Oplossingen Reeks 5

Oefening 1

```
{
      final int DIM = 20;  // tabellen bevatten 20 elementen
      int[] a = new int[DIM];  // tabel met 2x index
      int[] b = new int[DIM];  // tabel met 3x index
      int[] c = new int[DIM];  //tabel met a[i]*b[DIM-(i+1)]
      int i;                   // index om tabellen te doorlopen


      // invullen tabellen a en b
      for (i = 0; i < DIM; i = i + 1)
      {
          a[i] = i * 2;
          b[i] = i * 3;
      }
      System.out.println("index\t a\t b\t c");
      for (i = 0; i < DIM; i = i + 1)
      {
         c[i] = a[i] * b[(DIM-1)-i];
         System.out.println(i+"\t "+ a[i]+"\t "+ b[i]+"\t "+ c[i]);
      }
}
```

Oefening 2

```
{
     final int DIMPTN = 40;            // tabel ptn bevat max. 40 el.
     int[] ptn = new int[DIMPTN];      // tabel met punten examen
     int atlStud;                      // aantal studenten
     final int DIMFREQ = 11;           // tabel freq bevat 11 elementen
     int[] freq = new int[DIMFREQ];    // frequentietabel
     double somPtn;                    // som van alle punten
     double gemiddelde;                // klasgemiddelde
     int i;                            // index om tabellen te doorlopen
```

```
    // inbrengen aantal studenten
    atlStud = receive();

    if (atlStud > DIMPTN)
    {
     System.out.println("Het programma is slechts geschikt voor "
        + DIMPTN + " studenten");
    }
    else
    {
     // initialiseren tabel freq
     for (i = 0; i < DIMFREQ; i = i + 1) freq[i] = 0;
     // inbrengen en verwerken punten
     for (i = 0; i < atlStud; i = i + 1)
     {
      ptn[i] = receive();
      freq[ptn[i]] = freq[ptn[i]] + 1;
     }
     // afbeelden frequentietabel en gemiddelde somPtn = 0;
     System.out.println("PUNTEN\t FREQ");
     somPtn = 0;
     for (i = 0; i < DIMFREQ; i = i + 1)
     {
      somPtn = somPtn + (freq[i] * i);
      System.out.println(i+"\t"+freq[i]);
     }
     if (atlStud == 0)
     {
      System.out.println("Er werden geen punten ingegeven !");
     }
     else
     {
      gemiddelde = somPtn / atlStud;
      System.out.println("Het gemiddelde is" + gemiddelde);
     } // end if
    } // end if
}
```

Oefening 2 Versie BlueJ

```
{
    int[] ptn = {5,6,4,9,9,7,8,8,6,3,4,2,0,0,5,8,3,6,4,6,6,7,6,5,10,9,6,9,10};
                                        // tabel met punten examen
    int atlStud;                        // aantal studenten
    final int DIMFREQ = 11;             // tabel freq bevat 11 elementen
    int[] freq = new int[DIMFREQ];      // frequentietabel
    double somPtn;                      // som van alle punten
    double gemiddelde;                  // klasgemiddelde
    int i;                              // index om tabellen te doorlopen
```

```
        atlStud = ptn.length;
        // initialiseren tabel freq
        for (i = 0; i < DIMFREQ; i = i + 1) freq[i] = 0;
        // inbrengen en verwerken punten
        for (i = 0; i < atlStud; i = i + 1)
        {
          freq[ptn[i]] = freq[ptn[i]] + 1;
        }
        // afbeelden frequentietabel en gemiddelde somPtn = 0;
        System.out.println("PUNTEN\t FREQ");
        somPtn = 0;
        for (i = 0; i < DIMFREQ; i = i + 1)
        {
          somPtn = somPtn + (freq[i] * i);
          System.out.println(i+"\t "+freq[i]);
        }
        if (atlStud == 0)
        {
          System.out.println("Er werden geen punten ingegeven !");
        }
        else
        {
          gemiddelde = somPtn / atlStud;
          System.out.println("Het gemiddelde is " + gemiddelde);
        } // end if
}
```

