
Solutions

Opgave 1.

a)
```
A, A, A
B, B, A
C, C, A
K, C, A
K, A, K
L, A, L
M, A, M
```

b)
```
double sine (double x, int numberOfTerms) {
    int sign = 1,
        factorial = 1;
    double power = x,
           result = sign * power/factorial;

    for (int i = 1; i < numberOfTerms; i++) {
        sign = -sign;
        power *= x * x;
        factorial *= (2*i) * (2*i+1);
        result += sign * power/factorial;
    }

    return result;
}
```

c)
```
static final int NUMBER_OF_ROWS = 17,
                 NUMBER_OF_COLUMNS = 29;
char[][] matrix = new char[NUMBER_OF_ROWS][ NUMBER_OF_COLUMNS];

boolean nice (int[][] m) {
    int sum = 0;

    for (int i = 0; i < m.length; i++) {
        for (int j = 0; j < m[0].length; j++) {
            sum += m[i][j];
        }
    }

    return sum == m.length*m[0].length;
}
```

d)
```
-2, -5
-5, -3
-5, 10
5, 2
2, 0
```

Opgave 2.

a)
```
class AddressBook {
    static final int MAX_NUMBER_OF_CONTACTS = 2500;

    Contact[] contactArray;
    numberOfContacts;

    AddressBook () {
        contactArray = new Contact[MAX_ NUMBER_OF_CONTACTS];
        numberOfContacts = 0;
    }

    void add (Contact contact) {
        contactArray[numberOfContacts] = contact;
        numberOfContacts += 1;
    }
}
```

b)
```
add to the class Contact

boolean closeFamily () {
    return family and
           (telephoneNumber.charAt(0)!='0' or telephoneNumber.charAt(1)!='0');
}


add to the class AddressBook

AddressBook closeFamily () {
    AddressBook result = new AddressBook();
    for (int i = 0; i < numberOfContacts; i++) {
        if (contactArray[i].closeFamily()) {
            result(add(contactArray[i]));
        }
    }
    return result;
}
```

c)
```
AddressBook olderCloseFamily (int age) {
    return older(age).closeFamily();
}
```

d)
```
add to the class Contact

static final int NOT_NEEDED_AGE = 42;
static final String NOT_NEEDED_CITY = "Answer";

boolean notNeeded () {
    return city.equals(NOT_NEEDED_CITY) and age == NOT_NEEDED_AGE;
}

void cleanUp () {
    for (int i = 0; i < numberofContacts; i++( {
        if (contactArray[i].notNeeded()) {
            numberOfContacts -= 1;
            contactArray[i] = contactArray[numberOfContacts];
            i -= 1;
        }
    }
}
```

Opgave 3.

a)
```
int product (int n) {
    if (n == 1) {
        return 1;
    }

    return product(n-1) * n * product(n-1);
}
```

b)
```
int numberOfPairs (String s) {
    if (s.length() < 3) {
        return 0;
    }

    char firstChar = str.charAt(0),
         thirdChar = str.charAt(2);
    String rest = str.substring(1);

    return (firstChar == thirdChar ? 1 : 0) + numberOfPairs(rest);
}
```