```
Faculty of Exact Sciences                      examination Programming
Vrije Universiteit                 February 2nd, 2016 time: 2:45 hours
----------------------------------------------------------------------
```

Problem 1.

a) Let the classes A and B be

```
        class A {
            int g;
            B b;

            A (B b) {
                this.g = b.g - 2;
                this.b = b;
            }

            void subtract () {
                g -= 1;
            }
        }

        class B {
            int g;
            A a;

            B (int g) {
                this.g = g;
                this.a = null;
            }

            void add () {
                g += 1;
            }
        }
```

Further we have a program with the following statements/declarations

```
        B b = new B(6);
        A a1 = new A(b);
        PrintStream out = new PrintStream(System.out);

        void println (A x, B y) {
            out.printf("%d - %d - %d\n", x.g, x.b.g, y.g);
        }

        void doSomething () {
            println(a1, b);
            a1.subtract();
            println(a1, a1.b);
            A a2 = new A(a1.b);
            println(a2, b);
            b.a = a2;
            println(b.a, a1.b);
            a1.b.add();
            a2.b.add();
            println(b.a, b);
            a2.subtract();
            println(b.a, a1.b);
            b.a = a1;
            println(a2, a1.b);
            b.a.subtract();
            println(a1, b.a.b);
        }
```

What will be printed when the method doSomething() is called?

b) The following heading of a method calculate() is given

```
double calculate (double x, int numberOfTerms)
```

This method should calculate numberOfTerms terms of the series

```
x^0/(2*0!) - x^1/(3*2!) + x^2/(4*4!) - x^3/(5*6!) + x^4/(6*8!) - ....
```

x^n is the notation for "x to the power n" and n! is the notation for "the factorial of n". The number of terms that should be calculated is given by numberOfTerms. Implement this method. Do this without using any methods from the class Math. Assume: numberOfTerms≥1.

c) Give the declaration of a variable "matrix" with as type a 2-dimensional array of char's with 4 rows and 8 columns. Use constants when necessary.

Program a method oddBigger() that will be able, for any 2-dimensional array of int's, to check whether this array has the property that the sum of all the odd numbers in that array is bigger than the sum of all the even numbers in that array.

examples	1 3 2 5	has the	2 8 4 1	doesn’t have
	7 9 4 1	property	6 2 8 5	the property
	3 5 6 9	oddBigger	8 4 2 9	oddBigger

d)

```
  class Problem_1d {
    PrintStream out;
    int e, f;

    Problem_1d() {
        out = new PrintStream(System.out);
        e = -3;
        f = 4;
    }

    void print (int x, int y) {
        out.printf("%d %d\n", x, y);
    }

    int m1 (int d, int e) {
        e *= 2;
        int f = d + e;
        print(e, f);
        return d+1;
    }

    int m2 (int e) {
        print (e, f);
        e = m1(e, 3);
        print(e, f);
        f += 3;
        return e-f;
    }

    void start() {
        print (e, f);
        f = m1(f, e);
        print(e, f);
        e = m2(f);
        print(e, f);
    }

    public static void main(String argv[]) {
        new Problem_1d().start();
    }
}
```

What will be printed when this program is executed?

Problem 2.

a) Given are the following classes:

```
class Book {
    String title;
    boolean lentOut;
    int numberOfPages;
}

class Bookcase {
    String genre;
    Book[] bookArray; // All elements of this array have a value.
}
```

In both classes the constructors and the methods are omitted, as they are not necessary for this problem.
Make a class Library. This class should be able to store a maximum of 50 bookcases. Further the class should have a default constructor and a method add(). The default constructor should initialize the Library-object to an empty library. The method add() should make it possible to add 1 bookcase to the library.

b) Program in the class Library a method

```
    Library popularGenres ()
```

which, in a new Library-object, returns all the bookcases that contain popular genres. For this problem the genre of a bookcase is considered to be popular if more than 100 books are lent out.

Program sub problems in separate methods in the correct class. Use constants when necessary.

c) Program in the class Library a method

```
    int bigBookGenres ()
```

which returns the number of bookcases whose genre is considered to be a genre with a lot of big books. For this problem a book is defined to be a big book if the number of pages of that book is more than 500. Also for this problem, a genre is defined to be a big book genre if 22.5% (or more) of the books in that genre is a big book.

Program sub problems in separate methods in the correct class. Use constants when necessary.

d) Program in the class Library a method

```
    int bigPopularBookGenres ()
```

This method should return the number of bookcases with popular genres that contain a lot of big books. Program the method bigPopularBookGenres () without using a while-statement, a for-statement or a do-while-statement.

Problem 3.

a) Given is that the class String contains a method

```
String substring (int start)
```

that calculates the substring from the character on index position start till the last character (inclusive).

```
examples:   "abcdef".substring(3) gives "def"
            "abcdef".substring(0) gives "abcdef"
            "abcdef".substring(6) gives ""
```

Write a recursive method "String omit (String s)" that, given a string, removes all characters with an uneven index.

```
examples:
         ""         →    ""
         "a"        →    "a"
         "ab"       →    "a"
         "abc"      →    "ac"
         "abcd"     →    "ac"
         "abcde"    →    "ace"
```

b) Write a recursive solution for the method

```
String clean (String s)
```

that replaces in all substrings of n (n>=2) or more adjacent characters that are the same, the first n-1 characters with a dash.

```
examples:
         ""              →    ""
         "a"             →    "a"
         "aa"            →    "-a"
         "aaabbac"       →    "--a-bac"
         "XXyhsssertts"  →    "-Xyh--ser-ts"
         "qqqqq"         →    "----q"
```

grade:

Problem	a	b	c	d	total
1.	4	6	5	4	19
2.	3	6	7	2	18
3.	4	4			8
					-- +
					45

The grade E follows from the points P with the formula: E = P / 5 + 1