Oefening 3

```
{
    final int klLft = 15;
    final int grLft = 30;
    // minimumleeftijd 15j
    // maximumleeftijd 30j
    final int DIM = (grLft - klLft) + 1;
                    // tabellen bevatten 16 elementen
    double[] totLengte = new double[DIM];     // totale lengte per lftgroep
    int[] atl = new int[DIM];  // atl per lftgroep
    int i;                     // index om tabellen te doorlopen
    int lft;                   // leeftijd van een student
    int lengte;                // lengte van een student
    int index;                 // hulpvariabele
    double gemiddelde;    // gemiddelde voor een lftgroep
    double grGemLengte; // grootste gemiddelde lengte
    int lftGrGem;             // lftgroep grootste gemiddelde lengte
    char nog;                 // karakter ter controle om verder te gaan


  // initialiseren tabellen totLengte en atl
```

```
    for (i = 0; i < DIM; i = i + 1)
    {
      totLengte[i] = 0;
      atl[i] = 0;
    }
    // inlezen leeftijd en lengte van aantal studenten
    nog = 'j';
    while (nog != 'n')
    {
      lft = sc.nextInt();
      lengte = sc.nextInt();
      // bepalen van index voor tabellen totLengte en atl
      index = lft - klLft;
      totLengte[index] = totLengte[index] + lengte;
      atl[index] = atl[index] + 1;
      nog = sc.next().charAt(0);
    }
    //Bepalen leeftijdsgroep van grootste gemiddelde lengte
    grGemLengte = -1;
    lftGrGem = 0;
    for (i = 0; i < DIM; i = i + 1)
    {
      if (atl[i] != 0)
      {
        gemiddelde = totLengte[i] / atl[i];
        if (gemiddelde > grGemLengte)
        {
          grGemLengte = gemiddelde;
          lftGrGem = i + klLft;
        }
      } // end if
    }
    if (grGemLengte != -1)
    {
       System.out.println("De leeftijdsgroep met het grootste gemiddelde is "+
                          lftGrGem);
       System.out.println("De grootste gemiddelde lengte is " +
                          grGemLengte);
    }
    else
    {
       System.out.println("Er werden geen lengtes ingegeven !");
    }
}
```

Oefening 4

```
{
    final int DIM = 101;        // tabel WinNr bevat 100 elementen
                                //   + eindelement (-1)
    int[] winNr = new int[DIM]; // tabel met winnende nummers
    int lotNr;                  // te controleren lotnummer
    int i;                      // index om tabel te doorlopen



    // inlezen tabel met winnende nummers en lotnummer
    winNr = receive();
    lotNr = receive();
    // controleren of lotnummer winnend nummer is // (sequentieel zoeken)
    i=0;
    while ((winNr[i] != -1) && (lotNr != winNr[i]))
    {
      i = i + 1;
    }
    // afbeelden resultaat
    if (winNr[i] != -1)
      System.out.println("Dit is een winnend nummer");
    else
      System.out.println("Dit is geen winnend nummer");
}
```

Oefening 4 Versie BlueJ

```
{
    int[] winNr ={567,345,223,12,67,877,5,990,433,54,78,999,113,55};
                                // tabel met winnende nummers
    int lotNr;                  // te controleren lotnummer
    int i;                      // index om tabel te doorlopen



    System.out.print("Geef uw lotnummer in :");
    lotNr = sc.nextInt();
    // controleren of lotnummer winnend nummer is // (sequentieel zoeken)
    i=0;
    while ((i<winNr.length) && (lotNr != winNr[i]))
    {
      i = i + 1;
    }
```

```
    // afbeelden resultaat
    if (i != winNr.length)
      System.out.println("Dit is een winnend nummer");
    else
      System.out.println("Dit is geen winnend nummer");
}
```

