος, έχει πρόσβαση στα άλλα μέλη της περιέχουσας κλάσης. Ως μέλος της κλάσης OuterClass, μία εμφωλευμένη κλάση μπορεί να δηλωθεί με οιοδήποτε προσδιοριστικό πρόσβασης (private, protected, public ή default).

Οι εμφωλευμένες κλάσεις μπορεί να δηλωθούν ως static ή όχι. Οι μη static εμφωλευμένες κλάσεις ονομάζονται **εσωτερικές κλάσεις(inner classes)**.

```
class OuterClass {
    ...
    static class StaticNestedClass {
        ...
    }
```

```
    class InnerClass {
        ...
    }
}
```

Εμφωλευμένες static κλάσεις

Μία static εμφωλευμένη κλάση (όπως και τα άλλα static μέλη συνδέεται με την εξωτερική κλάση. Δεν μπορούμε από αυτή να χρησιμοποιήσουμε άμεσα αναφορές αντικειμένων ή μεθόδους της περιέχουσας κλάσης – μπορούν να χρησιμοποιηθούν μόνο μέσω αναφοράς αντικειμένου. Αναφερόμαστε σε αυτές μέσω του ονόματος της περιέχουσας κλάσης: `OuterClass.StaticNestedClass`

Πχ. Για να δημιουργήσουμε ένα αντικείμενο μιας εμφωλευμένης static κλάσης:

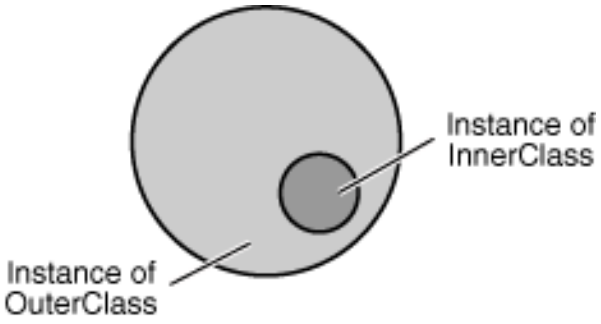
```
OuterClass.StaticNestedClass nestedObject = new
OuterClass.StaticNestedClass();
```

Εσωτερικές (Inner) κλάσεις

Η inner κλάση σχετίζεται με κάποιο αντικείμενο της εξωτερικής κλάσης και έχει άμεση πρόσβαση στα μέλη της. Επίσης επειδή η inner κλάση συνδέεται με αντικείμενα της εξωτερικής, δεν μπορεί να έχει static μέλη.

Τα αντικείμενα μιας inner κλάσης υπάρχουν μέσα σε αντικείμενα της εξωτερικής. classes:

```
class OuterClass {
    ...
    class InnerClass {
        ...
    }
}
```



Για να δημιουργήσετε αντικείμενο της εσωτερικής κλάσης θα πρέπει πρώτα να δημιουργήσετε αντικείμενο της εξωτερικής. Μετά δημιουργείτε αντικείμενο της εσωτερικής ως εξής:

```
OuterClass.InnerClass innerObject = outerObject.new InnerClass();
```

Υπάρχουν δύο ακόμη είδη inner κλάσεων. Μπορούμε να δηλώσουμε μία inner κλάση μέσα σε μία μέθοδο (τοπική κλάση - local inner class). Επίσης μπορούμε να δηλώσουμε μία inner κλάση μέσα σε μία μέθοδο χωρίς να την ονομάσουμε (ανώνυμη εσωτερική κλάση – anonymous class).

12.Α.2. ΑΣΚΗΣΗ

Υλοποιείστε σε Java, τα παρακάτω:

1. Αντιγράψτε τον παρακάτω κώδικα σε ένα αρχείο με όνομα Outer.java. **Μην** μεταγλωττίσετε τον κώδικα.

```
public class Outer {
     private String inString = "Outer inString";
     private String outString = "Outer outString";

     public interface Printable {
          public void print();
     }

     public class Inner {
          private String inString = "Inner inString";
          public String getInString() { return this.inString; }
          public String getOutString() { return outString; }
     }

     public static class StaticMan {
          public void method1() {}
          void method2() {}
          private void method3() {}
          protected void method4() {}
          public String getOutString() {
               String result = "";
               result = outString;
               return result;
          }
     }
}
```

2. Στην κλάση `Outer` ορίστε μία μέθοδο main και ένα δομητή της κλάσης χωρίς ορίσματα (default constructor)
3. Στην μέθοδο **`main`**:
    1. δημιουργείστε ένα αντικείμενο της **`Outer`** με όνομα **`out`**, και ένα αντικείμενο της `Inner` με όνομα `inn`.

