# Prüfung — Informatik D-MATH/D-PHYS
## 31. 08. 2007 09:00–11:00

Dr. Bernd Gärtner, Prof. Juraj Hromkovič

**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 (ein beidseitig bedrucktes Blatt und ein einseitig bedrucktes Blatt, bestehend aus 1 Deckseite und 2 Aufgabenseiten mit insgesamt 6 Aufgaben)!
2. Falls Sie während der Prüfung durch irgendeine Behinderung oder Störung beeinträchtigt werden, melden Sie dies bitte sofort der Aufsichtsperson! Spätere Klagen können nicht akzeptiert werden.
3. Erlaubte Hilfsmittel: **keine**.
4. Betrugsversuche führen zu sofortigem Ausschluss und können rechtliche Folgen haben.
5. Pro Aufgabe ist höchstens eine gültige Version eines Lösungsversuchs zulässig. Streichen Sie ungültige Lösungsversuche klar durch! Schreiben Sie auf separate Blätter, nicht auf die Aufgabenblätter!
6. Sie dürfen die Aufgaben in beliebiger Reihenfolge lösen. Konzentrieren Sie sich jeweils auf eine Aufgabe, aber teilen Sie sich Ihre Zeit ein!
7. 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. Die Prüfungsblätter sind dabei mit abzugeben!**
8. Die Prüfung ist bestanden, wenn Sie 60 von 120 erreichbaren Punkten erzielen.

# Viel Erfolg!

---

| 1 | 2 | 3 | 4 | 5 | 6 | $\sum$ |
|---|---|---|---|---|---|---|
| | | | | | | |

**Aufgabe 1. (20 Punkte)** Betrachten Sie die Funktionen

```
01: unsigned int g(unsigned int x, unsigned int y, unsigned int z)
02: {
03:   if (z-y < 2) return y;
04:   unsigned int a = (y+z)/2;
05:   unsigned int b = a*a;
06:   if (b*b*a > x) return g(x,y,a);
07:   return g(x,a,z);
08: }
09:
10: unsigned int f(unsigned int x) { return g(x,0,x+1); }
```

und geben Sie die Nachbedingungen der Funktionen `f` und `g` an! Die Nachbedingung muss den Rückgabewert der Funktion vollständig charakterisieren, in Abhängigkeit von den Werten der Funktionsparameter beim Aufruf.
Für die Funktion `g` können Sie auch eine Vorbedingung definieren, welche die beim Aufruf zulässigen Werte der Parameter geeignet einschränkt. Die beim Aufruf von `g` in `f` auftretenden Parameter müssen diese Vorbedingung erfüllen.

**Aufgabe 2. (20 Punkte)** Jede natürliche Zahl $n \geq 1$ lässt sich bekanntlich in eindeutiger Weise als Produkt von Primzahlen schreiben; $n$ heisst *quadratfrei*, wenn in diesem Produkt keine Primahl mehr als einmal vorkommt. Zum Beispiel ist $n = 6 = 2 \cdot 3$ quadratfrei, während $n = 12 = 2 \cdot 2 \cdot 3$ nicht quadratfrei ist.
Implementieren Sie eine Funktion

```
// PRE:  n > 0
// POST: returns true if and only if n is square-free
bool square_free (unsigned int n);
```

die `true` genau dann zurückgibt wenn `n` quadratfrei ist.

**Aufgabe 3. (26 Punkte)** Implementieren Sie eine Klasse `perm` zur Repräsentation von Permutationen der Menge $\{0, 1, 2\}$ (eine Permutation ist eine bijektive Abbildung der Menge auf sich selbst). Eine jede solche Permutation $\pi$ kann durch ein Tripel $(a_0, a_1, a_2)$ beschrieben werden, wobei $\pi(i) = a_i$, $i \in \{0, 1, 2\}$. Zum Beispiel beschreibt das Tripel $(0, 1, 2)$ die Identität.
Beschreiben Sie eine geeignete Repräsentation (Daten-Mitglieder) für `perm` und implementieren Sie folgende zwei Funktionalitäten als Mitglieds-Funktionen.

- Berechnung der Werte $\pi(i)$ der repräsentierten Permutation $\pi, i \in \{0, 1, 2\}$.
- Die Verkettung der repräsentierten Permutation $\pi$ mit einer Permutation $\phi$.

Die Verkettung $\circ$ zweier Permutationen ist als Hintereinanderausführung der beiden Abbildungen definiert. Das heisst, $\pi \circ \phi$ ist die Permutation, die $i$ auf $\pi(\phi(i))$ abbildet, $i \in \{0, 1, 2\}$. Zum Beispiel ist

$$(1, 0, 2) \circ (1, 2, 0) = (0, 2, 1) \neq (2, 1, 0) = (1, 2, 0) \circ (1, 0, 2).$$

Erstellen Sie sowohl die Klassendefinition als auch die Implementation (Daten-Mitglieder und Mitglieds-funktionen)! Konstruktoren müssen für dieses Klassenfragment nicht realisiert werden. Vergessen Sie dabei nicht, Vor– und Nachbedingungen für die Mitglieds-Funktion anzugeben! Verwenden Sie `const` wo immer möglich und sinnvoll!

**Aufgabe 4. (18 Punkte)** Geben Sie für jeden der folgenden Ausdrücke die logische Klammerung an! Werten Sie den jeweiligen Ausdruck in Einzelschritten aus! Das heisst, geben Sie **alle Zwischenschritte** der Auswertung an und für jeden Schritt **sowohl den Wert als auch den Typ** des jeweils berechneten Zwischenresultats!

(a) `1+1*6/4==3&&2.1/1.3<1.8`

(b) `12/6/2*3.0+1<4||2.0*2>1`

**Aufgabe 5. (20 Punkte)**

a) Definieren Sie die Begriffe "Programm" und "Algorithmus" und erläutern Sie die Unterschiede zwischen beiden! (6 Punkte)

b) Was bedeutet es, dass ein Programm eine Menge $M \subset \mathbb{N}$ akzeptiert? (2 Punkte)

c) Beschreiben Sie die Definition der Menge DIAG. (6 Punkte)

d) Beweisen Sie, dass es kein Programm gibt, das DIAG akzeptiert. (6 Punkte)

**Aufgabe 6. (16 Punkte)** Eine linere Gleichung hat die Form $ax = b$ mit $a, b \in \mathbb{R}$. Implementieren Sie folgende Funktion zur Lösung solcher linearen Gleichungen. (Bemerkung: Ignorieren Sie für diese Aufgabe mögliche Rundungsfehler.)

```
// POST: return value is true if and only if the equation ax=b has a
//       solution x. If the return value is true, s is a solution, i.e.
//       s satisfies as=b
bool solve_linear_equation (double a, double b, double& s);
```