---

Problem 1.

a) Let the classes A and B be

```
class A {
    char c;
    B b;

    A (B b) {
        c = 'A';
        this.b = b;
    }

    void next () {
        c = (char)(c + 1);
    }
}

class B {
    char c;
    A a;

    B (A a) {
        c = 'Z';
        this.a = a;
    }

    void prior () {
        c = (char) (c – 1);
    }
}
```

Further we have a program with the following statements/declarations

```
A a = new A(null);
B b = new B(a);
a = new A(b);
PrintStream out = new PrintStream(System.out);

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

void doSomething () {
    println(a, b);
    a.next();
    println(a, b);
    b.prior();
    println(a, b);
    a.b.prior();
    println(a, b);
    a = b.a;
    println(a, b);
    a.b = new B(a);
    println(b.a, a.b);
    b.a.next();
    a.b.prior();
    println(a, b);
}
```

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

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

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

This method should calculate $e^x$ in numberOfTerms terms.

e^x = 1 + x^1/1! + x^2/2! + x^3/3! + x^4/4! + ......

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 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 7 rows and 4 columns. Use constants when necessary.

Program a boolean method special() which will be able to tell, for any 2-dimensional array of int's, whether this array has the property "special". For this problem a 2-dimensional array with x columns has the property special when the first row contains the numbers 1, 2, 3, .., x; the second row contains the numbers 2, 4, 6, ..., x*2 and in general the $i^{th}$ row contains the number 1*i, 2*i, 3*i, ..., x*i.

Examples: array 1 2 3 4 and 1 2 are special and 1 2 is not
2 4 6 8 2 4 3 6
3 6 9 12 3 6 4 8

d)
```
class Problem_1d {
    PrintStream out;
    int r, s;

    Problem_1d() {
        out = new PrintStream(System.out);
        r = -7;
        s = 11;
    }

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

    int method_a (int p) {
        int s = 4 - p;
        p = this.s / 2;
        r = p + 2;
        print(r, s);
        return p+s;
    }

    int method_b (int p) {
        int r = p + 1;
        s = s - r;
        this.r += 1;
        print(r, s);
        return r;
    }

    void start() {
        r = method_a(r);
        print(r, s);
        s = method_b(method_a(4));
        print(r, s);
    }

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

```
}
```
What will be printed when this program is executed?

Problem 2.

a) Using the following class

```
class PileOfPotatoes {
    String species;
    double starchContent; //percentage
    int monthsInStorage;

    // The constructors en methods are omitted as they are not
    // necessary for this problem.
}
```

construct a class PotatoDistributionCentre. This class should be able to store a maximum of 200 piles of potatoes. Further the class should have a default constructor and a method add(). The default constructor should initialize the PotatoDistributionCentre-object as an empty potato distribution centre. The method add() should make it possible to add 1 pile of potatoes to the potato distribution centre.

b) Program a method

```
PotatoDistributionCentre floury ()
```

which, in a new PotatoDistributionCentre-object, returns all the piles of potatoes that contain floury potatoes. For this problem a potato is defined as floury when the starch content is more than 19.5%.

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

c) The following method can be assumed to be present in the class PotatoDistributionCentre. It can be used without having to program it.

```
PotatoDistributionCentre species (String s)
```

This method returns all the piles of potatoes from the specified species.

Now add to the class PotatoDistributionCentre a method

```
PotatoDistributionCentre flourySpecies (String s)
```

This method should return all the piles of potatoes that are floury and are of species s. Program the method flourySpecies() without using a while-statement, a for-statement or a do-while-statement.

d) Add to the class PotatoDistributionCentre a method

```
void cleanUp ()
```

This method should clean up the potato distribution centre by removing all the unsellable piles of potatoes. For this problem a pile of potatoes is considered unsellable when it has been in the potato distribution centre for more than 2 years.

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

Problem 3.

a) Given is that the class String contains a method

```
public 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"

Write a recursive method "String dashes (String s)" that, given a string, returns a new string where all the adjacent characters are now separated by a "-".

examples:

| | | |
|---|---|---|
| "" | ␣ | "" |
| "a" | ␣ | "a" |
| "ab" | ␣ | "a-b" |
| "abc" | ␣ | "a-b-c" |
| "abcd" | ␣ | "a-b-c-d" |

b) Given is the following method

```
void swap (char[] r, int i, int j)
```

that in the array r switches the elements with indices i and j.

Example:
```
char[] word = {'a', 'c', 'b'};
swap(word, 1, 2);
// the content of word is now {'a', 'b', 'c'};
```

and a method

```
void println(char[] r)
```

that prints the characters of r on the output.

Now write a recursive solution for the method

```
void permutations(char[] r, int length)
```

that will print all permutations of the characters in the array r. The parameter length represents the number of elements of the array that still has to be permutated.

Example, if r is {'A', 'B', 'C'} then the output should contain

```
ABC
ACB
BAC
BCA
CAB
CBA
```

grades:

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

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