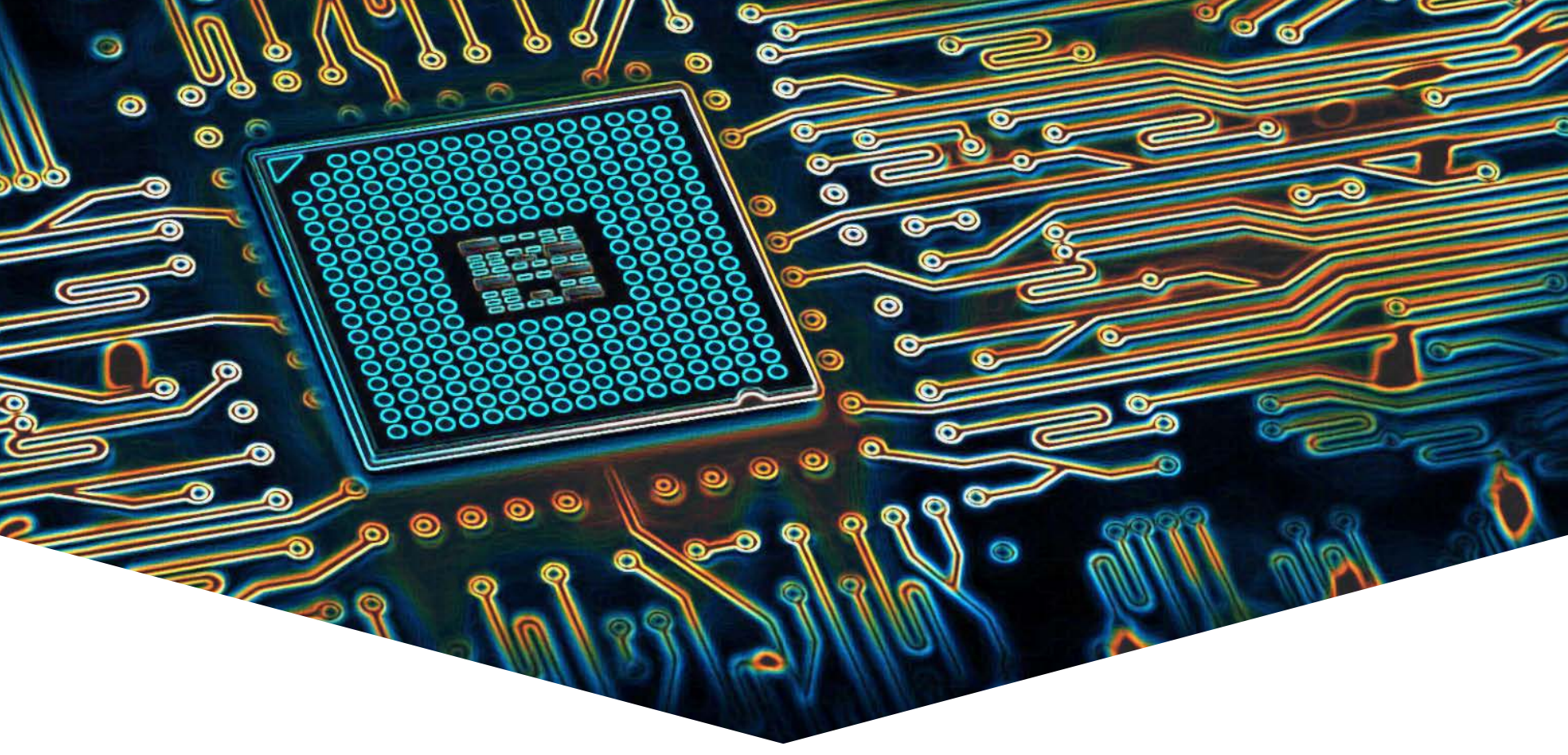


EMS20 Week 1 Lab 2

Leerdoelen Week 1 Lab 2

Leerdoelen week 1 lab 2. Je leert hoe je:

- het adres van de ene variabele kan opslaan in een andere variabele. Deze laatste variabele wordt dan een pointer genoemd;
- pointers kunt gebruiken om een C-functie meerdere waarden terug te laten geven;
- pointers kunt gebruiken om over een C-array of C-string te itereren;
- de executietijd van een stukje C-code kunt bepalen;
- de executietijd van een stukje C-code kunt verkorten door de optimalisatieinstellingen van de compiler te wijzigen.

Verwisselen

Schrijf een functie waarmee twee `int` variabelen verwisseld kunnen worden.

```
void wissel(int a, int b)
{
    int hulpje = a;
    a = b;
    b = hulpje;
}

int main(void)
{
    int x = 7, y = 8;
    printf("x = %d en y = %d\n", x, y);
    wissel(x, y);
    printf("x = %d en y = %d\n", x, y);
}
```

Eerste poging

```
x = 7 en y = 8
x = 7 en y = 8
```

Werkt **niet** goed!

