Prof. Dr. Stefan Brass
Institut für Informatik
MLU Halle-Wittenberg

19. Dezember 2006

Vorlesung "Objektorientierte Programmierung" — Probeklausur I —

Bitte geben Sie diese Probeklausur anonym ab. Damit Sie die Klausur bei der Rückgabe leichter finden können, tragen Sie bitte in die dafür vorgesehenen Kästchen eine dreistellige, zufällige Zahl ein. Merken Sie sich diese Zahl. Selbstverständlich ist es dennoch möglich, daß verschiedene Teilnehmer die gleiche Zahl wählen: Schauen Sie sich bei der Rückgabe ggf. nochmal Ihr Programm (Aufgabe 4) an.

Zahl zur Identifikation bei Rückgabe: ☐☐☐

Aufgabe	Punkte	von (Max.)	Zeit
1 (Arithmetischer Ausdruck)		1	3 min
2 (Logische Bedingung)		1	3 min
3 (Variablen, Pointer, Referenzen)		1	5 min
4 (Programm)		5	15 min
5 (Fehlersuche)		2	5 min
Summe		10	31 min

Hinweise:

- Bearbeitungsdauer: 35 Minuten
- Skript, Bücher, Notizen sind erlaubt. Notebooks, PDAs, etc. dürfen nicht verwendet werden.
- 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 erste Probeklausur ist einfacher als die eigentliche Klausur. Sie war ja nicht angekündigt und soll Ihnen und dem Dozenten einen ersten Eindruck von Ihrem Leistungsstand geben. Außerdem liegt ja auch noch ein wesentlicher Teil der Vorlesung vor uns. Falls Sie mit der Probeklausur keine Schwierigkeiten haben, heißt das keineswegs, daß Sie sich für die eigentliche Klausur nicht vorbereiten zu brauchen.
- Bitte nehmen Sie auch an der Umfrage auf der letzten Seite teil.

Aufgabe 1 (Arithmetischer Ausdruck) 1 Punkt

Was gibt das folgende Programm aus?

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

int main()
{
        int i = 7;
        int j = 10;

        int n = 5 + 3 * i % j << 1;
        cout << n;

        return 0;
}
```

Ausgabe: ______________________________

Hier nochmal die Tabelle mit den Prioritätsstufen der Operatoren:

18	`::`	Gültigkeitsbereich
17	`++` (Postfix), `.`, `->`, `[]`, `f()`, ...	Postfix-Operatoren
16	`-` (unär), `!`, `*` (deref), `++` (Präfix), ...	Präfix-Operatoren
15	`.*`, `->*`	Zeiger auf Komp.
14	`*`, `/`, `%`	Multiplikation etc.
13	`+`, `-`	Addition, Subtraktion
12	`<<`, `>>`	Shift etc.
11	`<`, `<=`, `>`, `>=`	kleiner etc.
10	`==`, `!=`	gleich, verschieden
9	`&`	Bit-und
8	`^`	Bit-xor
7	`|`	Bit-oder
6	`&&`	und
5	`||`	oder
4	`?:`	Bedingter Ausdruck
3	`=`, `+=`, `-=`, `*=`, `/=`, ...	Zuweisungen
2	`throw`	Exception auslösen
1	`,`	Sequenz

Bei gleicher Priorität sind alle Operatoren (außer Präfixoperatoren und Zuweisungen) implizit von links geklammert (linksassoziativ).

Aufgabe 2 (Logische Bedingung) 1 Punkt

Was gibt das folgende Programm aus?

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

int main()
{
        int i = 7;
        int j = 10;

        if(i == j)
                cout << "A";
        else if(i > j || j < 20)
                cout << "B";
        else if(i && j >= i)
                cout << "C";
        else
                cout << "D";

        return 0;
}
```

Ausgabe: ______________________________

Aufgabe 3 (Variablen, Pointer, Referenzen) 1 Punkt

Was gibt dieses Programm aus?

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

int main()
{
        int i = 1;
        int j = 2;
        int a[3] = { 3, 4, 5 };
        int &r = i;
        int *p = &j;
        int *q = a;
        r++;
        a[i] = j;
        *(q+2) += *p;
        cout << a[0] << "," << a[1] << "," << a[2] << "\n";
        return 0;
}
```

Ausgabe: ______________________________

Aufgabe 4 (Programm) 5 Punkte

Schreiben Sie ein Programm, das eine positive Zahl einliest, und dann eine "Pyramide" (oder "Tannenbaum") der entsprechenden Höhe ausgibt. Für die Eingabe 3 soll z.B. folgende Ausgabe erscheinen (drei Zeilen):

```
    *
  * * *
* * * * *
```

Für die Eingabe 1 würde entsprechend nur ein einzelner Stern gedruckt. Da in der Vorlesung die Ein-/Ausgabe noch nicht gründlich besprochen wurde, dürfen Sie voraussetzen, daß der Benutzer korrekt eine positive Zahl eingibt (und auch kein sonstiger Fehler bei der Eingabe auftritt). Bemühen Sie sich um Verständlichkeit und guten Programmierstil. Schreiben Sie das Programm vollständig und möglichst ohne Syntaxfehler.

Aufgabe 5 (Fehlersuche) 2 Punkte

Das folgende Programm enthält (mindestens) 3 Fehler, die der Compiler finden sollte. Bitte geben Sie zwei dieser Fehler an (es ist möglicherweise schwierig, alle drei zu entdecken). Sie bekommen keinen Extrapunkt, wenn Sie alle drei finden. Falls Sie mehr als zwei Fehler angeben, werden nur die ersten beiden gewertet. Geben Sie bitte die Zeilennummer mit an, in der der Compiler den Fehler finden müßte, wenn er das Programm von oben nach unten liest. Die in Klammern vorangestellten Zeilennummern sind natürlich nicht Bestandteil des Programms (und zählen nicht als Fehler).

```
 (1)  #include <iostream>
 (2)  using namespace std;
 (3)
 (4)  int main()
 (5)  {
 (6)          cin >> i;
 (7)          if(i != 0)
 (8)                  cout << "i ist ";
 (9)                  cout << i << ".\n";
(10)          else
(11)                  cout << "i ist null.\n";
(12)  }
```

Fehler 1:

Zeile: ______

Begründung: ____

____

Fehler 2:

Zeile: ______

Begründung: ____

____

Umfrage

1. Konnten Sie vor dieser Vorlesung schon programmieren?

   ☐ Ja

   ☐ Etwas

   ☐ Nein

   In welchen Sprachen?

   ____

2. Wieviel Prozent des Vorlesungsstoffs ist für Sie neu?

   ☐ 100%

   ☐ 75%

   ☐ 50%

   ☐ 25%

   ☐ 0%

3. Empfanden Sie diese Probeklausur als schwierig?

   ☐ viel zu schwierig

   ☐ etwas zu schwierig

   ☐ angemessen

   ☐ eher zu leicht

   ☐ trivial

4. Was sollte in der Vorlesung verbessert werden (Kritik)?

   ____

   ____

   ____

5. Was sollte so bleiben (Lob)?

   ____