```
Faculteit der Betawetenschappen    exam Introduction to Programming (Java)
Vrije Universiteit                         February 4th, 2019 time 2:45 hours
--------------------------------------------------------------------------
```

Problem 1.

a) Let the classes D, E and F be

```
class D {
    int g;
    E e;

    D (int g) {
        this.g = g;
        e = new E(4);
    }
}

class E {
    int g;
    F f;

    E (int g) {
        this.g = g;
        f = new F(6);
    }
}

class F {
    int g;

    F (int g) {
        this.g = g;
    }

    void add (int x) {
        g += x;
    }
}
```

Further we have a program with the following statements/declarations

```
D d = new D(-1);
E e = new E(0);
F f = new F(1);
PrintStream out = new PrintStream(System.out);

void println (D x, F y) {
    out.printf("%d - %d\n", x.e.f.g, y.g);
}

void doSomething () {
    println(d, f);
    e.f.g = 8;
    println(d, f);
    f = d.e.f;
    f.add(4);
    println(d, f);
    d.e = e;
    println(d, f);
    e.f.add(-6);
    println(d, f);
    f = e.f;
    f.add91);
    println(d, f);
}
```

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 n)
```

This method calculates the first n terms of the series

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

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 the parameter n. Implement this method. Do this without using any methods from the class Math. Assume: n≥1.

c) Give the declaration of a variable "matrix" with as type a 2-dimensional array of char's with 7 rows and 2 columns. Use constants when necessary.

Program a method twos() that will be able, for any 2-dimensional array of int's, to check if that array has the property twos. A 2-dimensional array has the property twos if there are never two adjecent 2’s in the array.

examples
```
      1 2 2   adjacent     1 2 1 1   adjacent         1 3 4   no adjecent
      7 9 4   2’s in       9 2 1 1   2’s in           5 2 6   2’s
      3 5 6   first row    1 1 9 1   second column    7 8 9
```

d)
```
class Problem_1d {
    PrintStream out;
    int t, u;

    Problem_1d() {
        out = new PrintStream(System.out);
        t = -3;
        u = -5;
    }

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

    int m1 (int t) {
        t += 1;
        u *= 2;
        int v = t + u;
        print(u, v);
        return t;
    }

    int m2 (int u, int v) {
        t = u - t;
        v = m1(t);
        print(v, u);
        u += 1;
        return u + v;
    }

    void start() {
        print (t, u);
        int v = m1(u);
        print(v, t);
        v = m2(t, v);
        print(t, v);
    }

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

What will be printed when this program is executed?

Problem 2.

For every subproblem of problem 2, program that subproblem in separate method in the correct class. Use constants when necessary.

a) Given is the following class:

```
class Painting {
    String style,
    int value,
        yearPainted;
}
```

The constructor and the methods are omitted, as they are not necessary for this problem.

Make a class Museum. This class should be able to store a maximum of 1300 paintings. Further the class should have a default constructor and a method add(). The default constructor should initialize the Museum-object to an empty museum. The method add() should make it possible to add 1 painting to the museum.

b) Program in the class Museum a method

```
Museum oldMasterWorks ()
```

which, in a new Museum-object, returns all the paintings that are old master works. For this problem it is defined that a painting is an old master work if its value is not less than 500000 and it was painted before 1803.

c) Program in the class Museum a method

```
void removeStyle (String s)
```

which removes all paintings of the style s from the museum.

d) Given is that the class Museum contains a method

```
int expensive ()
```

that returns in an int the total number of paintings in the museum that are expensive. You can use this method without having to program it.

Now add to the class Museum a method

```
int expensiveOldMasterWorks (String s)
```

which returns the total number of paintings in the museum that are not painted in the style s and that are expensive old master works. Program this method without using a for or while statement.

Problem 3.

a) Write a recursive method

```
int maximum (int[] r, int i) // 0 <= i <= r.length-1 && r.length > 0
```

that searches the array r, starting from index position i, and returns the maximum found among the searched elements.

```
examples: assume the array r contains {1, 2, 3, 4, 3, 2, 3, 2, 1}
            maximum(r, 0) gives 4
            maximum(r, 2) gives 4
            maximum(r, 4) gives 3
            maximum(r, 6) gives 3
            maximum(r, 8) gives 1
```

b) Given is that the class String contains a method

```
String substring (int start) // 0<=start<=length string
```

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

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

Write a recursive solution for the method

```
int countPairs (String s) // s.length() >= 0
```

that counts the number of pairs in s. A pair consists out of two identical character, seperated by a third character that differs from these two characters.

```
examples:   count("aba")            gives 1
            count("aaa")            gives 0
            count("bBbB")           gives 2
            count("ababab")         gives 4
            count("abaaaabbbab")    gives 2
            count("")               gives 0
```

grade:

Problem	a	b	c	d	total
1.	5	5	5	5	20
2.	2	6	6	3	17
3.	4	4			8
					-- +
					45

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