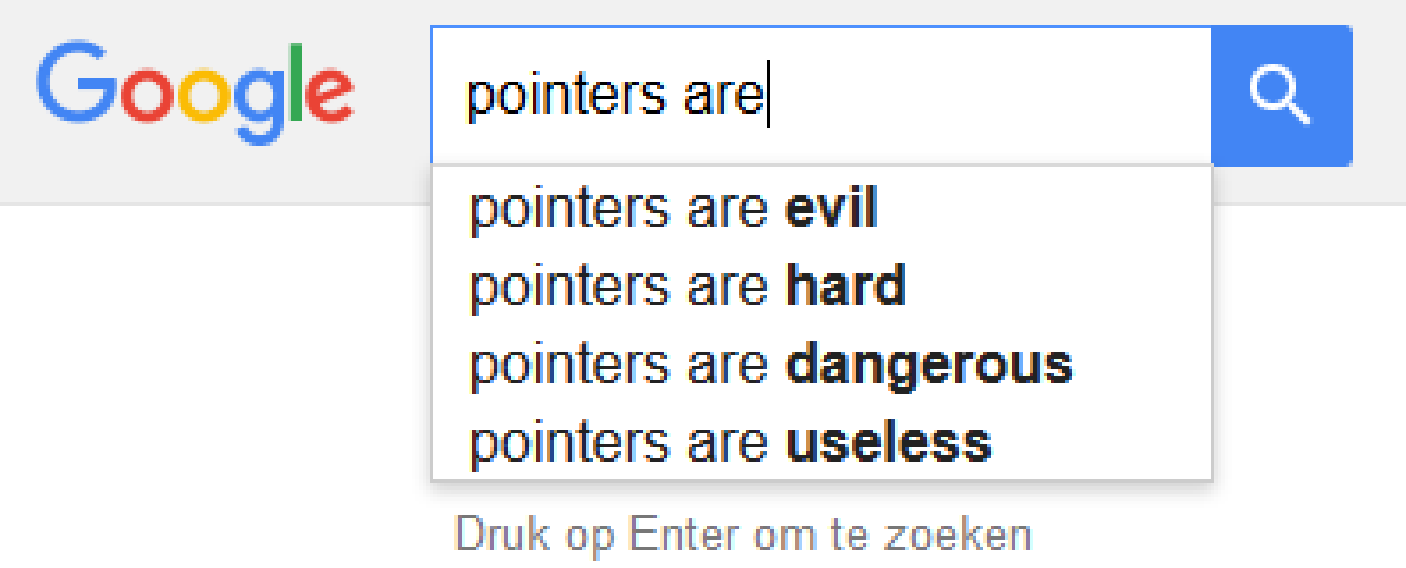
Bij **aanroep** van de functie wordt de **waarde** van het argument **gekopieerd** naar de parameter.

call by value

Pointers

C kent naast de “gewone” variabelen ook **pointer** variabelen.

Pointer are …



Allemaal **niet** waar! Laat je niet bang maken, maar let deze les wel goed op.

Geheugen

- Het geheugen van een computer bestaat uit **bytes** (groepje van 8 bits). Elk byte heeft zijn eigen **adres** (een nummer).

- Een **variabele** is opgeslagen in het geheugen vanaf een bepaald **adres**. Het **type** van de variabele bepaalt het aantal bytes wat nodig is om de waarde van de variabele op te slaan.

Adres en grootte opvragen

- Je kunt het **adres** van een variabele opvragen met behulp van de unary operator **&**.

```
int getal;
scanf("%d", &getal);
```

Adres van `getal`, zodat `scanf` de ingelezen waarde op dat adres kan opslaan.

- Je kunt het **aantal bytes** dat een variabele in beslag neemt opvragen met de operator **`sizeof`**.

```
int getal;
printf("%zu", sizeof getal);
```

Adres en grootte opvragen


```c
#include <stdio.h>
int global = 4;
int main(void)
{
    double local = 7.2;
    printf("global is opgeslagen op adres %p\n", &global);
    printf("en is %zu bytes groot\n", sizeof global);
    printf("local is opgeslagen op adres %p\n", &local);
    printf("en is %zu bytes groot\n", sizeof local);
    return 0;
}
```

Gebruik %p voor het printen van een adres in hexadecimale notatie.

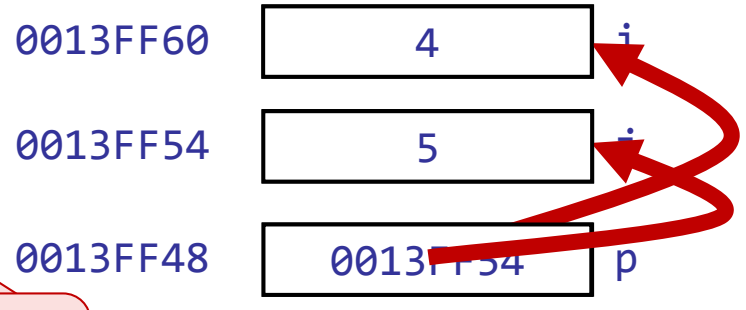
Gebruik %zu voor het printen van een size.

```
global is opgeslagen op adres 00417000
en is 4 bytes groot
local is opgeslagen op adres 0013FF5C
en is 8 bytes groot
```

Pointer

Je kunt het adres van een variabele opslaan in een pointer variabele.

```
int i = 3, j = 17;
int *p = &i;
*p = *p + 1;
p = &j;
*p = i + 1;
```



*p betekent:
waar p naar wijst

p is een int pointer die wordt geïnitialiseerd met het adres van i

Verwisselen

Schrijf een functie waarmee twee `int` variabelen verwisseld kunnen worden.

```
void wissel(int *p, int *q)
{
    int hulpje = *p;
    *p = *q;
    *q = hulpje;
}

int main(void)
{
    int x = 7, y = 8;
    printf("x = %d en y = %d\n", x, y);
    wissel(&x, &y);
    printf("x = %d en y = %d\n", x, y);
}
```

Tweede poging

```
x = 7 en y = 8
x = 8 en y = 7
```

Werkt **wel** goed!

call by reference

Opdracht

Bestudeer de werking van de functie `wissel` met behulp van de hand-out.

Volgende lab…

Structs en enums in C.

Aan de slag!

Aan de slag met [Opdrachten_Week_1_Lab_2.pdf]