Notities les 4

Jelle De Bock

October 13, 2015

1 Struct en pointers

```
cirkel * geefcirkel()
{
    cirkel c;
    //... al de creatie dingen
    return &c;
}
cirkel * d = geefcirkel();
```

1.1 Oefening met de cirkel

```
cirkel * zoek_punt(cirkel *t, int n, punt p)
{
    int i = 0;
    while(i<n)
    {
        if(t[i].mp.x == p.x && t[i].mp.y = p.y)
            return &t[i];
        i++;
    }
    return 0;
}

cirkel t[N];
punt p;
cirkel * cp;

zoek_punt(t, sizeof(t)/sizeof(cirkel),p);
```

2 Dynamisch geheugenbeheer

```
cirkel * geef_cirkel(){
    cirkel c;
```

```
    return &c;
}
```

Probleem hier is, van zodra geef_cirkel eindigt de cirkel c verdwijnt en je eingenlijk een loze pointer doorgeeft.

Oplossingen

- Kopie retourneren ... (slecht voor geheugenbeheer)
- Pointer meegeven, maar dan moet het ook in het hoofdprogramma wel al gereserveerd zijn.
- **Nieuw!!** malloc, geheugen reserveren voor variabele (in de functie)

Malloc Je zorgt ervoor dat iets wat gedeclareerd wordt in een functie stevig blijft staan, ook al eindigt deze functie. Malloc MOET altijd met een pointer aangezien het een adres teruggeeft naar waar het geheugen gereserveerd is. Ons voorbeeldje wordt dan...

```
cirkel * geef_cirkel(){
    cirkel *c = malloc(sizeof(cirkel));
    //je hebt nu een adres naar een
    //geheugenruimte waar zeker een cirkel in past
    return c;
```

free *free(pointer);* zorgt ervoor dat de ge-mallocte geheugenplaats terug vrijgemaakt wordt.

2.1 Voorbeeldje

```
int n,i;
scanf("%d",&n);
//je mag je array niet creeren als je at compile time
//niet weet hoe groot hij moet zijn
//t[n] mag NIET
int *t = malloc(n*sizeof(int));
//normaal geen pointers in hoofdprogramma
//hier kan het niet anders
for(i=0;i<n;i++)
{
      t[i]=i;
}
free(t);
```

2.2 Voorbeeldje

```
herhaal(const char * s, int n)
{
    int i=0;
    char * t = malloc((n*strlen(s)+1)*sizeof(char));
    t[0]=0; //of calloc gebruiken
    for(i=0;i<n;i++)
    {
        strcat(t,s);
    }
    return t;
}

int main()
{
    char s[80]="dag";
    char *u = herhaal(s,10);
    printf("%s",u);
    free(u);
}
```

3 Gelinkte lijsten

Houdt referentie naar het volgende veld bij. In C is dit een pointer naar een nieuw veld.