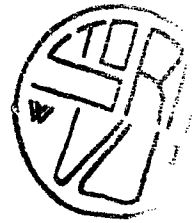
Faculteit der Exacte Wetenschappen — tentamen Inleiding Programmeren
Vrije Universiteit — 22 maart 200[illegible] ijdsduur: 3 uur

- N.B. MNW-studenten maken de opgaven 1 en 2 en krijgen hiervoor 2 uur de tijd -

Opgave 1.

a) Gegeven zijn de onderstaande classes



```
class A {
    String dataA;

    A (String s) {
        dataA = s;
    }

    void concat (String s) {
        dataA += s;
    }
}

class B {
    A a;
    String dataB;

    B (String s) {
        a = new A(s);
        dataB = s;
    }

    void put (String s) {
        a.concat(":" + s);
        dataB = s;
    }
}
```

en een programma dat de classes A en B gebruikt en waarin de volgende declaraties staan

```
Output out = new Output();
A a = new A(".");
B b = new B("xxx");

void print () {
    out.println(a.dataA);
    out.println(b.dataB);
    out.println(b.a.dataA);
}

void doeIets () {
    print();
    b.put("yyy");
    print();
    a.concat(":zzz");
    print();
    b.a = a;
    b.put("qqq");
    print();
}
```

Wat is de uitvoer van een aanroep van de methode doeIets()?

b) Schrijf voor n>=1 de methode : double sommeerTermen (int x, int n) die de sommatie uitrekent van de eerste n termen van de onderstaande in wiskunde genoteerde berekening

-1/2*x^3 + 2/4*x^4 - 3/8*x^5 + 4/16*x^6 - 5/32x^7 - .....

Hierbij staat de notatie a^b voor a tot de macht b.
Het is niet toegestaan om methodes uit de class Math te gebruiken.

c) Schrijf een methode meerPositiefDanNegatief() die voor een willekeurig 2-dimensionaal arrays van integers (matrix) via een boolean functieresultaat antwoord geeft op de vraag of in iedere kolom meer positieve getallen dan negatieve getallen staan (N.B. het getal 0 is zowel niet negatief als niet positief).

Bij de linker onderstaande matrix is het antwoord true. Bij de twee matrices rechts daarvan is het antwoord false. De meest rechtse matrix levert false op omdat er in de eerste kolom zowel 0 positieve als 0 negatieve getallen zitten waardoor het aantal getallen > 0 in die kolom gelijk is aan het aantal getallen < 0 (en dus niet groter).

voorbeelden:

```
 1  1  1        -1 2 3        0 1 2
 2 -2  2        -1 2 3        0 1 2
 3  3 -3         1 2 3        0 1 2
```

d) Geef de uitvoer van het onderstaande programma:

```
class Opgave1d {
    Output out;
    int x, y;

    Opgave1d() {
        out = new Output();
        x = 2;
        y = 3;
    }

    void print (int p, int q) {
        out.println(p);
        out.println(q);
    }

    void m1(int y) {
        x = 2 * y;
        y = 2 * x;
    }

    int m2 (int a, int b, int c) {
        a = x / y;
        b = a + 1;
        c = a + b;
        print(a, c);
        m1(c);
        print(x, y);
        return b;
    }

    void start() {
        print(x, y);
        m1(x);
        print(x, y);
        y = m2(y, x, x+2);
        print(x, y);
    }

    public static void main(String argv[]) {
        new Opgave1d().start();
    }
}
```

2.0.2.

a) Gegeven zijn de onderstaande classes welke gebruikt kunnen worden voor het opslaan van gegevens van studenten.

```
class Student {

    String naam;
    double ectsPerJaar;
    int aantalJaarIngeschreven.

    // De constructors en methodes doen er voor deze opgave niet toe.
}

class Faculteit {

    static final int MAX_AANTAL_STUDENTEN = 1500;

    Student[] studentArray;
    int aantalStudenten;

    ...

}
```

Schrijf een default constructor en een methode voegToe() voor de class Faculteit. De default constructor moet het Faculteit-object initialiseren op een lege faculteit met 0 studenten waaraan maximaal 1500 studenten kunne worden toegevoegd. De methode voegToe moet de mogelijkhied bieden om 1 Student-object aan een faculteit-object toe te voegen.

b) Voorzie de class Faculteit van een methode

```
int aantalGoedeStudenten ()
```

welke retourneert hoeveel studenten van de faculteit goed studeren. Voor deze opgave is gedefinieerd dat een student goed studeert als het aantal ects per jaar van die student groter is dan 55.

Indien je een deelprobleem herkent, programmeer dit dan in een aparte methode in de juiste class.

c) Voeg aan de class faculteit een methode selectie() toe die, in een nieuw Faculteit object, alle topstudenten retourneert. Voor deze opgave is een topstudent gedefinieerd als een goede student die minimaal 4 jaar is ingeschreven.

Indien je een deelprobleem herkent, programmeer dit dan in een aparte methode in de juiste class.

Opgave 3.

a) Schrijf een recursieve methode

```
int over (int n, int k) // gegeven n >= k >= 0
```

voor de berekening van over(n, k). De formule over(n, k) is als volgt gedefinieerd

$$over(n,k) = \begin{cases} over(n-1,k) * over(n-1,k-1) & \text{als } n>k>=1 \\ 1 & \text{als } n=k \ (n,k>0) \\ 1 & \text{als } k=0 \end{cases}$$

b) Gegeven de onderstaande hulpmethode

```
String aantalKeer (char c, int x) {
    String resultaat = "";
    for (int i = 0; i < x; i++) {
       resultaat += c;
    }
    return resultaat;
}
```

Geef een recursieve oplossing voor de onderstaande methode

```
String voegHaakjesToe (String s, int i)
// s bevat alleen cijfers, 0 <= i <= s.length()
```

die voor een String die alleen cijfers bevat, voor alle cijfers vanaf het cijfer op de positie i, ronde haakjes openen en sluiten toevoegt. Het aantal haakjes openen en sluiten is gelijk aan de waarde van het cijfer (bij het cijfer 6 worden er dus 6 ronde haakjes openen en 6 ronde haakjes sluiten toegevoegd). Alle ronde haakjes sluiten komen aan het eind van de resultaatstring.

Voorbeelden:

de aanroep		levert
voegHaakjesToe("0", 0)		"0"
voeghaakjesToe("1", 0		"(1)"
voeghaakjesToe("2", 0)		"((2))"
voeghaakjesToe("12", 0)		"(1((2)))"
voeghaakjesToe("231", 0)		"((2(((3(1)))))))"
voeghaakjesToe("231", 1)		"(((3(1))))"
voeghaakjesToe("231", 2)		"(1)"
voeghaakjesToe("231", 3)		""
voeghaakjesToe("40", 0)		"((((40))))"

N.B. De waarde van de variabele i na de assignment

```
int i = '5' - '0';
```

is 5.

Waardering

Opgave	a	b	c	d	totaal
1.	4	4	4	4	16
2.	2	4	6		12
3.	3	5			8
					-- +
					36

Voor MNW-studenten:
Het eindcijfer E volgt uit het puntentotaal T als volgt : E = T*9/28 + 1

Voor alle andere studenten:
Het eindcijfer E volgt uit het puntentotaal T als volgt : E = T / 4 + 1