Oefening 5

```
{
    final int DIM = 12;          // tabellen sprong, rugNr bevatten
                                 // 12 elementen
    int[] sprong = new int[DIM]; // tabel met beste sprongen
    int[] rugNr = new int[DIM]; // tabel met rugnummers van spelers
    int nummer;                  // in te lezen rugnummer
    int i;                       // index om tabellen te doorlopen
                                 // index van beginel van ongesorteerd deel
    int j;                       // index om ongesorteerd deel te doorlopen
    int g;                       // index van grootste el van ongesorteerd deel
    int hulp;                    // hulpvariabele om te verwisselen


    // inlezen tabel met sprongen en rugnummer
    for (i = 0; i < DIM; i = i + 1)
    {
      nummer = sc.nextInt();
      sprong[nummer-1] = sc.nextInt();
      rugNr[nummer-1] = nummer;
    }

    // beide tabellen sorteren volgens lengte sprong
    // (sortering door selectie)
    for (i = 0; i < DIM - 1; i = i + 1)
    {
      // grootste lengte sprong zoeken
      g = i;
      for (j = i + 1; j < DIM; j = j + 1)
      {
        if (sprong[j] > sprong[g])
        {
          g = j;
        }
      } // end for
      // zowel tab sprong, als tab rugNr sorteren volgens
      // lengte sprong
      hulp = sprong[i];
      sprong[i] = sprong[g];
      sprong[g] = hulp;
      hulp = rugNr[i];
      rugNr[i] = rugNr[g];
      rugNr[g] = hulp;
    } // end for
```

```
    // afbeelden gesorteerde lijst print "Rugnr Sprong";
    for (i = 0; i < DIM; i = i + 1)
    {
      System.out.println(rugNr[i]+" "+sprong[i]);
    }
}
```

Oefening 5 Versie bis

```
{
    final int DIM = 12;              // tabellen sprong, rugNr bevatten
                                     // 12 elementen
    int[] sprong = new int[DIM]; // tabel met beste sprongen
    int[] rugNr = new int[DIM];   // tabel met rugnummers van spelers
    int nummer;                      // in te lezen rugnummer
    int i;                           // index om tabellen te doorlopen
                                    // index van beginel van ongesorteerd deel
    int j;                          // index om ongesorteerd deel te doorlopen
    int g;                          // index van grootste el van ongesorteerd deel
    int hulp;                       // hulpvariabele om te verwisselen


    // inlezen tabel met sprongen en rugnummer
    for (i = 0; i < DIM; i = i + 1)
    {
      nummer = sc.nextInt();
      sprong[nummer-1] = sc.nextInt();
      rugNr[nummer-1] = nummer;
    }

    // Alleen de tabel van de rugnummers sorteren volgens de grootte van de
    // sprong
    // (sortering door selectie)
    for (i = 0; i < DIM - 1; i = i + 1)
    {
      // grootste lengte sprong zoeken
      g = i;
      for (j = i + 1; j < DIM; j = j + 1)
      {
        if (sprong[rugNr[j]-1] > sprong[rugNr[g]-1])
        {
          g = j;
        }
      } // end for
      // enkel tabel rugNr wijzigen (volgens sprong)
      hulp = rugNr[i];
      rugNr[i] = rugNr[g];
      rugNr[g] = hulp;
    } // end for
```

```
    // afbeelden gesorteerde lijst print "Rugnr Sprong";
    for (i = 0; i < DIM; i = i + 1)
    {
      System.out.println(rugNr[i]+" "+sprong[rugNr[i]-1]);
    }
}
```

