Prüfung — Informatik D-MATH/D-PHYS
24. 01. 2011 **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 allgemeinen Bemerkungen gelesen und verstanden habe.

Unterschrift: .....

Allgemeine Bemerkungen und Hinweise:

1. Überprüfen Sie die Vollständigkeit der ausgeteilten Prüfungsunterlagen (drei Blätter, bestehend aus 1 Deckseite, 4 Aufgabenseiten mit insgesamt 6 Aufgaben und eine leere Seite)!
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. Einzige Ausnahme sind Wörterbücher**.
4. Betrugsversuche führen zu sofortigem Ausschluss und können rechtliche Folgen haben.
5. Pro Aufgabe ist höchstens eine gültige Version eines Lösungsversuches 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 oder wenn Sie frühzeitig abgeben möchten, übergeben Sie Ihre Lösungen bitte einer Aufsichtsperson und verlassen zügig den Raum. **Jedes Ihrer Lösungsblätter muss mit Ihrem Namen beschriftet sein. Die Prüfungsblätter sind mit abzugeben!**
8. Die Prüfung ist bestanden, wenn Sie 60 von 120 erreichbaren Punkten erzielen.

1	2	3	4	5	6	$\sum$

Aufgabe 1. (21 Punkte) Geben Sie zu den folgenden Ausdrücken jeweils den Typ und den Wert an. Es gibt keine Punkte für die Teilschritte der Auswertung. Sie brauchen diese nicht hinzuschreiben.

Zu Beginn jeder der Teilaufgaben gilt: Die Variable `x` ist vom Typ `int` und hat den Wert 2. Die Variable `b` ist vom Typ `bool` und hat den Wert `true`.

a) `1.5 / 2 + 3 * 3`

b) `x - 1 < 2 || x + 3 == 4 || 3 / x == 1.5`

c) `!b && 3.7 / 2 * 3 / 5 == 1.11`

d) `33 * 5 % 7 * 14 / 2 % 7`

e) `2.0f + x++`

f) `3 >= x * 2 || 20 % 7 == 6`

g) `- x-- - 1 / 2.0`

Aufgabe 2. (28 Punkte) Diese Aufgabe besteht aus sieben unabhängigen und selbsterklärenden Teilaufgaben, von denen jede vier Punkte gibt.

a) Wie oft wird der Rumpf der folgenden `while`-Schleife ausgeführt?

```
int n = 16;
while ((n /= 2) > 0) {
  ++n;
}
```

b) Was ist die Ausgabe des folgenden Codes?

```
for (int i = 0; i < 16; i += 3) {
  if (i % 2 != 0) std::cout << i;
}
```

c) Welche Typen müssen die Variablen `a`, `b` und `c` haben, so dass der folgende Ausruck fehlerfrei kompiliert und **keine** impliziten Typenkonversionen erzeugt.

```
a % 10 == 1 && b && c - 1.2 > 0.0
```

Tipp: Die Antwort ist eindeutig.

d) Gegeben seien die folgenden beiden Funktionsdefinitionen.

```
int f(int a)
{
  if (a <= 1) return 2;
  return g(1,f(a-1));
}

int g(int a, int b)
{
  return f(b-a);
}
```

Terminiert der Funktionsaufruf `f(3)`? Was ist der Rückgabewert?

e) Bei welchem der folgenden Ausdrücke kommt **keine** Kurzschlussauswertung zum Zug?

1. `true || false && true`
2. `false || false && true`
3. `false && true || true`
4. `true && false || true`

f) Seien `a`, `b` und `c` drei `bool` Variablen. Welcher der folgenden vier Ausdrücke hat **nicht** immer den gleichen Wert wie die anderen drei?

1. `!(a && b) && !c`
2. `!(!(a && b) || c)`
3. `!(!(!a || !b) || c)`
4. `(!a || !b) && !c`

g) Was ist die Ausgabe des folgenden Codes? Nehmen Sie den IEEE Standard für Fliesskommazahlen an.

```
double d = 0.4;
double e = 0.125;
if (1000 * e == 125) {
  std::cout << "A";
} else {
  std::cout << "B";
}
if (10 * d == 4) {
  std::cout << "C";
} else {
  std::cout << "D";
}
```

Aufgabe 3. (10 / 6 Punkte) Die Funktion `foo` operiert auf einem `int`-Feld, das durch das Argument `array` gegeben ist. Die Länge des Feldes ist `n`. Die Vorbedingung ist, dass `n` eine gerade Zahl ist. Es gibt noch zwei weitere Argumente `d1` und `d2`.

