Prüfung Informatik D-MATH/D-PHYS 4. 3. 2005

Dr. Bernd Gärtner

Kandidat/in:

Name: .....

Vorname: .....

Stud.-Nr.: .....

Ich bezeuge mit meiner Unterschrift, dass ich die Prüfung unter regulären Bedingungen ablegen konnte und dass ich die untenstehenden allgemeinen Bemerkungen gelesen und verstanden habe.
Unterschrift: .....

Allgemeine Bemerkungen und Hinweise:

1. Überprüfen Sie die Vollständigkeit der ausgeteilten Prüfungsunterlagen (4 einseitig bedruckte Blätter).
2. Falls Sie während der Prüfung durch irgendeine Behinderung oder Störung beeinträchtigt werden, melden Sie dies bitte sofort der Aufsichtsperson. Spätere Klagen werden nicht akzeptiert.
3. Erlaubte Hilfsmittel: zwei A4-Blätter (doppelseitig), selbstverfasst
4. Betrugsversuche führen zu sofortigem Ausschluss und können rechtliche Folgen haben.
5. Pro Aufgabe ist höchstens eine gültige Version eines Lösungsversuch zulässig. Streichen Sie ungültige Lösungsversuche klar durch.
6. Alle Antworten müssen für den Korrektor verständlich begründet werden. Schreiben Sie die wesentlichen Lösungsgedanken in klaren Sätzen oder Stichworten hin. Unverständliche oder nicht begründete Antworten werden nicht honoriert.
7. Sie dürfen die 6 Aufgaben in beliebiger Reihenfolge lösen. Konzentrieren Sie sich jeweils auf eine Aufgabe, aber teilen Sie sich Ihre Zeit ein.
8. Nach Ablauf der Prüfungszeit verlassen Sie bitte den Raum und lassen Sie nur die Blätter auf Ihrem Platz liegen, die zur Abgabe bestimmt sind. **Diese müssen alle mit Ihrem Namen beschriftet sein.**
9. Die Prüfung ist bestanden, wenn Sie 60 von 120 erreichbaren Punkten erzielen.

Aufgabe 1 (15 Punkte). Werten Sie die folgenden Ausdrücke aus und bestimmen Sie jeweils den Wert. Geben Sie alle Zwischenschritte der Auswertung an.

(a) `3.0 / 2 + 3 / 2 * 4.0`

(b) `3 / 4 > 0 && 3.0 / 4.0 > 0 || 5.0 == -1.0f + 6.0f`

Aufgabe 2. (15 Punkte) Implementieren Sie eine Funktion

```
double pow (double x, int i);
```

zur Berechnung der i-ten Potenz einer Zahl x. Berücksichtigen Sie, dass i negativ sein kann, und dass $x^i = (1/x)^{-i}$ sowie $0^0 = 1$ gilt. Was ist die *Precondition* (Voraussetzungen für x und i) Ihrer Funktion?

Aufgabe 3. (20 Punkte) Geben Sie die *Postcondition* (was "macht" die Funktion?) der folgenden Funktion an.

```
unsigned int g (unsigned int n, unsigned int& f_) {
  if (n == 0) {
    return 0;
  }
  if (n == 1) {
    f_ = 0;
    return 1;
  }
  unsigned int f = g(n-1, f_) ;
  unsigned int ff = f + f_;
  f_= f;
  return ff;
}
```

Aufgabe 4. (20 Punkte) Gegeben sei die kontextfreie Grammatik $G = (V, \Sigma, S, R)$ mit Nichtterminalsymbolen (Variablen) $V = \{\mathtt{p}\}$, Terminalsymbolen $\Sigma = \{(,), *, \mathtt{1}\}$, Startsymbol $S = \{\mathtt{p}\}$ und den Regeln

$$\begin{aligned} \mathtt{p} &\rightarrow \mathtt{p} * \mathtt{p} \\ \mathtt{p} &\rightarrow (\mathtt{p}) \\ \mathtt{p} &\rightarrow \mathtt{1} \end{aligned}$$

Geben Sie für die folgenden Wörter jeweils eine Ableitung an oder beweisen Sie, dass das jeweilige Wort nicht mittels G ableitbar ist.

(a) `1*(1*1)*1`

(b) `(1*1)(1*1)`

(c) `((1*1)*1)*1`

Aufgabe 5. (20 Punkte) Wir haben drei Varianten kennengelernt, eine Funktion `f` mit einen formalen Parameter vom Typ `T` (und Rückgabetyp zum Beispiel `int`) zu deklarieren:

```
int f(T x);
int f(T& x);
int f(const T& x);
```

Beschreiben Sie die jeweilige Bedeutung der drei Varianten und geben Sie Anwendungsszenarien für alle drei Varianten an.

Aufgabe 6. (30 Punkte) Die *dreiwertige* Logik enthält ausser den Wahrheitswerten *true* und *false* noch den Wert *unknown*. Die logischen Operatoren $\wedge$ (logisches und) und $\vee$ (logisches oder) sind wie folgt definiert.

$\wedge$	*false*	*unknown*	*true*
false	*false*	*false*	*false*
unknown	*false*	*unknown*	*unknown*
true	*false*	*unknown*	*true*

$\vee$	*false*	*unknown*	*true*
false	*false*	*unknown*	*true*
unknown	*unknown*	*unknown*	*true*
true	*true*	*true*	*true*

Wir wollen in `C++` eine Klasse `Tribool` implementieren, die dreiwertige Logik realisiert. Die logischen Operatoren sollen als globale Operatoren implementiert werden. Der `public`-Teil einer entsprechenden Klassendeklaration sieht wie folgt aus.

```
class Tribool {
public:
  // Initialisierung mit Wahrheitswert "unknown"
  // POST: *this wurde mit "unknown" initialisiert
  Tribool ();

  // Benutzerdefinierte Konversion von bool nach Tribool
  // POST: *this wurde mit dem Wert von b initialisiert
  Tribool (bool b);

  // logisches und
  // POST: gibt den Wert von "x und y" gemaess obiger Tabelle zurueck
  friend Tribool operator&& (Tribool x, Tribool y);

  // logisches oder
  // POST: gibt den Wert von "x oder y" gemaess obiger Tabelle zurueck
  friend Tribool operator|| (Tribool x, Tribool y);

private:
  // Repraesentation eines Tribools
  ...
};
```

(a) Ergänzen Sie den `private`-Teil, indem Sie eine geeignete Repräsentation (im Hinblick auf Teil (b)) wählen. Hier gibt es sicher mehrere sinnvolle Möglichkeiten.

(b) Implementieren Sie (unter Verwendung der Repräsentation aus (a)) die beiden Konstruktoren

```
Tribool::Tribool();
Tribool::Tribool (bool b);
```

sowie die beiden logischen Operatoren

```
Tribool operator&& (Tribool x, Tribool y);
Tribool operator|| (Tribool x, Tribool y);
```