Klausur

Programmieren 2

Sommersemester 2004

02.07.2004

Name:

Vornam

Matr.Nr.

Studieng:

Aufgabe	Punkte	Erreichte Punkte
1	16	
2	16	
3	18	
4	14	
Insgesamt	64	
Note:		

Arbeiten Sie gewissenhaft und lesen Sie die Aufgabenstellung sorgfältig durch. Beantworten Sie die Fragen eindeutig!

Hilfsmittel: Bücher Ihrer Wahl, Skript, Folien und Ihre Unterlagen. Es sind keine Altmeister und keine Taschenrechner, Laptop, PDA oder ähnliches erlaubt!!!

Aufgabe 1 (16 P)

Gegeben ist folgendes Programmstück.

```
struct verwaltung{
 int * dim[2];
 };
void main () {
 struct verwaltung d1, d2;
 int *daten1, *daten2,i;
 daten1 = (int *) calloc(4,sizeof(int));
 daten2 = (int *) calloc(4,sizeof(int));
 for (i = 0; i < 3; ++i) {
   daten1[i] = i;
   daten2[i] = i*i;
 }
 d1.dim[0] = daten2;
 d1.dim[1] = d1.dim[0]+2;
 d2 = d1;
 d2.dim[0] += 2;
 --d2.dim[1];
 d1.dim[0] = &daten1[2];
 d1.dim[1] = d1.dim[0] - 2;
 for (i = 0; i < 2; ++i) {
   printf("*d1.dim[%d]: %d \n",i,*(d1.dim[i]));
   printf("*d2.dim[%d]: %d \n",i,*(d2.dim[i]));
 }
 *(d1.dim) = daten1;
 *(d2.dim) = &d1.dim[0][2];
 d1.dim[1] =  ++*(d1.dim);
 *(d2.dim+1) = &daten2[1];
 ++(*(*(d2.dim)));
 d1.dim[0][1] = &(d1.dim[1][2]) - d1.dim[1];
 for (i = 0; i < 2; ++i) {
   printf("d1.dim[%d][%d]: %d\n",i,i,d1.dim[i][i]);
   printf("d2.dim[%d][%d]: %d\n",i,i,d2.dim[i][i]);
 }
}
```

Geben Sie die Ausgaben an. Beachten Sie die unterschiedlichen printf-Anweisungen!!!

Aufgabe 2 (16 P)

Gegeben ist folgender unvollständig implementierter LIFO-Stack:

```
class stack{
  private:
    int size;
    int top;
    int* stck;
    // Methode resize muss noch ergänzt werden

  public:
    // Konstruktor(en) & Destruktor fehlen ?

    void push(int wert) {
      if (top == size)
        resize(size*2);
      stck[top] = wert;
      ++top;
    }

    int pop(void) {
      int erg = 0;
      if (top > 0) {
        --top;
        erg = stck[top];
      }
      else
        cerr << "Fehler in pop: Stack ist leer!\n";
      return erg;
    }
};
```

Ergänzen Sie folgende fehlenden Methoden:

- Einen Konstruktor mit einem default-Parameter der eine Stack der Gröbse 16 vereinbart, wenn kein Parameter angegeben wurde.
- Wenn nötig einen Copy-Konstruktor. Wenn kein Copy-Kostruktor nötig ist, dann begründen Sie dies!
- Wenn nötig einen Destruktor. Wenn kein Destruktor nötig ist, dann begründen Sie dies!

Aufgabe3 (18 P)

Gegeben ist folgende Datenstruktur zum Aufbau eines Verzeichnisbaums:

```
typedef struct verzeichnis {
  char name[64];
  struct verzeichnis *next;
    // Verweis zum nächsten Verzeichnis auf
    // der gleichen Ebene (gleicher Vater!)
  struct verzeichnis *sohn;
    // Verweis zu den Söhnen, die per *next als
    // einfach verkettete Liste gespeichert sind
} t_verzeichnis;
```

Für diese Datenstruktur ist eine rekursiv arbeitende Funktion copy () gesucht, die einen Zeiger auf eine Verzeichnisbaum übergeben bekommt und als Rückgabewert einen Zeiger auf eine Kopie des Verzeichnisbaums liefert. Geben Sie:

a) Die Schnittstelle der Funktion an.
b) Die Implementierung der Funktion an.

Nutzen Sie folgende grobes Vorgehen in Ihrer Funktion:

1. Wenn etwas zu kopieren ist
   i. Erzeuge neues Verzeichnis
   ii. Kopiere Name des Verzeichnisses
   iii. Wenn ein Sohn vorhanden ist
      1. Kopiere ersten Sohn
      2. Solange weitere Söhne in Sohn-Liste
         a. Kopiere nächsten Sohn
2. Gib erzeugte Kopie zurück.

Aufgabe 4 (14 P)

Für einen Reifenhandel soll dessen Verkauf und Lager im Rechner modelliert werden. Hierzu werden folgende Klassen benötigt:

a) Die abstrakte Klasse Artikel
Ein Objekt dieser Klasse besitzt folgende **öffentliche** Datenfelder:
- Einen Namen (maximal 16 Zeichen)
- Einen Einkaufspreis
- Die verfügbare Anzahl

Sowie die folgende abstrakte Methode:
- Verkaufspreis()
  In dieser parameterlosen Methode wird je nach Ausprägung des Artikels auf eine unterschiedliche Art und Weise der Verkaufspreis berechnet.

b) Die Klasse Felge
Diese Klasse erweitert die in Artikel definierten Datenfelder um folgendes Datenfeld:
- Material (Stahl oder Alu)

Weiterhin wird die Methode Verkaufspreis implementiert, indem der Einkaufspreis mit 1,5 multipliziert wird.

c) Die Klasse Reifen
Diese Klasse erweitert die in Artikel definierten Datenfelder um folgendes Datenfeld:
- Dimension (String mit maximal 8 Zeichen)

Weiterhin wird die Methode Verkaufspreis implementiert, indem der Einkaufspreis mit 1,3 multipliziert wird.

d) Die Klasse Rad
Diese Klasse nutzt neben den in Artikel definierten Datenfeldern noch ein privates Datenfeld der Klasse Felge und ein privates Datenfeld der Klasse Reifen. Der Verkaufspreis wird ermittelt, indem der Verkaufspreis der Felge zum Verkaufspreis des Reifens addiert wird und von dieser Summe dann 10% abgezogen werden.

Aufgabe: Implementieren Sie die genannten Klassen mit den angegebenen Datenfeldern und Methoden.

Wichtigere Hinweise:
- Weitere (durchaus benötigte) Methoden und Datenfelder brauchen Sie nicht zu implementieren.
- Es wird kein Hauptprogramm benötigt.
- Das Einlesen und Initialisieren der Datenfelder braucht nicht implementiert zu werden.