DE HAAGSE HOGESCHOOL
Technology, Innovation & Society Delft

VOORBLAD SCHRIFTELIJKE TOETSEN

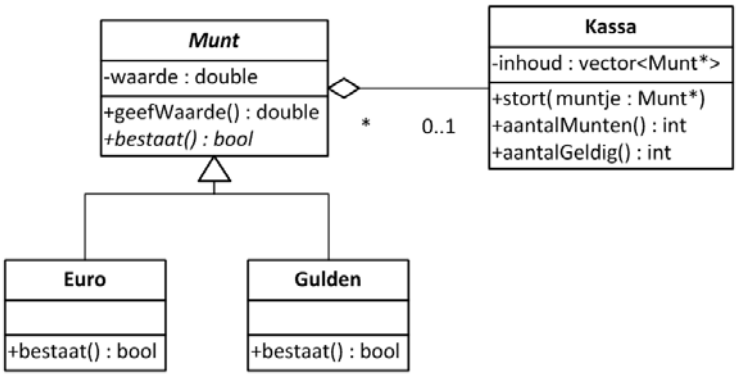
OPLEIDING	**: MECHATRONICA**
TOETSCODE	**: PLC-T1**
GROEP	**: MEH1**
TOETSDATUM	**: OEFENTENTAMEN**
TIJD	**: ~~09.00 – 10.30 uur~~** **~~11.00 – 12.30 uur~~** **~~13.00 – 14.30 uur~~** **~~15.00 – 16.30 uur~~** **~~18.00 – 19.30 uur~~** **~~20.15 – 21.45 uur~~**
AANTAL PAGINA'S (incl. voorblad)	**: 4**
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	:

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

1) Bestudeer de programmacode hieronder en de bijbehorende **`main`**-functie op de volgende pagina.
Beantwoord de vragen op de volgende pagina, die betrekking hebben op deze code.

```
// Tentamen1.cpp : main project file
#include <vector>
#include <iostream>

using namespace std;

class Munt{
// methoden
public:
	virtual double geefWaarde() {return waarde;}
	virtual bool bestaat() = 0;
// attributen
private:
	double waarde;
};

class Euro: public Munt{
	virtual bool bestaat(){
		return true;
	}
};

class Gulden: public Munt{
	virtual bool bestaat(){
		return false;
	}
};

class Kassa{
public:
	void stort(const Munt* muntje);
	int aantalMunten();
	int aantalGeldig();
private:
	vector<const Munt*> inhoud;
};

void Kassa::stort(const Munt* muntje){
	// TODO
}

int Kassa::aantalMunten(){
	// TODO
}

int Kassa::aantalGeldig(){
	// TODO
}
```



```
int main()
{
	Kassa* kas = new Kassa();
	Euro euro;
	kas->stort(&euro);
	kas->stort(new Euro());
	kas->stort(new Gulden());

	int geldig = kas->aantalGeldig();

	cout << geldig << endl;

	return 0;
}
```

a) Is het mogelijk om in dit programma objecten te creëren van de klasse Munt?
Zo *ja*: wat zou er moeten worden aangepast om dat *onmogelijk* te maken?
Zo *nee*: wat zou er moeten worden aangepast om dat *mogelijk* te maken?

b) Waarom is het nodig om **`using namespace std`** in de programmacode op te nemen?

c) Waarom staan er elke keer als er een methode van **`kas`** wordt aangeroepen *pijlen* voor de methodenaam?

d) Hoeveel geheugenlekken zitten er in dit programma? Omschrijf (in woorden, dus zonder de code te geven) hoe dat lek of die lekken zouden moeten worden opgelost.

e) Geef de programmacode (één statement) van de methode **`stort`** .
Deze methode moet het argument dat wordt meegegeven toevoegen aan de vector **`inhoud`** .

f) Geef de programmacode (één statement) van de methode **`aantalMunten`** .
Deze methode moet het aantal munten retourneren dat in de vector is opgenomen.

