Functies

checklist

- doelen:
  - duidelijkheid, aanpasbaarheid, onderhoudbaarheid,herbruikbaarheid .
- definitie en declaratie (=prototype)
- parameters en return
- formele en actuele parameters
- call by value (=**kopie**) return by value (=**kopie**)
- globale en lokale variabelen.
  - gebruik **geen** **globale** variabelen
    - slecht voor hierboven genoemde doelen
- livetime en scope.
- array als parameter
  - call by reference (=**geen** **kopie**)
- recursieve functie
  - roept zichzelf aan.
  - pas op voor stack overflow.

THRijswijk

Functie

Zonder parameter

```
#include <stdio.h>

void sla2RegelsOver(void) {
   printf("\n");
   printf("\n");
}

int main() {
   void sla2RegelsOver(void);
   sla2RegelsOver();
   printf("Hallo\n");
   sla2RegelsOver();
   printf("daar!\n");
   getchar();
   return 0;
}
```

THRijswijk

Functie

Met parameter

```
#include <stdio.h>

int main() {
   void slaRegelsOver(int n);
   slaRegelsOver(2);
   printf("Hallo\n");
   slaRegelsOver(4);
   printf("daar!\n");
   getchar();
   return 0;
}

void slaRegelsOver(int n) {
   int i;
   for (i=0; i<n; i=i+1) {
      printf("\n");
   }
}
```

THRijswijk

Functie

Met parameters en return

```
#include <stdio.h>

int main() {
   int leesWaardeTussen(int min, int max);
   int begin, eind, stap, i;
   printf("Geef startwaarde ");
   begin=leesWaardeTussen(-1, 99);
   printf("Geef eindwaarde ");
   eind=leesWaardeTussen(begin, 101);
   printf("Geef stap ");
   stap=leesWaardeTussen(0, eind-begin+1);
   for (i=begin; i<=eind; i=i+stap) {
      printf("%3d %6d\n", i, i*i);
   }
   getchar(); getchar();
   return 0;
}

int leesWaardeTussen(int min, int max) {
   int waarde;
   do {
      printf("tussen %d en %d: ", min, max);
      scanf("%d", &waarde);
   }
   while (waarde<=min || waarde>=max);
   return waarde;
}
```

THRijswijk

Functie

Met globale variabele


```
#include <stdio.h>

int n;

int main() {
   void slaRegelsOver(void);
   n=2;
   slaRegelsOver();
   printf("Hallo\n");
   n=4;
   slaRegelsOver();
   printf("daar!\n");
   getchar();
   return 0;
}

void slaRegelsOver(void) {
   int i;
   for (i=0; i<n; i=i+1) {
      printf("\n");
   }
}
```

THRijswijk

Variabelen

Scope en Livetime

- Scope is een statische eigenschap van een variabele = eigenschap tijdens het vertalen van het programma
  - waar kun je de variabele in de programmacode gebruiken?
    - **local** => block waarin de variabele gedefinieerd is.
    - **global** => hele programma.
- Livetime is een dynamische eigenschap van een variabele = eigenschap tijdens het uitvoeren van het programma
  - wanneer bestaat de variabele tijdens het uitvoeren van het programma?
    - **local** => vanaf binnenkomst van de functie tot aan de return uit de functie.
    - **global** => vanaf start programma tot aan einde programma.

THRijswijk

Local var

Scope en Livetime

```
#include <stdio.h>

void f(void);
void g(void);

void f(void) {
   int i=13;
   g();
   printf("i = %d\n", i);
}

void g(void) {
   printf("i is nu wel in leven,");
   printf("maar niet in scope!\n");
/* printf("i = %d\n", i); */
/* bovenstaande regel geeft foutmelding:
   'i' undeclared */
}

int main() {
   f();
   getchar();
   return 0;
}
```

THRijswijk

Global var

Scope en Livetime

```
#include <stdio.h>

int i;

void f(void);

void f(void) {
   int i=13;
   printf("globale i is nu wel in leven,");
   printf("maar niet in scope!\n");
   printf("i = %d\n", i);
/* bovenstaande regel gebruikt lokale i */
}

int main() {
   i=26;
   f();
   printf("i = %d\n", i);
   getchar();
   return 0;
}
```

THRijswijk

Recursief

Voorbeeld

```
#include <stdio.h>

int main() {
   int fac(int n);
   printf("3! = %d\n", fac(3));
   getchar();
   return 0;
}

int fac(int n) {
   int res;
   printf("begin n = %d\n", n);
   if (n==0)
      res=1;
   else
      res=n*fac(n-1);
   printf("eind n = %d\n", n);
   printf("res = %d\n", res);
   return res;
}
```

Uitvoer:

```
begin n = 3
begin n = 2
begin n = 1
begin n = 0
eind n = 0
res = 1
eind n = 1
res = 1
eind n = 2
res = 2
eind n = 3
res = 6
3! = 6
```

THRijswijk

Recursief

Beter van niet (in dit geval)

```
#include <stdio.h>

int main() {
   int fac(int n);
   printf("3! = %d\n", fac(3));
   getchar();
   return 0;
}

int fac(int n) {
   int i, res;
   res=1;
   for (i=1; i<=n; i=i+1)
      res=res*i;
   return res;
}
```