1 Basisconcepten C++/strings/templates/default param/bestanden

Oefening 101

```
#include <iostream>
using std::cout;
using std::endl;

int main(){
    cout << "Hello world!\n";   // of cout << "Hello world!" << endl;
    for(int i=10; i>0; i--){
        cout << i << " ";
    }
    cout << endl << "START";
    return 0;
}
```

Oefening 102

```
#include <iostream>
#include <iomanip>
using std::cout;
using std::endl;
using std::setw;
using std::oct;
using std::hex;
using std::dec;
// of kortweg
// using namespace std;

int main(){
    for(int i=0; i<=64; i++){
        cout << setw(6) << oct << i << setw(6) << dec << i << setw(6)
             << hex << i << endl;
    }

    return 0;
}
```

Oefening 103

```
/* Het eerste stukje werkt niet. De reden: de code
  "" + c
start met een constante c-string. Daar tel je met de +-operator iets bij.
Maar c-strings tel je niet op met '+', wel met strcat.

Je zou kunnen argumenteren dat dit ook de notatie is voor een constante
standard-string, maar hoe moet de compiler het verschil weten?
De c-strings waren er eerst, dus de compiler interpreteert "" als een c-string.

Het tweede stukje werkt wel. De reden: de code
  string w = "";
zorgt ervoor dat de constante "" c-string gecast wordt naar een (standard-)string.
Strings kan je wel aan elkaar plakken met de +-operator. */
```

Oefening 104

```
#include <string>
#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;

string genereer_string(int n){
    string s = "";
    for(int i=0; i<n; i++){
        s += ('a'+rand()%26);
    }
    return s;
}

void vul_array_met_strings(string * tab, int n, int len){
    for(int i=0; i<n; i++){
        tab[i] = genereer_string(len);
    }
}

void schrijf(const string * tab, int n){
    for(int i=0; i<n; i++){
        cout<<tab[i]<<"  -  ";
    }
}

void bepaal_min_en_max(const string * tab, int n, string & min, string & max){
    min = tab[0];
    max = tab[0];
    for(int i=1; i<n; i++){
        if (min > tab[i]){
            min = tab[i];
        }
        else if (max < tab[i]){
            max = tab[i];
        }
    }
}

int main(){

    srand(time(NULL));

    cout<<genereer_string(10)<<endl;
    cout<<genereer_string(10)<<endl;

    string tab[10];
    vul_array_met_strings(tab,10,3);
    schrijf(tab,10);

    string min, max;
    bepaal_min_en_max(tab,10,min,max);
    cout<<endl<<"min is "<<min;
    cout<<endl<<"max is "<<max<<endl;

    return 0;
}
```

Oefening 105

```
#include <iostream>
#include <string>
using namespace std;

/*************** STRUCT EN INITIALISATIE / PRINT VAN STRUCT *************/

struct Persoon{
     string naam;
     int leeftijd;
     double lengte;
};

void initialiseer(Persoon & p, const string & naam, int leeftijd, double lengte){
     p.naam = naam;
     p.leeftijd = leeftijd;
     p.lengte = lengte;
}

void print(const Persoon& p){ //geen kopie nemen
     cout << p.naam << " (" << p.leeftijd << " jaar, "
          << (int)p.lengte <<"m" << ((int)(p.lengte-1)*100) <<")";
}

/****************** FUNCTIE 'grootte' VOOR VERSCHILLENDE TYPES********/

double grootte(double x){
     return x;
}

int grootte(const string & woord){
     return woord.size();
}

int grootte(const Persoon & p){
     return p.leeftijd;
}

/* // alternatief: grootte van persoon wordt bepaald door zijn/haar lengte
double grootte(const Persoon & p){
     return p.lengte;
}*/

/* // alternatief: grootte van persoon wordt bepaald door lengte van zijn/haar naam
int grootte(const Persoon & p){
     return grootte(p.naam);
}*/

/******************* FUNCTIE 'grootste' VOOR VERSCHILLENDE TYPES VAN ARRAYS ***/

template <class T>
T grootste(const T * array, int lengte){
     T gr = array[0];
     for(int i=1; i<lengte; i++){
          if(grootte(gr) < grootte(array[i])){
               gr = array[i];
          }
     }
     return gr;
}
```

```
/************* Aanvullingen in main*********************************************/
     Persoon personen[3];
     initialiseer(personen[0],"samuel",12,1.52);
     initialiseer(personen[1],"jente",22,1.81);
     initialiseer(personen[2],"idris",42,1.73);
     Persoon gr = grootste(personen,3);
     print(gr);
```

Oefening 106

```
#include <iostream>
using namespace std;

void schrijf(const int * array, int aantal, bool achterstevoren = false, char
   tussenteken = ' ');

int main(){}
//default-waarden niet herhalen
void schrijf(const int * array, int aantal, bool achterstevoren, char tussenteken){
    if(achterstevoren){
        cout << array[aantal-1];
        for(int i=aantal-2; i>=0; i--){
            cout << tussenteken << array[i];
        }
    }
    else{
        cout << array[0];
        for(int i=1; i<aantal; i++){
            cout << tussenteken << array[i];
        }
    }
    cout << endl;
}
```

Oefening 107

```
#include <iostream>
#include <fstream>
using namespace std;

int main(){
    int frequentie[26] = {0};
    ifstream invoer("lord.txt");
    if(!invoer.is_open()){
        cout<<"bestand niet gevonden"<<endl;  // wordt later een exceptie
    }
    else{
        char letter;
        invoer >> letter;
        while( ! invoer.fail() ){
            if('a' <= letter && letter <= 'z'){
                frequentie[(letter-'a')] ++;
            }
            invoer >> letter;
        }
        for(int i=0; i<26; i++){
            cout<<(char)('a'+i)<<"  "<<frequentie[i]<<endl;
        }
```

```
    }
    invoer.close(); //optioneel, gebeurt automatisch op het einde
    return 0;
}
```

Oefening 108

```
#include <iostream>
#include <fstream>
#include <string>
using std::ifstream;
using std::ofstream;
using std::cout;
using std::endl;
using std::string;

int main(){
    ifstream in_1("stationnetje.txt");
    ifstream in_2("paddestoel.txt");
    ofstream uit("mix.txt");

    if(!in_1.is_open() || !in_2.is_open()){
        cout<<"minstens een bestand werd niet gevonden - jammer.... "<<endl;
    }
    else{
        int teller = 0;
        string zin_1,zin_2;
        getline(in_1,zin_1);
        getline(in_2,zin_2);
        while(!in_1.fail() && !in_2.fail()){
            if(teller%2==0){
                uit << zin_1 << endl;
            }
            else{
                uit << zin_2 << endl;
            }
            teller++;
            getline(in_1,zin_1);
            getline(in_2,zin_2);
        }
    }
    return 0;
}

// De oplossing van de uitbreiding laten we aan eigen inventiviteit over.
// In ieder geval: bewaar de invoerstreams in een array (of later in een vector).
```