2. καλέστε τις μεθόδους **`getInString()`** και **`getOutString()`** για το αντικείμενο **`inn`** και εμφανίστε αυτό που επιστρέφουν. Σημειώστε τι περιμένετε να εμφανιστεί για κάθε μέθοδο (Δεν έχετε ακόμη μεταγλωττίσει την Outer.java)
   `inn.getInString()`: ***"Inner inString"***_____
   `inn.getOutString()`: ***"Outer outString"***____

3. Δημιουργείστε ένα αντικείμενο της κλάσης `StaticMan` με όνομα `sm`. Καλέστε τις μεθόδους **`method1()`**, **`method2()`**, **`method3()`** και **`method4()`** για το **`sm`**. Υπάρχουν κάποιες που δεν θα μεταγλωττιστούν και γιατί; ***Η getOutString δε θα τρέξει επειδή η StaticMan είναι δηλωμένη σαν static, άρα δεν είναι ουσιαστικά inner κλάση της Outer.***

4. Η μέθοδος getOutString() στην εσωτερική κλάση StaticMan δεν μεταγλωττίζεται. Γιατί; ***Επειδή η StaticMan είναι δηλωμένη σαν static.***

5. Τροποποιείστε την getOutString() στην εσωτερική static class StaticMan έτσι ώστε να επιστέφει "" αντί του outString κάνοντας σχόλιο (//) μία από τις 3 γραμμές της μεθόδου
   Τη γραμμή result = outString;

6. Δημιουργείστε μία νέα εσωτερική κλάση με όνομα ConInner, ως εξής:

```
class ConInner {
            public void method() {
               System.out.println( "ConInner: " + outString );
               System.out.println( "ConInner: " + inString );
            }
      }
```

   7. Στον δομητή της Outer, δημιουργείστε ένα αντικείμενο της `ConInner` με όνομα `ci`. Καλέστε την μέθοδο method() για το αντικείμενο `ci`. Τι περιμένετε να εμφανιστεί;

      "ConInner: Outer outString" & "ConInner: Outer inString"_____

      Μπορεί η νέα εσωτερική κλάση `ConInner` να οριστεί στον δομητή της Outer ή στην μέθοδο main; ***Μπορεί και στα δυο.***

   8. Στον δομητή της Outer, δημιουργείστε μία ανώνυμη κλάση η οποία υλοποιεί την διασύνδεση Printable. Σε αυτή την κλάση υπερβείτε την μέθοδο print() εμφανίζοντας το όνομά σας ***("Panos")***. Δημιουργείστε ένα αντικείμενο αυτής της κλάσης με όνομα p, και καλέστε την μέθοδο print για το p:
   9. Μεταγλωττίστε και τρέξτε το πρόγραμμά σας.

Μέρος Β: - Γενικεύσεις/Γενικοί Τύποι (Generics)

Υλοποιείστε σε Java, τα παρακάτω:

2. Τη γενική/παραμετρική κλάση SetClass με παράμετρο τύπου T. Η κλάση διαθέτει έναν πίνακα N στοιχείων τύπου T (αρχικά NULL) και τις παρακάτω λειτουργίες:
   a. `void insert(T elem)`: εισάγει ένα στοιχείο στον πίνακα. Το στοιχείο εισάγεται στην πρώτη κενή θέση του πίνακα. Δεν ελέγχετε αν το στοιχείο έχει ήδη τοποθετηθεί στον πίνακα ή αν ο πίνακας είναι γεμάτος.
   b. `delete(T elem)`: διαγράφει το στοιχείο του πίνακα που ισούται με τη παράμετρο της μεθόδου `elem`. Χρησιμοποιείστε τη μέθοδο `equals()` για να κρίνετε την ισότητα των στοιχείων.
   c. `int find(T elem)`: επιστρέφει τη θέση του πρώτου στοιχείου του πίνακα που ισούται (χρήση `equals()`) με το στοιχείο το οποίο δίνεται ως παράμετρος.

   Ελέγξτε τον κώδικά σας χρησιμοποιώντας ένα πίνακα με String και ένα πίνακα με Integers με δύο διαφορετικά αντικείμενα SetClass. Στο ένα θα χρησιμοποιήσετε τον πίνακα String και στο άλλο τον πίνακα Integer. Μπορείτε να εισάγετε ένα Integer στον πίνακα με τα String; Εξηγείστε τι συμβαίνει, όταν προσπαθήσετε να το κάνετε.

3. Δηλώστε την εξαίρεση `ArrayOverflow` την οποία δημιουργεί/ρίχνει η μέθοδος insert όταν ο πίνακας είναι γεμάτος. Γράψτε ένα try/catch block για να ελέγξετε την εξαίρεση. Η εξαίρεση πρέπει να επεκτείνει την `Exception` και όχι την `RuntimeException`.

4. Γράψτε τη μέθοδο `equals` η οποία συγκρίνει δύο πίνακες τύπου T για ισότητα. Δύο πίνακες είναι ίσοι όταν περιέχουν το ίδιο πλήθος ίσων στοιχείων -χρήση `equals()`- με την ίδια σειρά. Προσοχή μπορεί ενδιάμεσα να υπάρχουν κενά στοιχεία.
   Μπορείτε να χρησιμοποιήσετε την "`instanceof T`" στην μέθοδο `equals`; Γιατί μπορείτε ή γιατί δεν μπορείτε;