

Examen Software Ontwikkeling I  
2e Bachelor Informatica  
Academiejaar 2006-2007  
23 Januari, 2007

**\*\*BELANGRIJK\*\*** : Schrijf je naam onderaan dit blad

## Deel I. Gesloten Boek deel (8u30-8u45)

Leg volgende 3 begrippen kort en bondig uit :

a. Concurrent Version System (CVS)

b. Make-bestand

c. Cross-compilatie

Examen Software Ontwikkeling I  
2e Bachelor Informatica  
Academiejaar 2006-2007  
23 Januari, 2007

**\*\*BELANGRIJK\*\***

1. Schrijf je naam onderaan op elk blad
2. Dit deel handelt enkel over C++ (deel III enkel over C)
3. Veel succes!

## Deel II. Open Boek gedeelte op papier (8u45-9u45)

Voor het beheer van een voertuigpark in een bedrijf (alle bedrijfswagens en bestelwagens) gebruikt men C++ software. Bedrijfswagens worden gecategoriseerd volgens het aantal passagiers en bestelwagens volgens de dimensies van hun laadruimte. Bijgevoegd vind je het header-bestand `voertuig.h` met de declaraties van de klassen.

Vraag 1:

[op 2 punten van de 20]

Zoals aangegeven in het header-bestand, wordt in de klasse `VoertuigPark` een array van pointers naar `Voertuigen` bijgehouden, waarbij het aantal array-elementen in de constructor doorgegeven wordt en opgeslagen in het private attribuut `maximaalAantalVoertuigen`.

Schrijf de constructor en destructor van de klasse `VoertuigPark`.

a. Constructor:

```
VoertuigPark::VoertuigPark(int maximaalAantalVoertuigen)
```

b. Destructor:

```
VoertuigPark::~VoertuigPark()
```

Geef ook de implementatie van een constructor en de destructor van de klasse Voertuig.

c. Constructor:

```
Voertuig::Voertuig (const string& inschrijvingsNummer,  
                    const string& merk,  
                    const string& fabricant,  
                    float pk)
```

d. Destructor:

```
Voertuig::~~Voertuig()
```

Vraag 2:

[op 1 punt van de 20]

Leg uit waarom:

1. de methoden `printEigenschappen()` en `berekenFiscaleIndex()` in de klasse `Voertuig` als virtual gedefinieerd zijn en gelijk aan 0.
2. in de klasse `VoertuigPark` een array van pointers naar `Voertuigen` bijgehouden wordt (en niet een array van `Voertuigen`).

1.

2.

Vraag 3:

[op 2 punten van de 20]

Men wil in het publieke deel van de klasse `VoertuigPark` een operator `+` definiëren, zodat men objecten van dit type kan samennemen als bedrijven samensmelten.

Zorg ervoor dat volgende code ook mogelijk is:

```
vp_new = vp1+vp2+vp3;
```

```
/*vp_new, vp1, vp2, vp3 zijn geïntantieerde objecten van de  
VoertuigPark klasse */
```

a. Schrijf de `+` operator (signatuur + implementatie) van de klasse `VoertuigPark` :

b. Leg het belang van het return type van deze operator uit.

Vraag 4:

[op 2 punten van de 20]

Men wil in de `VoertuigPark` klasse een STL container gebruiken in plaats van de array als privaat attribuut. Voor gemakkelijk opzoeken van voertuigen op basis van hun inschrijvingsnummer kiest men voor een associatieve container en meerbepaald een map (sleutel: `inschrijvingsNummer` en waarde: pointer naar `Voertuig` object).

a. Schrijf de declaratie van deze map als privaat attribuut van de `VoertuigPark` klasse:

b. Schrijf de methode `printEigenschappenAlleVoertuigen` in het geval van implementatie met de STL container `map`. Maak gebruik van een iterator.

`void VoertuigPark::printEigenschappenAlleVoertuigen()`

KLADPAPIER:

Examen Software Ontwikkeling 1  
2e Bachelor Informatica  
Academiejaar 2006-2007  
23 Januari, 2007

**\*\*BELANGRIJK\*\***

1. Schrijf je naam in commentaar bovenaan elk bestand !!
2. Maak je project in de folder C:\SOI\ aan !
3. Bewaar je werk regelmatig in C:\SOI\ !
4. Deze opgave handelt over C, schrijf dus ook enkel C code en stel je compiler daarop in (zie appendix)
5. Aanpassen van de bijgeleverde header files is niet toegelaten, de main mag je wel aanpassen maar dat is in principe niet nodig
6. Warnings van MS Visual C++ Express Edition over het gebruik van strcpy mag je verwaarlozen

### Deel III. Open Boek op computer (10u00-12u00)

Implementeer **in C** een adresboek (bijvoorbeeld voor een gsm) dat contacten bijhoudt met hun nummers. Maak hiertoe gebruik van de bijgeleverde header files en het testprogramma.