Versuchen Sie zu verstehen, was die Funktion macht und beantworten Sie die folgenden Fragen.

```
// PRE: n ist gerade
01: void foo(const int[] array, int n, int& d1, int& d2) {
02:   int index = 0;
03:   if (n > 1) {
04:     d1 = array[index++];
05:     d2 = array[index++];
06:     if (d1 > d2) {
07:       int temp = d1;
08:       d1 = d2;
09:       d2 = temp;
10:     }
11:     int b, c;
12:     while (index < n) {
13:       b = array[index++];
14:       c = array[index++];
15:       if (b < c) {
16:         if (b < d1) d1 = b;
17:         if (c > d2) d2 = c;
18:       } else {
19:         if (c < d1) d1 = c;
20:         if (b > d2) d2 = b;
21:       }
22:     }
23:   }
24:   return;
25: }
```

a) Formulieren Sie eine Nachbedinung, die genau beschreibt, was die Funktion macht. Versuchen sie dabei nicht zu beschreiben, was einzelne Anweisungen machen. Dafür gibt es keine Punkte. Wichtig ist, was mit den Argumenten geschieht.

b) Geben sie in Abhängigkeit von `n` an, wie viele Vergleiche mit Elementen des Feldes gemacht werden. Zählen Sie dabei die Anzahl Ausführungen der Zeilen 06, 15, 16, 17, 19 und 20. Die Vergleiche in Zeile 03 und 12 werden dabei **nicht** gezählt.

Aufgabe 4. (20 Punkte) Gegeben sei das normalisierte Fliesskommazahlensystem $\mathcal{F}^*(2, 53, -1022, 1023)$, das der double precision nach IEEE entspricht. Das heisst in diesem System wird eine Zahl dargestellt als

$$\pm d_0.d_1 \dots d_{52} \cdot 2^e,$$

wobei $d_i \in \{0, 1\}$, für $1 \leq i \leq 52$, und d_0 immer gleich 1 ist. Ausserdem gilt $-1022 \leq e \leq 1023$.

Ein Paar von Zahlen $a, b \in \mathcal{F}^*(2, 53, -1022, 1023)$ heisst *gut*, wenn auch ihre Summe exakt dargestellt werden kann, also $a + b \in \mathcal{F}^*(2, 53, -1022, 1023)$ gilt. Falls dies nicht gilt, heisst das Zahlenpaar *schlecht*.

Die Frage, die Sie beantworten sollen, ist, ob es im System $\mathcal{F}^*(2, 53, -1022, 1023)$ mehr gute oder schlechte Zahlenpaare gibt? Begründen Sie Ihre Antwort mit konkreter Bezugnahme auf die Argumente des Systems. Es ist jedoch nicht nötig, exakte Angaben über die Zahl der schlechten und guten Paare zu machen.

Tipp: Überlegen sie sich für ein fixes a, welche Bedingung die Zahl b erfüllen muss, damit das Paar (a, b) gut ist.

Aufgabe 5. (15 Punkte) Ist die Menge der rationalen Zahlen $\mathbb{Q}$ abzälbar? Begründen Sie Ihre Antwort!

Eine Menge heisst abzählbar, wenn es eine Bijektion zwischen ihr und den natürlichen Zahlen $\mathbb{N}$ gibt. Beweisen oder widerlegen sie dies. Die Argumentation muss nicht formal, aber detailliert sein.

Aufgabe 6. (20 Punkte) Implementieren Sie eine Funktion, die die Reihenfolge der Elemente in einem Feld umdreht. Das Feld ist durch zwei Zeiger `begin` und `end` gegeben.

Wenn das Feld zu Beginn zum Beispiel die Werte {`3,2,4,5,1`} enthält, so soll es nach dem Aufruf der Funktion `reverse` die Werte {`1,5,4,2,3`} enthalten.

```
// PRE:  begin und end sind Zeiger auf ein Feld.
//       begin zeigt auf das erste Element und end zeigt auf das
//       Element nach dem letzten.
// POST: Die Reihenfolge der Elemente im Feld [begin, end) ist
//       umgedreht worden.
void reverse(int* begin, int* end);
```