

Opgave 1.

a) Gegeven zijn de class

```
class X {
    String s;
    char c;

    X () {
        s = new String();
        c = 'x';
    }

    X (String string) {
        s = string;
        c = 'x';
    }

    void insert (char ch) {
        s = s + ch;
        c = ch;
    }
}
```

en de declaraties

```
Output out = new Output();
X xObject = new X("opgave 1a");

void m1 (X x) {
    x.s = x.s + "1";
    x.c = '1';
    out.println(xObject.c);
    out.println(xObject.s);
    x.insert('?');
    m2(x.s, x.c);
}

void m2 (String s, char c) {
    s = s + '.';
    c = '.';
}
```

Wat wordt er bij dit gegeven door de onderstaande statements afgedrukt?

```
out.println(xObject.c);
out.println(xObject.s);
m1(xObject);
out.println(xObject.c);
out.println(xObject.s);
```

b) Schrijf voor $n \geq 1$ een methode : `int berekenReeks (int x, int n)` die de sommatie uitreken van de eerste n termen van een reeks. De i -de term ($i \geq 1$) van deze reeks is als volgt gedefinieert.

$$((-1)^i * i) / (x^i)$$

Met de notatie `a*b` wordt " a tot de macht b " bedoeld.

Met de notatie `a!` wordt " a faculteit" oftewel $1 \times 2 \times \dots \times a$ bedoeld. Het is niet toegestaan om methodes uit de class `Math` te gebruiken.

d)

Indien gewenst kan gebruik gemaakt worden van het feit dat de meest negatieve `int` waarde in Java gelijk is aan: `Integer.MIN_VALUE`

Gegeven is het programma in de onderstaande class:

```
class Opgave_1d {
    Output out;
    int a = -3,
    b = 2;

    Opgave_1d() {
        out = new Output();
    }

    void print (int a, int b) {
        out.println(a);
        out.println(b);
    }

    int method1(int b) {
        b = b / a;
        a = b - a;
        print(a, b);
        return a;
    }

    int method2 (int b, int c) {
        int a = b;
        b = a - c;
        c = method1(a);
        print(a, b);
        return b;
    }

    void start() {
        a = method1(a);
        b = method2(a, b);
        print(a, b);
    }

    public static void main(String argv[]) {
        new Opgave_1d().start();
    }
}
```

welke uitvoer geeft dit programma?

matrix m van gehele getallen met maximum van de kolom som van de kolom negatieve int waarde in Java gelijk is aan: Integer.MIN_VALUE is de sommatie van de getallen in die kolom.

int maxOnevenKolomSom (int[][] m)

59

- Voorrie de class `Auto` van een constructor met parameters om alle velden van de class `Auto` op zinnvolle waarden te initialiseren bij het aanmaken van een nieuw `Auto` object.

Denk er aan dat er voor iedere class altijd een default constructor gemaakt moet worden die alle variabelen een initiële waarde geeft.

- Voorzie de class `Snelweg` van een methode `voegToe()` om een `Auto` object toe te kunnen voegen aan het `Snelweg` object.

- c) Voeg aan de class Snelweg de onderstaande methode zoek() toe die, gegeven het merk van een auto, in een nieuw Snelweg object, alle auto's van dat merk retourneert die nog openstaande boetes hebben.

Snelweg zoek (string merk)

Pak dit gestructureerd aan. Voeg indien er behoefte is aan methodes die iets doen met een Auto object deze methodes toe aan de class Auto.

HINTS:

- a) Schrijf een recursieve methode `printReeks()` om de X -reeks voor gehele getallen > -1 af te drukken, waarbij de X -reeks van een getal gedeclineerd is als:

als n - 1

```
X = reeks(n DIV 2)  n X reeks(n DIV 2)  als n > 1
```

Voorbeelden: n x -reeks

1	1	1
2	2	1
3	3	1
4	2	4
5	3	6
6	1	3
7	4	1
8	2	4

- 59

Deze opvallende over- en onderdrukkingen zijn te verklaren door de verschillen in de manier waarop de operatoren de uitdrukkingen interpreteren. De operatoren zien de uitdrukkingen als een geheel, terwijl de leerlingen de uitdrukkingen zien als een verzameling van termen. Dit kan worden geïllustreerd aan de hand van de volgende voorbeelden:

De operanden kunnen zelf ook weer expressies in prefixnotatie staan. De enige operatoren die voor deze opgaven gebruikt worden zijn $+$ en $*$ voor respectievelijk sommatie en product.

Voorbeelden van int-expressies in prefixnotatie zijn

7		7	
(+2, 3)	betekent	(2 + 3)	
(+4, 5)	betekent	(4 * 5)	
(*(+2, 3), 4)	betekent	((2 + 3) * 4)	
(*(+2, 3), (*4, 5))	betekent	((2 + 3) * (4 * 5))	

of, al wat ingewikkelder

$$(+(*3(+(*2,4),(+3,5))),(+(*10,7),6)).$$

wat overeenkomt met

$$((3 * (2 * 4) + (3 + 5))) + ((10 + 7) + 6).$$

Als gegeven is dat de leeswijzer het eerste karakter van een (NUL-leege) regel invoer aanijnt en dat de invoer een int-expressie in prefixnotatie bevat, schrijf dan een recursieve methode expressie() die de uitkomst van de int-expressie op de invoer retourneert. De heading van de methode expressie() is:

de invoer retourneert. De heading van de melhode `expresie()` is:

```
int expresse (
```

HINTS:

Als we naar het voorbeeld $(*(1,2,3), (1,4,5))$ kijken dan zien we dat om de totale expressie uit te rekenen, de operanden $(+2,3)$ en $(1,4,5)$ naderkend moeten worden. Deze twee operanden zijn zelf ook weer (kleinere) inf-expressies in prefixnotatie. Hier herkennen we dus de recursiestap.

Bij het uitlekken van b.v. (+2,3) hoeven de operanden 2 en 3 slechts ingelezen te worden om hun waarde te bepalen, m.a.w. als een operand niet met een '1' begint is de waarde ervan te bepalen met in.readint(). Dit leidt tot de stroponditie.

Wardlering

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

Het eindcijfer E volgt uit het puntentotaal T als volgt : $E = T / 4 + 1$