Oefening 6

```
{
    final int DIM = 30;          // tabellen pos, neg bevatten max. 30 elementen
    int[] pos = new int[DIM]; // tabel met positieve getallen
    int[] neg = new int[DIM]; // tabel met negatieve getallen
    int getal;                   // in te lezen getal
    int atlGetallen;             // aantal ingelezen getallen
    int atlPos;                  // aantal ingelezen pos getallen
    int atlNeg;                  // aantal ingelezen neg getallen

    // inlezen getallen en onderbrengen in juiste tabel
    atlPos = 0;
    atlNeg = 0;
    for (atlGetallen = 0; atlGetallen < DIM; atlGetallen++)
    {
      getal = sc.nextInt();
      // bepalen in welke tabel
      if (getal < 0)
      {
        neg[atlNeg] = getal;
        atlNeg = atlNeg + 1;
      }
      else
      {
        pos[atlPos] = getal;
        atlPos = atlPos + 1;
      } // end if
    } // end for
}
```

Oefening 7

```
{
    final int DIM = 30;          // tabel neg bevat max. 30 elementen
    int[] neg = new int[DIM]; // tabel met negatieve getallen
    int atlNeg;                  // aantal ingelezen neg getallen
    int i;                       // index beginelement ongesorteerd deel
    int j;                       // index rechterelement van koppel
    int k;                       // index van rechterelement van laatst
                                 // verwisselde koppel
    int hulp;                    // hulpvariabele om te verwisselen
```

```
    neg = receive();
    atlNeg = receive();

    // sorteren van tabel neg via sortering door uitwisseling
    for (i = 0; i < atlNeg - 1; i = k)
    {
      k = atlNeg - 1;
      for (j = atlNeg - 1; j > i; j = j - 1)
      {
        if (neg[j] > neg[j-1])
        {
          hulp = neg[j-1];
          neg[j-1] = neg[j];
          neg[j] = hulp;
          k = j;
        } // end if
      }
    } // end for
}
```

Oefening 7 Versie BlueJ

```
{
    int[] neg = {-4,-7,-2,-56,-88,-3,-12,-45,-5,-77,-40,-3,-2,-6,-8};
                        // tabel met negatieve getallen

    int i;                  // index beginelement ongesorteerd deel
    int j;                  // index rechterelement van koppel
    int k;                  // index van rechterelement van laatst
                            // verwisselde koppel
    int hulp;               // hulpvariabele om te verwisselen
    final int DIM = neg.length;

    // sorteren van tabel neg via sortering door uitwisseling
    for (i = 0; i < DIM - 1; i = k)
    {
      k = DIM - 1;
      for (j = DIM - 1; j > i; j = j - 1)
      {
        if (neg[j] > neg[j-1])
        {
          hulp = neg[j-1];
          neg[j-1] = neg[j];
          neg[j] = hulp;
          k = j;
        } // end if
      }
    } // end for
    for(int getal : neg) System.out.print(getal+" "); // gebruik for-each !!
}
```

Oefening 8

```
{
    final int DIM = 24;    // tabel data bevat max. 24 elementen
    final int RIJDIM = 3;  // aantal rijen van matrices
    final int KOLDIM = 4;  // aantal kolommen van matrices
    int[] data = new int[DIM];  // tabel met gehele getallen
    int[][] a = new int[RIJDIM][KOLDIM]; // matrix a
    int[][] b = new int[RIJDIM][KOLDIM]; // matrix b
    int i;              // rijindex
    int j;              // kolomindex
    int index;          // index om data te doorlopen
    int rijsomA;        // rijsom matrix a
    int rijsomB;        // rijsom matrix b


    data = receive();

    // opvullen van beide matrices
    index = 0;
    for (i = 0; i < RIJDIM; i = i + 1)
    {
     for (j = 0; j < KOLDIM; j = j + 1)
     {
       a[i][j] = data[index];
       b[i][j] = data[index + 12];
       index = index + 1;
     }
    }
    // bepalen van grootste rijsom van a of b per rij
    for (i = 0; i < RIJDIM; i = i + 1)
    {
     rijsomA = 0;
     rijsomB = 0;
     for (j = 0; j < KOLDIM; j = j + 1)
     {
       rijsomA = rijsomA + a[i][j];
       rijsomB = rijsomB + b[i][j];
     }
     if (rijsomA > rijsomB)
       System.out.println("In rij " + (i+1) +
                   " heeft matrix A de grootste rijsom nl. " + rijsomA);
     else
       System.out.println("In rij " + (i+1) +
                   " heeft matrix B de grootste rijsom nl. " + rijsomB);
    }
}
```