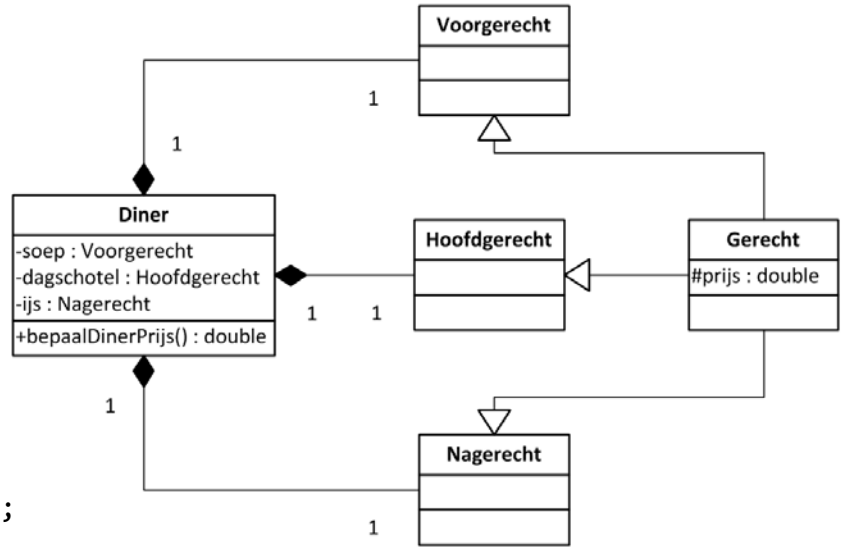
g) Geef de programmacode van de methode **`aantalGeldig`** .
Deze methode moet het aantal munten in de vector retourneren waarvoor geldt dat ze bestaan, oftewel het aantal munten waarvoor de methode **`bestaat`** de waarde **`true`** retourneert.

h) Wat is de betekenis van **`= 0`** achter de methode **`bestaat`** ?

i) Geef de programmacode (één statement) die aan de methode **`stort`** moet worden toegevoegd om de **`waarde`** van het muntje dat als argument wordt meegeven te veranderen.

2) Bestudeer de programmacode hieronder.
Beantwoord onderstaande vragen, die betrekking hebben op deze code.

```
// Tentamen2.cpp : main project file.
#include <iostream>

class Gerecht{
protected:
   double prijs;
};
class Voorgerecht: Gerecht{
};
class Hoofdgerecht: Gerecht{
};
class Nagerecht: Gerecht{
};

class Diner{
private:
   Voorgerecht soep;
   Hoofdgerecht dagschotel;
   Nagerecht ijs;
public:
   double bepaalDinerPrijs();
};

double Diner::bepaalDinerPrijs(){
   // Code nog in te vullen
};

int main()
{
    Diner hapje;
    cout << hapje.bepaalDinerPrijs() << endl;
    return 0;
}
```



a) Zijn er zinvolle redenen om aan de *default constructor* van de klasse **`Diner`** code toe te voegen? Zo ja, welke?

b) Wat moet er aan de declaratie van de klasse **`Diner`** veranderen om objecten te kunnen maken die niet altijd een **`Voorgerecht`** hebben?

c) Wat moet er aan de declaratie van de klasse **`Diner`** veranderen om objecten te kunnen maken die een willekeurig aantal gerechten van elke gang bevatten? (Een maaltijd mag dus bijvoorbeeld bestaan uit één bord soep, twee dagschotels en zeven soorten ijs.)
Volsta bij het geven van je antwoord met een beschrijving / uitleg, geen code.

d) Geef de programmacode van de methode **`bepaalDinerPrijs`** . Je mag ervan uitgaan dat de *constructors* van de klassen **`Voorgerecht`**, **`Hoofdgerecht`** en **`Nagerecht`** een default waarde toekennen aan het attribuut **`prijs`** .

3) Bestudeer de programmacode hieronder.
Beantwoord onderstaande vragen, die betrekking hebben op deze code.

```
#include <iostream>

using namespace std;

#define ORDERGROOTTE 5
#define PRIJS 12.50

class Bestelling{
private:
	double prijs;
	int aantal;
};

int main()
{
	Bestelling order[ORDERGROOTTE];
	for (int i = 0; i < ORDERGROOTTE; i++)
	{
		// Voer de orders in
		order[i].prijs = PRIJS;
		order[i].aantal = i + 1;
	}
	for (int i = 0; i < ORDERGROOTTE; i++)
	{
		// Voer de orders in
		cout << order[i].prijs << " " << order[i].aantal << endl;
	}
	return 0;
}
```

a) Waarom is **`#include <iostream>`** nodig in dit programma?

b) Welke *compiler error* levert deze programmacode op? Hoe zou je die kunnen verhelpen?

c) Aan het einde van de beide regels die beginnen met **`#define`** staat geen puntkomma. Is dat fout?
Zo *ja*: wat gaat er daardoor mis?
Zo *nee*: wat zou er mis gaan als er *wel* een puntkomma stond aan het einde van die regels?

d) Bevat dit programma een of meer geheugenlekken? Zo ja, waar?

e) Wat moet er aan het programma worden veranderd om de array met objecten van de klasse **`Bestelling`** uit pointers te laten bestaan?
(Je mag de code van het programma geven, of de verandering nauwkeurig in woorden omschrijven.)