ΕΡΓΑΣΤΗΡΙΟ 7
Garbage Collector

1. Μελετήστε το παρακάτω πρόγραμμα Java. Πόσα αντικείμενα δεσμεύουν χώρο στη μνήμη; Τι γίνεται στο παρασκήνιο;

```
public class DynamicStringConcatenation {
    public static void main(String[] args) {
        String result = "";
        for (int i = 0; i < 1e6; i++) {
            result += "some more data";
        }
        System.out.println(result);
    }
}
```

Χρησιμοποιήστε το εργαλείο javap για να μπορείτε να εξηγήσετε.

2. Μελετήστε το παρακάτω πρόγραμμα Java.

```
class HappyGarbage01 {
    public static void main(String args[])
    {
        HappyGarbage01 h = new HappyGarbage01();
        h.methodA(); /* Line 6 */
    }
    Object methodA()
    {
        Object obj1 = new Object();
        Object [] obj2 = new Object[1];
        obj2[0] = obj1;
        obj1 = null;
        return obj2[0];
    }
}
```

Πότε καλείται ο garbage collector;

α. Μετά τη γραμμή 9 του κώδικα
β. Μετά τη γραμμή 10 του κώδικα
γ. Μετά τη γραμμή 11 του κώδικα
δ. Δεν καλείται ποτέ στη μέθοδο `methodA()`

3. Μελετήστε το παρακάτω πρόγραμμα Java. Μετά τρέξτε το πρόγραμμα και δείτε τα αποτελέσματα. Τι παρατηρείτε;

```
public class FreeMemory {

   public static void main(String[] arg) {
       int[][] table = new int[3][];
       int i;

       System.out.println("The maximum : " + Runtime.getRuntime().maxMemory());
       System.out.println("The total   : "  + Runtime.getRuntime().totalMemory());
       System.out.println("Free        : " + Runtime.getRuntime().freeMemory());
       System.out.println();
       for (i = 0; i < 3; i++) {
         table[i] = new int[400000];
         System.out.println("i = " + i + ", free : " + Runtime.getRuntime().freeMemory());
       }
       System.gc();
       System.out.println();
       System.out.println("After deallocation, free : " + Runtime.getRuntime().freeMemory());
       System.out.println();
       for (i = 0; i < 3; i++) {
         table[i] = null;
         System.gc();
         System.out.println("i = " + i + ", free : " + Runtime.getRuntime().freeMemory());
       }
   }
}
```

4. Τι θα εκτυπώσει το παρακάτω πρόγραμμα. Πώς γίνεται η εκτέλεση του προγράμματος και η δέσμευση στη μνήμη;

```
class StaticComponents {
      static int staticVariable;

      static {
            System.out.println("StaticComponents SIB");
            staticVariable = 10;
      }

      static void staticMethod() {
            System.out.println("From StaticMethod");
            System.out.println(staticVariable);
      }
}

public class MainClass {
      static {
            System.out.println("MainClass SIB");
      }

      public static void main(String[] args) {
          //Static Members directly accessed with Class Name
            StaticComponents.staticVariable = 20;
            StaticComponents.staticMethod();
      }
}
```

5. Τι θα εκτυπώσει το παρακάτω πρόγραμμα. Πώς γίνεται η εκτέλεση του προγράμματος και η δέσμευση στη μνήμη;

```
class A {
     int nonStaticVariable;
     static int staticVariable;

     static void staticMethod() {
          System.out.println(staticVariable);
     //   System.out.println(nonStaticVariable);
     }

     void nonStaticMethod() {
          System.out.println(staticVariable);
          System.out.println(nonStaticVariable);
     }
}

class MainClass {
     public static void main(String[] args) {
          A.staticVariable = 10;
     //   A.nonStaticVariable = 10;
          A.staticMethod();
    //    A.nonStaticMethod();

          A a1 = new A();
          A a2 = new A();

          System.out.println(a1.nonStaticVariable);
          System.out.println(a1.staticVariable);
          a1.nonStaticMethod();
          a1.staticMethod();

          System.out.println(a2.staticVariable);
          a1.staticVariable = 20;
          System.out.println(a2.staticVariable);
     }
}
```

6. Τι θα εκτυπώσει το παρακάτω πρόγραμμα; Γιατί;

```
public class Test2 {
    Test2(int x) {
           System.out.println("ONE argument constructor");
    }

    Test2() {
           System.out.println("No argument constructor");
    }

    static {
           System.out.println("1st static init");
    }

    {
           System.out.println("1st instance init");
    }

    {
           System.out.println("2nd instance init");
    }

    static {
           System.out.println("2nd static init");
    }

    public static void main(String[] args) {
           new Test2();
           new Test2(8);
    }
}
```