Oefening 8 Versie BlueJ

```
{
    final int RIJDIM = 3;  // aantal rijen van matrices
    final int KOLDIM = 4;  // aantal kolommen van matrices
    int[] data = {1,2,3,4,13,14,15,16,9,10,11,12,
                  17,18,19,20,5,6,7,8,21,22,23,24};
                  // tabel met gehele getallen
    int[][] a = new int[RIJDIM][KOLDIM]; // matrix a
    int[][] b = new int[RIJDIM][KOLDIM]; // matrix b
    int i;               // rijindex
    int j;               // kolomindex
    int index;           // index om data te doorlopen
    int rijsomA;         // rijsom matrix a
    int rijsomB;         // rijsom matrix b

    // opvullen van beide matrices
    index = 0;
    for (i = 0; i < RIJDIM; i = i + 1)
    {
      for (j = 0; j < KOLDIM; j = j + 1)
      {
        a[i][j] = data[index];
        b[i][j] = data[index + 12];
        index = index + 1;
      }
    }
    // bepalen van grootste rijsom van a of b per rij
    for (i = 0; i < RIJDIM; i = i + 1)
    {
      rijsomA = 0;
      rijsomB = 0;
      for (j = 0; j < KOLDIM; j = j + 1)
      {
        rijsomA = rijsomA + a[i][j];
        rijsomB = rijsomB + b[i][j];
      }
      if (rijsomA > rijsomB)
        System.out.println("In rij " + (i+1) +
                    " heeft matrix A de grootste rijsom nl. " + rijsomA);
      else
        System.out.println("In rij " + (i+1) +
                    " heeft matrix B de grootste rijsom nl. " + rijsomB);
    }
}
```

