

Oplossing oefeningen Hoofdstuk 4

Oefening 1:

```
{
    // declaratie
    int vta1, verm;    // vermenigvuldigtal en vermenigvuldiger
    int prod;          // sommatie van vermenigvuldigtal
    int atIOpt;        // aantal reeds uitgevoerde optellingen

    // inbrengen vermenigvuldigtal en vermenigvuldiger
    System.out.println("Geef vermenigvuldigtal en vermenigvuldiger: ");
    vta1 = sc.nextInt();
    verm = sc.nextInt();
    // initialiseren op nul optellingen
    prod = 0;
    // opeenvolgend product verhogen met vermenigvuldigtal
    for (atIOpt = 0; atIOpt < verm; atIOpt = atIOpt + 1)
    {
        prod = prod + vta1;
    } // end for
    // afbeelden product
    System.out.println(vta1 + " * " + verm + " = " + prod);
}
```

Uitbreiding oefening 1:

```
{
    // declaratie
    int vta1, verm;    // vermenigvuldigtal en vermenigvuldiger
    int prod;          // sommatie van vermenigvuldigtal
    int atIOpt;        // aantal reeds uitgevoerde optellingen

    // inbrengen vermenigvuldigtal en vermenigvuldiger
    System.out.println("Geef vermenigvuldigtal en vermenigvuldiger: ");
    vta1 = sc.nextInt();
    verm = sc.nextInt();
    // initialiseren op nul optellingen
```

```

    prod = 0;
    // opeenvolgend product verhogen met vermenigvuldigtal
    for (atlOpt = 0; atlOpt < Math.abs(verm); atlOpt = atlOpt + 1)
    {
        prod = prod + vtal;
    } // end for
    // indien vermenigvuldiger negatief, product tegengesteld maken
    if (verm < 0)
    {
        prod = -prod;
    }
    // afbeelden product
    System.out.println(vtal+" * "+verm+" = "+prod);
}

```

Oefening 2:

```

{
    Scanner sc = new Scanner(System.in);
    // declaratie
    int atlFibon;           // aantal af te beelden fibonacci getallen
    int term1, term2, term3; // 3 opeenvolgende termen in de rij
    int atlTermen;          // aantal reeds verwerkte termen

    // inbrengen strikt positief: aantal termen
    System.out.print("Geef het aantal termen in: ");
    atlFibon = sc.nextInt();

    // toewijzen eerste 2 termen van de rij
    term1 = 0;
    term2 = 1;
    // afbeelden eerste 2 termen
    System.out.print(term1 + " ");
    System.out.print(term2 + " ");
    // volgende termen berekenen en afbeelden
    for (atlTermen = 2; atlTermen < atlFibon; atlTermen = atlTermen + 1)
    {
        term3 = term1 + term2;
        System.out.print(term3 + " ");
        // 2 vorige termen opschuiven
        term1 = term2;
        term2 = term3;
    } // end for
}

```

Oefening 3:

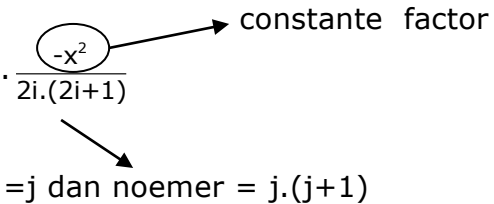
Ontwikkeling van het algoritme

$$1^{\circ} \text{ term} = x$$

$$2^{\circ} \text{ term} = -\frac{x^3}{3!} = x \cdot \left(\frac{-x^2}{2 \cdot 3}\right) = (1^{\circ} \text{ term}) \cdot \left(\frac{-x^2}{2 \cdot 3}\right)$$

$$3^{\circ} \text{ term} = \frac{x^5}{5!} = \left(-\frac{x^3}{3!}\right) \cdot \left(\frac{-x^2}{4 \cdot 5}\right) = (2^{\circ} \text{ term}) \cdot \frac{-x^2}{4 \cdot 5}$$

$$4^{\circ} \text{ term} = -\frac{x^7}{7!} = \left(\frac{x^5}{5!}\right) \cdot \left(\frac{-x^2}{6 \cdot 7}\right) = (3^{\circ} \text{ term}) \cdot \frac{-x^2}{6 \cdot 7}$$

$$(i+1)^{\circ} \text{ term} = i^{\circ} \text{ term} \cdot \frac{-x^2}{2i \cdot (2i+1)}$$


stel $2i = j$ dan noemer = $j \cdot (j+1)$

```
{
// declaratie
double x;      // argument van de sinus functie
double sin;    // sin(x)
double term;   // laatst berekende term
double factor; // vast gedeelte -x^2
int dubbelnr;  // dubbele van volgnummer van laatst berekende term

// inbrengen argument
System.out.print("Geef het argument voor sinus: ");
x = sc.nextDouble();

// initialiseren op de eerste term
term = x;
dubbelnr = 2;
sin = x;
factor = -x * x;

// opeenvolgende nieuwe termen berekenen en optellen bij sin
while (Math.abs(term) >= 1E-5)
{
    term = (term * factor)/(dubbelnr*(dubbelnr+1));
    sin = sin + term;
    dubbelnr = dubbelnr + 2;
} // end while

System.out.println("Sinus waarde van "+x+": "+sin);
}
```

Oefening 4:

Ontwikkeling van het algoritme

$$1^{\circ} \text{ term} = 1$$

$$2^{\circ} \text{ term} = -\frac{x^2}{2!} = 1 \cdot \left(\frac{-x^2}{1 \cdot 2}\right) = (1^{\circ} \text{ term}) \cdot \left(\frac{-x^2}{1 \cdot 2}\right)$$

$$3^{\circ} \text{ term} = \frac{x^4}{4!} = \left(-\frac{x^2}{2!}\right) \cdot \left(\frac{-x^2}{3 \cdot 4}\right) = (2^{\circ} \text{ term}) \cdot \frac{-x^2}{3 \cdot 4}$$

$$4^{\circ} \text{ term} = -\frac{x^6}{6!} = \left(\frac{x^4}{4!}\right) \cdot \left(\frac{-x^2}{5 \cdot 6}\right) = (3^{\circ} \text{ term}) \cdot \frac{-x^2}{5 \cdot 6}$$

$$(i+1)^{\circ} \text{ term} = i^{\circ} \text{ term} \cdot \frac{\overset{\text{constante factor}}{\textcircled{-x^2}}}{\underset{\text{stel } 2i = j \text{ dan noemer} = (j-1) \cdot j}{(2i-1) \cdot 2i}}$$

```
{
    // declaratie
    double x;      // argument van de cosinus functie
    double cos;    // cos(x)
    double term;   // laatst berekende term
    double factor; // vast gedeelte -x^2
    int dubbelnr;  // dubbele van volgnummer van laatst berekende term

    // inbrengen argument
    System.out.print("Geef het argument voor de cosinus: ");
    x = sc.nextDouble();

    // initialiseren op de eerste term
    term = 1;
    dubbelnr = 2;
    cos = 1;
    factor = - x * x;

    // opeenvolgende nieuwe termen berekenen en optellen bij cos
    while (Math.abs(term) >= 1E-5)
    {
        term = (term * factor)/((dubbelnr-1)*dubbelnr);
        cos = cos + term;
        dubbelnr = dubbelnr + 2;
    } // end while

    System.out.println("Cosinus waarde van "+ x +": "+cos);
}
```

Oefening 5:

Ontwikkeling van het algoritme

Vb. Stel ingebracht getal is 18

	getal	rest	oneven	atlafr	vierkw
gegeven	18				
initialisatie		18	1	0	
stap 1		17	3	1	
stap 2		14	5	2	
stap 3		9	7	3	
stap 4		2	9	4	
afsluiting					4

```
{
    // declaratie
    int getal;    // in te voeren positief getal
    int rest;     // overschot na aftrekkingen
    int oneven;   // volgend oneven getal om af te trekken
    int vierkw;   // vierkantswortel uit getal

    // inbrengen positief geheel getal
    System.out.println("Geef een positief geheel getal in: ");
    getal = sc.nextInt();

    // initialisatie
    rest = getal;
    oneven = 1;
    vierkw = 0;

    // opeenvolgende oneven getallen aftrekken van de rest
    while (rest >= oneven)
    {
        rest = rest - oneven;
        oneven = oneven + 2;
        vierkw = vierkw + 1;
    }

    System.out.println("De vierkantswortel van "+getal+" = "+vierkw);
}
```

Oefening 6:

Black box

Inputvariabele VorigeBM

<i>Equivalentieklassen</i>	<i>Vertegenwoordigers</i>
VorigeBM < 0	-1 (1) (ongeldig)
$0 \leq \text{VorigeBM} \leq 22$	0 (2) (geldig)
	22 (3) (geldig)
VorigeBM > 22	23 (4) (ongeldig)

Inputvariabele atlOng

<i>Equivalentieklassen</i>	<i>Vertegenwoordigers</i>
atlOng < 0	-1 (5) (ongeldig)
atlOng = 0	0 (6) (geldig)
atlOng = 1	1 (7) (geldig)
atlOng > 1	2 (8) (geldig)

Outputvariabele NieuweBM

<i>Equivalentieklassen</i>	<i>Vertegenwoordigers</i>
NieuweBM < 0	-1 (9) (ongeldig, aanpassen 0)
$0 \leq \text{NieuweBM} \leq 22$	0 (10) (geldig)
	22 (11) (geldig)
NieuweBM > 22	23 (12) (ongeldig, aanpassen 22)

Testbatterij

<i>Test</i>	<i>VorigeBM</i>	<i>atlOng</i>	<i>NieuweBM</i>	<i>Verantwoording</i>
1	0	1	4	geldige klassen (2), (7)
2	22	0	21	geldige klassen (3), (6)
3	0	2	9	geldige klassen (2), (8)
4	18	1	22	geldige klasse (11)
5	1	0	0	geldige klasse (10)

6	-1			ongeldige klasse (1)
7	23			ongeldige klasse (4)
8		-1		ongeldige klasse (5)
9	0	0	-1 → 0	ongeldige klasse (9): aanpassen nieuwe BM
10	19	1	23 → 22	ongeldige klasse (12): aanpassen Nieuwe BM

Oefening 7:

White box

Input			output		
X	Y	A	X	Y	A
10	10	0	9	9	18
10	10	-1	9	9	-1
10	9	0	9	8	0
11	11	0	11	11	22
11	11	-1	11	11	-1
11	10	0	11	10	0

Oefening 8:

- a) onderhoudbaarheid en aanpasbaarheid
- b) doorzichtigheid
- c) onderhoudbaarheid en aanpasbaarheid
- d) betrouwbaarheid
- e) doorzichtigheid
- f) efficiëntie
- g) gebruiksvriendelijkheid
- h) aanpasbaarheid
- i) doorzichtigheid
- h) alles tezamen