

ModProg – I

1.1-2: Programmeren

computer processor en geheugen
opdracht processor verandert geheugen
methode opdr.→methode
klasse opdr.→meth.→klasse
namespace opdr.→meth.→klas.→namespace
variabele geheugenplaats met een naam
object variabelen→object
objecten hebben als type een **class**

2. C# programma's

opbouw broncode **class**{methode{opdracht}}}
class **public/private**
class-members (**public/private**)(**static**)(**type/void**)
method body declaratie/opdracht
declaratie reserveert geheugen
declaratie **int x; string naam;**
opdracht aanroep/toekenning
opdracht-aanroep roept andere methode aan
opdracht-aanroep **Console.WriteLine(x);**
opdracht-toekenning verandert het geheugen
opdracht-toekenning **x = 1; naam.Text = "a";**
property eigenschap van een object
subklasse subversie van bestaande klasse
subklasse **class scherm : Form**
constructormethode maakt **new** object
constructormethode methode met naam van class
this object dat door methode bewerkt wordt
public bruikbaar in andere klassen
property van object **naam.Length**
niet-static methoden werkt op object
niet-static methoden **naam.ToUpper();**
static class ipv object
static properties **Color.Yellow**
static methoden **Console.WriteLine(x)**
Main altijd **static**
constructor nooit **static**
eventhandler **this.Paint+=teken;**
eventhandler toekenning (geen aanroep)

3. Teken

library **using System.Drawing;**
Paint-event (object o, **PaintEventArgs** pea)
pea.Graphics property van pea
pea.Graphics.DrawLine() tekent lijnen

DrawLine,Rectangle,Ellipse gebruiken **Pen**
FillRectangle,Ellipse gebruiken **Brush**
DrawString gebruikt **Brush**

3.3-5 Berekeningen

expressie alle code met een waarde
expressie ... **int, string, object-waarde, constructor new**
new Form() heeft **Form**-object als waarde
expressie kun je *uitrekenen* en heeft *waarde*
opdracht kun je *uitvoeren* en heeft *effect*
expressies kunnen deel uitmaken van opdracht
modulo % $8\%3=2$
const onaanpasbare declaratie
var declaratie met automatische typebepaling
void-methode geldt als opdracht
methode mét returntype geldt als expressie

4 Variabelen

sbyte [-128,127] (1 byte, 2^8)
short 2 bytes (2^{16} , ook -)
int 4 bytes (2^{64} , ook -)
long 8 bytes (2^{128} , ook -)
byte, ushort, uint, ulong zonder -
float 4 bytes met punt
double 8 bytes met punt
decimal 16 bytes met punt
opdrachten veranderen variabelen
methoden bewerken objecten
twee object-variabelen **struct** en **class**
bij struct .. toekenningen reserveren geheugenruimte
bij class toekenningen verwijzen naar objecten
class object verandert ... impact op alle verwijzingen

4.4 typeringen

int i; double d;
d = i; converteert **int** naar **double**
i = d; **ERROR**
i = (int) d; cast **double** naar **int**

6.2-5: Herhaling

bool waarde waarde is **true** of **false**
while(waarde) herhaling bij **true**
while(waarde) stopt bij **false**
vergelijkings-operatoren **<, <=, >, >=, ==, !=**
<, ==, != kleiner dan, is gelijk aan, ongelijk aan
Logische operatoren **&&(en), ||(of), !(niet)**
i++ **i** wordt opgehoogd
for(i=0;i<10;i++) herhaalt 10x

niet uitgevoerde herhaling **while(1==0)**
oneindige herhaling **while (1==1)**

Created by:
Raoul Grouls, Casper van Laar
2017