Prof. Dr. Stefan Brass
Dipl.-Inform. Annett Thüring
Institut für Informatik
MLU Halle-Wittenberg

01. Februar 2006

Vorlesung "Objektorientierte Programmierung" — Probeklausur III —

Bitte bearbeiten Sie diese Probeklausur zu Hause im Rahmen Ihrer Prüfungsvorbereitung. Zu dieser Probeklausur wird eine Lösung ins Netz gestellt. Falls noch Fragen auftreten sollten, können Sie sich selbstverständlich an die Übungsleiterin oder den Dozenten wenden. (Bitte schauen Sie aber möglichst vorher auf der Webseite, ob Ihre Frage schon beantwortet ist. Beachten Sie möglichst auch die Sprechstunden.)

Aufgabe	Punkte	von (Max.)	Zeit
1 (Rekursive Funktion)		1	8 min
2 (Verständnisfragen)		4	12 min
3 (Vererbung)		1	5 min
4 (Programm: Klassendefinition)		8	20 min
5 (Fehlersuche, Speicherverwaltung)		2	5 min
Summe		16	50 min

Hinweise:

- Bearbeitungsdauer: 60 Minuten
- Skript, Bücher, Notizen sind erlaubt. Notebooks, PDAs, etc. dürfen nicht verwendet werden. Natürlich dürfen Sie auch nicht mit dem Nachbarn sprechen oder Papiere und andere Dinge austauschen. Notfalls rufen Sie eine Aufsichtsperson zur Kontrolle.
- Bitte benutzen Sie den vorgegebenen Platz. Wenn Sie auf die Rückseite ausweichen müssen, markieren Sie klar, daß es eine Fortsetzung gibt.
- Fragen Sie, wenn Ihnen eine Aufgabe nicht klar ist!
- **ACHTUNG!** Diese Probeklausur ist möglicherweise einfacher als die eigentliche Klausur. Falls Sie mit der Probeklausur keine Schwierigkeiten haben, heißt das nicht, daß Sie sich für die eigentliche Klausur nicht vorbereiten müssen.

Aufgabe 1 (Rekursive Funktion) 1 Punkt

Was gibt das folgende Programm bei der Eingabe `a = 4` und `b = 2` aus? Es empfiehlt sich, zur Lösung der Aufgabe einen Rekursionsbaum zu zeichnen (welcher Funktionsaufruf erzeugt welche anderen Funktionsaufrufe, wobei die Funktionsaufrufe durch die aktuellen Eingabeparameter dargestellt sind).

```
#include <iostream>
using namespace std;

unsigned long f(unsigned long a, unsigned long b);

int main()
{
  unsigned long a, b;

  cout << "Bitte geben Sie a ein: ";
  cin >> a;

  cout << "Bitte geben Sie b ein: ";
  cin >> b;

  cout << "Ergebnis = " << f(a, b) << endl;

  return 0;
}

unsigned long f(unsigned long a, unsigned long b)
{
  if (a == b || b == 0)
    return 1L;

  if (a < b)
    return 0L;

  return f(a - 1, b) + f(a - 1, b - 1);
}
```

Ausgabe bei `a=4, b=2`: ____________________

Aufgabe 2 (Verständnisfragen) 4 Punkte

Beantworten Sie bitte folgende Fragen:

1. Was ist der Unterschied zwischen einem Feld (Array) und einem Zeiger?

2. Wie ist der Zusammenhang zwischen Zeigern und Referenzen?

3. In welcher Situation braucht man einen Kopierkonstruktor?

4. Wie viele Rückgabewerte kann eine Funktion haben?

Aufgabe 3 (Vererbung) **1 Punkt**

Was gibt das folgende Programm aus?

```
#include <iostream>
using namespace std;

class A {
    public:
        int f() { return 1; }
        virtual int g() { return 2; }
};

class B : public A {
    public:
        int f() { return 3; }
        int g() { return 4; }
};

int main()
{
        A a;
        B b;
        A* p1 = &a;
        A* p2 = &b;
        B* p3 = &b;

        cout << "p1: " << p1-> f() << ' ' << p1->g() << '\n';
        cout << "p2: " << p2-> f() << ' ' << p2->g() << '\n';
        cout << "p3: " << p3-> f() << ' ' << p3->g() << '\n';

        return 0;
}
```

p1: __________________

p2: __________________

p3: __________________

Aufgabe 4 (Programm) 8 Punkte

Definieren Sie eine Klasse `Strecke` zur Darstellung einer Strecke im zweidimensionalen kartesischen Koordinatensystem. Diese besitzt die folgenden vier `private`-Datenelemente: `start_x, start_y, end_x, end_y` vom Typ `double`.

Deklarieren und definieren Sie folgende drei Methoden als `public`:

- `set_start`:
  Diese Methode soll den den Startpunkt (`start_x, start_y`) der Strecke initialisieren. Die Werte `x` und `y` für den Startpunkt werden als Parameter übergeben.
- `set_end`:
  Diese Methode soll entsprechend den Endpunkt (`end_x, end_y`) der Strecke initialisieren. Die Werte `x` und `y` für den Endpunkt werden als Parameter übergeben.
- `get_laenge`:
  Diese Methode soll die Länge der Strecke liefern (als Returnwert vom Typ `double`). Die Berechnung der Länge erfolgt über den Satz des Pythagoras, d.h.

  $$\sqrt{(\texttt{end\_x} - \texttt{start\_x})^2 \; + \; (\texttt{end\_y} - \texttt{start\_y})^2}$$

  Verwenden Sie zur Berechnung der Quadratwurzel die Funktion

  `double sqrt(double x).`

  Diese Funktion wird in der Header-Datei `math.h` deklariert.

Bemühen Sie sich um Verständlichkeit und guten Programmierstil (es können auch Punkte für schlechten Stil abgezogen werden).

Schreiben Sie die Klassendefinition vollständig und ohne Syntaxfehler. Sie brauchen keinen Konstruktor und Destruktor zu definieren. Schreiben Sie bitte nur die Klassendefinition, kein Hauptprogramm.

Aufgabe 5 (Fehlersuche, Speicherverwaltung) 2 Punkte

Das folgende Programm enthält drei Fehler im Umgang mit Speicher. Diese Fehler werden durch den Compiler nicht unbedingt erkannt, können dann aber zur Laufzeit des Programmes zum Absturz oder anderem Fehlverhalten führen.

Bitte geben Sie zwei dieser Fehler an (es ist möglicherweise schwierig, alle drei zu entdecken). Falls Sie mehr als zwei Fehler angeben, werden nur die ersten beiden gewertet.

```
#include <iostream>
using namespace std;

const int LEN=32;
const int ITER=1000000;
typedef struct Bell
{
    char *ding;
    char *dong;
};

char* f1 (void)
{
   char carray[LEN];
   for(int i = 0; i < LEN; i++)
      carray[i] = 'a';
   carray[LEN] = '\0';
   return carray;
}

char* f2()
{
  return new char[1024];
}

int main (void)
{
   Bell *f;

   for(int i = 0; i < ITER; i++) {
      f = new Bell;
      f->ding = f1();
      f->ding[0] = 'b';
      f->dong = f2();
      f->dong[0] = 'c';
      f->dong[1] = '\0';
      cout << f->ding << f->dong << '\n';
      delete f;
   }
   return 0;
}
```

Fehler 1:

Zeile: ______

Begründung: ______________________________

____

Fehler 2:

Zeile: ______

Begründung: ______________________________

____