DE HAAGSE HOGESCHOOL
Technology, Innovation & Society Delft

VOORBLAD SCHRIFTELIJKE TOETSEN

OPLEIDING	: MECHATRONICA
TOETSCODE	: PLC-T1
GROEP	: MEH1
TOETSDATUM	: DONDERDAG 11 APRIL
TIJD	: ~~09.00 – 10.30 uur~~ ~~11.00 – 12.30 uur~~ ~~13.00 – 14.30 uur~~ 15.00 – 16.30 uur ~~18.00 – 19.30 uur~~ ~~20.15 – 21.45 uur~~
AANTAL PAGINA'S (incl. voorblad)	: 4
DEZE TOETS BESTAAT UIT	: 3 x 6 open vragen (aantal) ..... meerkeuzevragen (aantal)
GEBRUIK HULPMIDDELEN	: JA/~~NEE~~
TOETSOPGAVE INLEVEREN	: JA/~~NEE~~
TOEGESTANE HULPMIDDELEN	: **Alle aantekeningen / boeken / dictaat**
OVERIGE OPMERKINGEN	:
OPSTELLER VAN DEZE TOETS	: **G.M. Tuk**
TWEEDE LEZER VAN DEZE TOETS	: **J.J. Visser**

BELANGRIJKSTE PUNTEN UIT ARTIKEL 12 VAN DE ONDERWIJS- EN EXAMENREGELING:

- je dient je via Osiris ingeschreven te hebben voor deze toets
- schrijf je naam, je studentnummer, de toetscode en de naam van de docent meteen op het tentamenpapier
- leg je identiteitsbewijs op de hoek van de tafel
- zet alle elektronische communicatiemiddelen (mobiele telefoon, PDA, etc.) uit en stop deze in je tas; deze mogen niet als calculator of klok worden gebruikt
- je mag het lokaal het eerste halfuur niet verlaten
- volg de instructies op het toetsvoorblad
- steek je hand op als je een vraag hebt

Opgave 1

Gegeven zijn de klasse `Bestelling` en een `main` -methode, waarvan de code er als volgt uitziet.

```
class Bestelling{
public:
	Bestelling(int a, double p){
		aantal = a;
		prijs = p;
	};
	void reduceerPrijs(){
		prijs = prijs * 0.8;
	}
	double berekenTotaal(){
		return aantal * prijs;
	}
private:
	int aantal;
	double prijs;
};


int main(int argc, char* argv[])
{
	Bestelling bestelling(12, 2.95);
	bestelling.reduceerPrijs();
	double t = bestelling.berekenTotaal();

	return 0;
}
```

a) Leg uit waarom het wel of niet nodig is om voor de klasse `Bestelling` een *copy constructor* te maken.

b) Geef de code van de constructor zoals die eruit zou zien als de attributen `aantal` en `prijs` hun waarden zouden krijgen in een *initialization list*.

c) De methode `double berekenTotaal()` zou gedeclareerd kunnen worden als `const`. Welke gevolgen heeft dat voor het programma?

d) Wat zou het gevolg zijn als het object `bestelling` gedeclareerd werd als `const` ?

e) Herschrijf de `main` -methode zodanig dat het geheugen dynamisch wordt aangemaakt en opgeruimd.

f) Geef de code van een methode die het aantal van de bestelling verdubbelt.

Opgave 2

Bestudeer onderstaande programmacode en beantwoord de bijbehorende vragen.

```
class Motor{
private:
   string merk;
   Zijspan* zijspan;
public:
   Motor();
   ~Motor();
   Motor(Motor& motor);
   string getMerk() const;
   void setMerk(string merknaam);
   Zijspan* getZijspan() const;
};

Motor::Motor(){
   zijspan = new Zijspan();
}

Motor::Motor(Motor motor){
   merk = motor.getMerk();
   if (motor.zijspan != NULL)
         zijspan = new Zijspan(*motor.zijspan);
}

int main(int argc, char* argv[])
{
   Motor fiets;
   fiets.setMerk("BMW");

   cout << "een " << fiets.getMerk();
   if (fiets.getZijspan() != NULL)
         cout << " met zijspan";
   return 0;
}
```

a) Geef de betekenis van het woord `private` in deze context.
Zou het programma ook werken als `private` werd vervangen door `public`?
Waarom heeft `private` de voorkeur?

b) Wat is er mis met het argument van de copy constructor van de klasse Motor ?

c) Is de syntax van het statement `zijspan = new Zijspan(*motor.zijspan);` correct?
Zo nee, wat is er fout aan?
Zo ja, waarom staan er een ster en een punt in?

d) Geef de implementatie van de destructor van de klasse `Motor`.

e) Geef het statement dat nodig is om een tweede object van de klasse `Motor` aan te maken die een kopie is van het object fiets.

f) Geef code die nodig is om een variabele aan te maken die een referentie is naar het object fiets.

Opgave 3

Gegeven zijn twee klassen, AppelStroop en SchenkStroop, die beide overerven van een superklasse Stroop, en een klasse Pannenkoek. Zie hieronder de programmacode van de klasse Stroop.

```
class Pannenkoek{
};

class Stroop{
      Pannenkoek pannenkoek;
      virtual double bepaalSuiker(Stroop& Stroop) = 0;
};

class AppelStroop: public Stroop{
      bool biologisch;
      virtual double bepaalSuiker(Stroop& Stroop){...};
};

class SchenkStroop: public Stroop{
      double suikergehalte;
      virtual double bepaalSuiker(Stroop& Stroop){...};
};
```

a) Is het nodig om een *copy constructor* te schrijven voor één of meer van de bovenstaande klassen als je kopieën wil kunnen maken van de objecten? Licht je antwoord toe.

b) Is het mogelijk om met deze implementatie objecten aan te maken van één van beide subklassen AppelStroop en SchenkStroop, zonder dat er een object van de klasse Pannenkoek wordt aangemaakt?
Zo ja: leg uit hoe het komt dat dat mogelijk is.
Zo nee: geef aan hoe de code moet worden aangepast om dit mogelijk te maken.

Een programmeur wil in zijn code het volgende statements uitvoeren:

```
AppelStroop appelstroop;
Stroop stroop = appelstroop;
```

c) Is er hier sprake van het *slicing* probleem? Licht je antwoord toe.

d) Zou je antwoord op de vorige vraag veranderen als Stroop werd vervangen door Stroop& ? Licht je antwoord toe.

e) Zou de methode ook kunnen werken als Stroop niet *by reference* werd meegegeven maar *by value*? Zo nee, waarom niet? Zo ja, wat zou het voor- of nadeel daarvan zijn?

f) Wat is de betekenis van = 0 bij de methode bepaalSuiker?