```
Faculty of Exact Sciences                      examination Programming
Vrije Universiteit              December 14th, 2015 time: 2:45 hours
----------------------------------------------------------------------
```

Problem 1.

a) Let the classes A and B be

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

            A (int g) {
                this.g = g;
                this.b = null;
            }

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

        class B {
            int g;
            A a;

            B (A a) {
                this.g = 3 * a.g;
                this.a = a;
            }

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

Further we have a program with the following statements/declarations

```
        A a = new A(1);
        B b1 = new B(a);
        PrintStream out = new PrintStream(System.out);

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

        void doSomething () {
            println(a, b1);
            a.add();
            println(b1.a, b1);
            B b2 = new B(b1.a);
            println(a, b1);
            a.b = b2;
            println(b2.a, b2);
            b1.a.add();
            b2.a.add();
            println(a, a.b);
            b2.subtract();
            println(b2.a, b1);
            a.b = b1;
            println(b2.a, b1);
            a.b.subtract();
            println(a, b1);
        }
```

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

```
(5*x^2)/1! – (6*x^3)/2! + (7*x^4)/3! – (8*x^5)/4! + (9*x^6)/5! - ....
```

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 9 rows and 7 columns. Use constants when necessary.

Program a method shift() that will be able, for any 2-dimensional array of int's, to shift the int’s in every row. If the number of columns of the rows equals k, the int’s in the last k-1 columns have to be shifted one to the left and the int in the first column has to be shifted to the last column.

```
examples  after shifting    1  2  3  4   the result   2  3  4  1
          all the rows in   5  6  7  8   should be    6  7  8  5
          the matrix        9 10 11 12               10 11 12  9
                           13 14 15 16               14 15 16 13
```

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

    Problem_1d() {
        out = new PrintStream(System.out);
        e = -5;
        f = 6;
    }

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

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

    int m2 (int e) {
        print (e, f);
        e = m1(2, e);
        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 Course {
    String name;
    double grade;
}

class Student {
    String name;
    int year; // number of years registered at the university
    Course[] results; // 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 University. This class should be able to store a maximum of 4000 students. Further the class should have a default constructor and a method add(). The default constructor should initialize the University-object to an empty university. The method add() should make it possible to add 1 student to the university.

b) Program in the class University a method

```
University topStudents ()
```

which, in a new University-object, returns all the students that are a top student. For this problem a student is considered to be a top student if the average of the grades is 9.0 or higher.

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

c) Program in the class University a method

```
int doingOK ()
```

which returns the number of students that are doing OK in their study. For this problem it is defined that a student is doing OK if 95% (or more) of all the grades is a passed grade. A grade is a passed grade when that grade is >= 5.5

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

d) Given is that the class University contains a method

```
University registered (int years)
```

This method returns, in a new University-object, all the students that have already been registered for ‘years’ years. Since the method registered() is given, it can be used without having to program it.

Now program in the class University a method

```
int doingOkAndRegistered (int years)
```

This method should return the number of students that are doing Ok and have been registered for ‘years’ years. Program the method doingOkAndRegistered() without using a while-statement, a for-statement or a do-while-statement.

Problem 3.

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"
```

a) Write a recursive method "String reverse (String s)" that, given a string, returns a new string where the characters are now in reverse order.

examples:

""	→	""
"a"	→	"a"
"ab"	→	"ba"
"abc"	→	"cba"
"abcd"	→	"dcba"

b) Write a recursive solution for the method

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

that reduces all adjacent characters that are the same to a single character.

examples:

""	→	""
"a"	→	"a"
"aa"	→	"a"
"aaabbac"	→	"abac"
"XXyhsssertts"	→	"Xyhserts"

grade:

Problem	a	b	c	d	total
1.	4	4	4	4	16
2.	2	4	4	2	12
3.	4	4			8
					-- +
					36

The grade E follows from the points P with the formula: E = P / 4 + 1