Software-ontwikkeling en object-georiënteerde talen 1
Examen van 20 januari 2005

Ten geleide

Dit examen omvat drie vragen :

- Vraag 1 is de hoofdvraag
- Vraag 2 peilt naar je basiskennis van C. Let wel : Een conceptuele oplossing brengt ook punten op, en de examinator is geen compiler ...
- Het antwoord op vraag 3 kan kort zijn.

De typografische conventies in wat volgt zijn :
- "mensentaal" staat in 12-punt Times New Roman (dit lettertype dus)
- C++-uitdrukkingen, taalwoorden, programmafragmenten staan in `10-punt Courier New.`

Je mag de vragen beantwoorden in de volgorde die je wenst. Lees wel telkens eerst de volledige vraagstelling om verrassingen te voorkomen : de beschikbare tijd kan een kritieke factor worden bij dit examen, maar is voor iedereen dezelfde. Bezin dus voor je begint, maar niet te lang ...

Gelieve de hoeveelheid papier die je produceert te beperken tot het essentiële.

Veel succes.

HT

Vraag 1 – Het uitgeversbedrijf

Beschouw een uitgeversbedrijf, dat zowel boeken als tijdschriften uitgeeft.
- een boek heeft een titel, een auteur en een verschijningsdatum
- elk tijdschrift heeft een titel
- van een tijdschrift verschijnen nummers, met een periodiciteit typisch voor dat tijdschrift (wekelijks, tweewekelijks, maandelijks, ...)
- elk nummer heeft een volgnummer, een verschijningsdatum, en bevat een aantal artikels
- elk artikel heeft een titel en een auteur
- een auteur heeft een naam, een adres en een bedrag (saldo nog aan de auteur te betalen)
- de auteur van een boek heeft een contract met een uitgever; dat contract is opgesteld op een bepaalde datum en bepaalt welk percentage de auteur ontvangt op de verkoop als vergoeding
- auteurs van artikels in tijdschriften zenden spontaan hun artikel in. De datum van inzending wordt bijgehouden. Na nazicht door de redactie wordt beslist of het artikel al dan niet aanvaard wordt. De datum van deze beslissing wordt eveneens bijgehouden. Per artikel ontvangt de auteur een vergoeding, naargelang de omvang van het artikel (aantal gedrukte lijnen) maar onafhankelijk van het aantal verkochte tijdschrijften.

Ontwerp de nodige C++ klassen die dit kunnen implementeren, in het bijzonder hun attributen en relaties. Doe dit op een zo hoog mogelijk niveau van abstractie en/of genericiteit. Verantwoord uw ontwerpsbeslissingen. Een grafische voorstelling van je ontwerp kan zeer verhelderend zijn, maar hoeft niet. Als je geen UML notatie gebruikt, geef dan een verklaring van de grafische symbolen.

Enkel de attributen van de klassen, die voortvloeien uit bovenstaande gegevens, moeten aangegeven worden. Lidfuncties, andere dan constructoren en destructoren, moeten niet vermeld, noch geïmplementeerd worden.

Hulpklassen die beschikbaar zijn in standaard C++ , inclusief STL, mogen (moeten indien nodig !) uiteraard gebruikt worden.

Let wel : er is geen eenduidig antwoord op deze vraag. Bespreek dus je ontwerpsbeslissingen. Probeer er vooral geen dingen bij te fantaseren. Het is vooral van belang dat je aantoont hoe je zulk een probleem aanpakt.

Vraag 2. Basis C.

Nota : in deze vraag gaat het om C, en niet om C++. Enige kennis van het taalwoord `static` helpt je een eind op de goede weg.

De bedoeling is de definitie en implementatie van een module, die een lijst van strings beheert. Deze strings zijn van een willekeurige lengte, bepaald door de gebruiker van de module. Intern wordt de lijst voorgesteld als een array van pointers (type `char *`). Zowel deze array als de strings worden op de heap opgeslagen. Geheugen op de heap wordt enkel gealloceerd als er behoefte aan is.
Het aantal strings dat op elk ogenblik in de lijst zit, wordt intern in de module bijgehouden. Dat geldt ook voor de locatie (adres) van de array van pointers, en het maximaal aantal strings dat op een bepaald ogenblik in de lijst kan opgeslagen worden.

Strings worden aan de lijst toegevoegd in de volgorde waarop de gebruiker ze opgeeft.

Volgende functionaliteit is voorzien :

```
int lijst_init(int aantal) ;
```

Initialiseer een lijst die plaats biedt voor `aantal` strings.
Functiewaarde : 1 = fout
0 = in orde

```
int lijst_add(char * string) ;
```

Voeg string `string` toe aan de lijst.
Functiewaarde : 1 = fout (bv. lijst vol)
0 = in orde

```
char * lijst_get(int plaats) ;
```

Functiewaarde :
- een pointer naar de string op plaats `plaats` in de lijst,
- ofwel een null-pointer als die plaats onbezet is

```
void lijst_delete(void) ;
```

Vernietig de lijst.

De enige vraag is : geef de implementatie van bovenstaande functies.

Vraag 3 – Principes van object-oriëntatie in C++

In C++ is het mogelijk dat er meerdere functies zijn die dezelfde naam hebben. Er zijn twee mechanismes in de taal die dit ondersteunen : *overloading* en herdefinitie van functies (bvb. om polymorfisme te ondersteunen).

Bespreek het onderscheid tussen beide, o.a. qua toepassingsgebied en *binding.* Is de aanwezigheid van *overloading* in de taal strikt noodzakelijk opdat C++ zou voldoen aan de basisprincipes van object-oriëntatie ?