Oefening 9 (de 5 modules na elkaar)

```
{
        // MODULE 1
    final int ATLVAK = 12;        // aantal vakken
    int[] max = new int[ATLVAK];  // tabel met te behalen punten van de
                                  // vakken
    double totBehaald;            // maximum te behalen punten
                                  // voor alle vakken samen
    max = receive();
    // bepalen maximum te behalen punten
    totBehaald = 0;
    for (i = 0; i < ATLVAK; i = i + 1)
    {
     totBehaald = totBehaald + max[i];
    }
    // end module 1

     // MODULE 2
    final int ATLVAK = 12;              // aantal vakken
    final int ATLSTUD = 35;             // aantal studenten
    int[][] res=new int[ATLVAK][ATLSTUD]; // resultaten van studenten
    String[] vak = new String[ATLVAK];        // namen van de vakken
    int[] max = new int[ATLVAK]; // maximum te behalen punten van de vakken
    int klasnummer;                // klasnummer van student
                                   // van wie het rapport wordt afgebeeld

    double behaald;                // totaal aantal punten behaald
                                   // door student

    double totBehaald;             // maximum te behalen punten
                                   // voor alle vakken samen
    int i;                         // index om tab vak te doorlopen

    res = receive();
    vak = receive();
    totBehaald = receive();
    // inlezen klasnummer
    klasnummer = receive();
    // bepalen percentage en afbeelden rapport
    behaald = 0;
    for (i = 0; i < ATLVAK; i = i + 1)
    {
     System.out.println(vak[i]+" "+res[i][klasnummer - 1]+" op "+max[i]);
     behaald = behaald + res[i][klasnummer - 1];
    }
    System.out.println("Procent: "+((behaald / totBehaald) * 100)+"%");

    // end module 2

    // MODULE 3
```

```
final int ATLVAK = 12;   // aantal vakken
final int ATLSTUD = 35;  // aantal studenten
int[][] res = new int[ATLVAK][ATLSTUD]; // resultaten van studenten
double[] proc = new double[ATLSTUD];    // percentages van studenten
int[] klasnr = new int[ATLSTUD];        // klasnummers van studenten


double behaald;          // totaal aantal punten behaald
                         // door student
double totBehaald;       // maximum te behalen punten voor
                         // alle vakken samen
int i;                   // rij index
int j;                   // kolomindex
                         // rij index voor proc, klasnr
int hulp;                // hulpvariabele om te verwisselen

res = receive();
totBehaald = receive();

// opbouwen tabellen proc en klasnr
for (j = 0; j < ATLSTUD; j = j + 1)
{
  behaald = 0;
  for (i = 0; i < ATLVAK; i = i + 1)
  {
    behaald = behaald + res[i][j];
  }
  proc[j] = (behaald / totBehaald) * 100;
  klasnr[j] = j + 1;
}

// sorteren van beide tabellen (door selectie)
int g;              // index van grootste element
                    // ongesorteerd deel
for (i = 0; i < DIM - 1; i = i + 1)
{
  // grootste procent zoeken
  g = i;
  for (j = i + 1; j < DIM; j = j + 1)
  {
    if (proc[j] > proc[g])
    {
      g = j;
    }
  } // end for
  hulp = klasnr[i];
  klasnr[i] = klasnr[g];
  klasnr[g] = hulp;
  hulp = proc [i];
  proc [i] = proc [g];
  proc [g] = hulp;
```

```
    } // end for

    // end module 3

       //MODULE 4

    // declaratie
    final int ATLSTUD = 35;                 // aantal studenten
    int[] proc = new int[ATLSTUD];          // percentages van studenten
    int[] klasnr = new int[ATLSTUD];        // klasnummers van studenten
    int i;                                  // index om tabellen proc
                                            // klasnr te doorlopen
    proc = receive();
    klasnr = receive();
    //Overzicht afbeelden van 20 beste studenten
    for (i = 0; i < 20; i = i + 1)
    {
      System.out.println(klasnr[i]+" "+proc[i]);
    }

    // einde module 4

      // MODULE 5
    final int ATLVAK = 12;                  // aantal vakken
    final int ATLSTUD = 35;                 // aantal studenten
    int[] res = new int[ATLVAK][ATLSTUD];   // resultaten van studenten
    int[] vak = new int[ATLVAK];            // namen van de vakken
    int[] max = new int[ATLVAK];            // maximum te behalen punten
                                            //van de vakken

    double somVak;          // totaal aantal punten
                            // behaald door student
    int i;                  // rij index
    int j;                  // kolomindex

    res = receive();
    vak = receive();
    max = receive();

    // overzichtslijst van gemiddelde per vak
    for (i = 0; i < ATLVAK; i = i + 1)
    {
      somVak = 0;
      for (j = 0; j < ATLSTUD; j = j + 1)
      {
        somVak = somVak + res[i][j];
      }
      System.out.println(vak[i]+" "+max[i]+" "+somVak/ATLSTUD);
    }
}
```

Oefening 10 Versie BlueJ

```
{
      int[][] a = {{1,2,3},
                   {4,5,6},
                   {7,8,9}
                  };
      int[][] b = {{0,2,4},
                   {6,8,10},
                   {12,14,16}
                  };
      int[][] c = new int[3][3];
      int i;
      int j;
      int k;
      int som;

      if(a[0].length == b.length) // atl kolommen a == atl rijen b
      {
         for (i = 0; i < a.length; i = i + 1)
         {
          for (j = 0; j < b[0].length; j = j + 1)
          {
            som = 0;
            for (k = 0; k < a[0].length; k = k + 1)
            {
              som = som + a[i][k] + b[k][j];
            }
            c[i][j] = som;
          }
         }
         // afdrukken c
         for(i = 0; i < c.length; i++)
         {
          for(j = 0; j < c[0].length; j++)
          {
             System.out.print(c[i][j]+" ");
          }
          System.out.println();
         }
      }
      else
      {
         System.out.println("Vermenigvuldiging is onmogelijk");
      }
}
```