Uitwerking Tentamen Programmeermethoden maandag 6 januari 2003

1.a.
```
int zoek (int A[ ], int n, int X) {
  int i = 0;
  while ( i < n )
    if ( A[i] == X )
      return i;
    else i++;
  return -1;
}  // zoek
```
b.
```
void draaien (int A[ ], int n) {
  int temp = A[0]; int i;
  for ( i = 0; i < n-1; i++ )
    A[i] = A[i+1];
  A[n-1] = temp;
} // draaien
```
c.
```
int eennakleinste (int A[ ], int n) {
  int i = 0; int r = -1;
  while ( r == -1 ) {
    r = zoek (A,i);
    i++;
  }  // while
  // nu is i-1 de kleinste
  r = -1;
  while ( r == -1 ) {
    r = zoek (A,i);
    i++;
  }  // while
  return i-1;
}  // eennakleinste
```

2.a. formeel: in functieheading, bijvoorbeeld int f (int x, bool y) { ...
actueel: bij aanroep, bijvoorbeeld r en y in z = f (r,y);
call by value: waarde wordt doorgegeven aan locale kopie; actuele parameter verandert niet
call by reference (met & erbij): variabele zelf wordt doorgegeven, en kan dus ook veranderen (eigenlijk wordt het adres doorgegeven)
globaal: kan overal gebruikt worden; bovenaan programma aangemaakt
locaal: plaatselijk in functie geldig

b. 2 4 9 16 4 10 1 5 30 30 1 5 10

c. 2 4 9 16 3 10 1 3 29 29 1 3 10
2 4 9 16 4 10 1 4 30 30 1 4 10

d. Fout. Op de plek van eerste parameter van plus moet een call by value parameter staan in verband met aanroep plus (plus (x,y),y).

e. 9 7 9 26 7 10 8 8 43 43 1 5 10

3.a.
```
bool randomplaats (char puzzel[ ][n], int & rij, int & kol) {
  int tel = 1;
  while ( tel <= 10000 ) {
    tel++;
    rij = random ( ) % m;  // m en n globaal ...
    kol = random ( ) % n;
    if ( puzzel[rij][kol] == '.' )
      return true;
  }  // while
  return false;
}  // randomplaats
```
b.
```
bool plaatswoord (char puzzel[ ][n], int rij, int kol,
                  char woord [ ], int woordlen) {
  int j; bool okee = true;  // kan ook met een return ...
  for ( j = 0; j < woordlen; j++ )
    if ( kol+j < n ) {
      if ( puzzel[rij][kol+j] != '.' && puzzel[rij][kol+j] != woord[j] )
        okee = false;
    }  // if
    else
      okee = false;  // woord te lang
  if ( okee )
    for ( j = 0; j < woordlen; j++ )
      puzzel[rij][kol+j] = woord[j];
  return okee;
}  // plaatswoord
```
c.
```
bool vindwoord (char puzzel[ ][n], char woord[ ], int woordlen) {
```

```
      int i, j, ri; bool okee;
      for ( i = 0; i < m; i++ )
        for ( j = 0; j < n; j++ )
          for ( ri = 0; ri < 2; ri++ ) {
            i1 = i;
            j1 = j;
            okee = true;
            for ( k = 0; k < woordlen; k++ ) {
              if ( i1 >= m || j1 >= n || puzzel[i1][j1] != woord[k] )
                okee = false;
              if ( ri == 0 )
                i1++;
              else
                j1++;
            }  // for
            if ( okee )
              return true;
          }  // for
      return false;
    }  // vindwoord

4.a. void route::voeg_terrein_toe (int terrein) {
      terrein_element* nieuw = new terrein_element;
      nieuw->type = terrein;
      nieuw->volgende = begin;
      begin = nieuw;
    }  // voeg_terrein_toe
  b. void push (terrein_element* rotsen_array[ ], int & index,
                terrein_element* pijl) {
      rotsen_array[index] = pijl;
      index++;
    }  // push
  c. terrein_element* pop (terrein_element* rotsen_array[ ], int & index) {
      if ( index > 0 ) {
        index--;
        return rotsen_array[index];
      }  // if
      else
        return NULL;
    }  // pop
  d. int sprongen (terrein_element* rotsen_array[ ], int & index) {
      int teller = 0;
      terrein_element* pijl = pop (rotsen_array,index);
      while ( pijl != NULL ) {
        if ( pijl->volgende != NULL && pijl->volgende->type == 3 )
          teller++;
        pijl = pop (rotsen_array,index);
      }  // while
      return teller;
    }  // sprongen
```