#### 1. Contact record (6 / 20 punten):

Gegeven de volgende header file, contact\_record.h:

```
enum number_type {MOBILE = 0, PRIVATE = 1, WORK = 2};

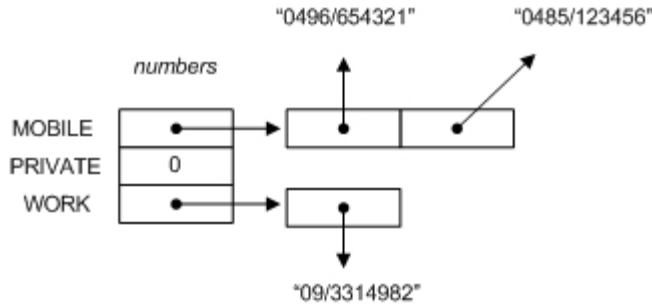
struct contact_record {
    char* contact_name;
    char*** numbers;
    int* numbers_size;
};

typedef struct contact_record contact;

contact init_record(char* contact_name);
void add_number(contact* c, enum number_type numbertype,
               char* number);
void delete_number(contact* c, char* number);
void delete_contact(contact* c);
void print_contact(contact* c);
```

Een contact heeft dus een naam en een array `numbers` die de mobiele, privé- en werknummers bijhoudt.

De interne structuur van de `numbers` array kan als volgt schematisch worden weergegeven:



De `numbers_size` array houdt bij hoeveel nummers van elke soort aanwezig zijn in `numbers`.

Gevraagd is de functies `init_record`, `add_number`, `delete_number`, `delete_contact` en `print_contact` te implementeren in een file `contact_record.c`. In het bijgeleverde bestand `main.c` staat reeds code om deze functies te testen.

## 2. Contact lijst (3 / 20 punten):

Gegeven de volgende header file, `contact_list.h`:

```
#include "contact_record.h"

struct listhdr {
    struct listhdr *prev, *next;
};

void list_init(struct listhdr *list);
void list_addfront(struct listhdr *list, struct listhdr *item);
void list_remove(struct listhdr *list, struct listhdr *item);

struct contactlist_record {
    struct listhdr list;
    contact c;
};
```

Maak een dubbelgelinkte lijst van `struct listhdr` elementen door bovenstaande functies te implementeren. Deze functies verwachten als eerste argument “het anker” van de gelinkte lijst. Dit is een element dat tezelfdertijd het begin als het einde van de lijst markeert. Het anker is wel geen “echt” element, het behoort niet tot de elementen van de lijst. De `next` pointer wijst naar het eerste element in de gelinkte lijst en de `prev` pointer wijst naar het laatste. Bij een lege lijst wijzen beide pointers naar “het anker” zelf.

## 3. Adresboek (2,5 / 20 punten)

In `contact_list.h` staat nog 1 extra functie, namelijk:

```
char** get_numbers(struct listhdr *list, char* contactname,
                  enum number_type, int* number_count);
```



Deze functie zoekt de nummers van een bepaald type van een bepaald contact op in de opgegeven lijst, en geeft deze terug in een array. Het aantal nummers dat in de array bevat zit, wordt in de functie ingesteld in de variabele `number_count`. Implementeer de functie `get_numbers`.

Hint: je kan een `struct contactlist_record` casten naar een `struct listhdr` (omdat het eerste element een `struct listhdr` is).  
Voor het testen van deel 2 en 3 is in het bijgeleverde bestand `main.c` reeds code in commentaar voorzien om deze functies te testen.

Veel succes!

Appendix (voor de volledigheid):

Project aanmaken in MS Visual C++ EE:

- Kies **File – New – Project** en selecteer als Project Type **Win32**.
- Als Template selecteer je **Win32 Console Application**.
- Onderaan vul je dan een projectnaam, de opslaglocatie (C:\SOI) en de naam van de (nieuwe) Solution in.
- Na bevestigen wordt een Win32 Application wizard opgestart. Druk op Next, check of Console Application als Application type wordt weergegeven en vink **Empty Project** aan. Vervolgens kan je deze wizard beëindigen. Er is nu een nieuw project en een nieuwe Solution aangemaakt.
- De header files toevoegen kan je door rechts te klikken op de '**Header files**' folder en **Add – Existing Item** aan te klikken

Compiler instellen op C code:

**Properties – Configuration Properties – C/C++ - Advanced – Compile As...**  
=> compile as C

KLADPAPIER:

Naam: .....