Prüfung — Informatik D-MATH/D-PHYS
4. 8. 2009 **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 (1 einseitig bedrucktes Deckblatt und 2 zweiseitig bedruckte Aufgabenblätter 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: **Wörterbücher sind erlaubt; sonst keine Hilfsmittel**.
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 (ausser es wird explizit verlangt)!
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.

1	2	3	4	5	6	$\sum$

Aufgabe 1. (5 / 10 Punkte)

a) Welche der folgenden Aussagen treffen zu (Ja) respektive nicht zu (Nein)? Kreuzen Sie die entsprechenden Antworten direkt auf dem Aufgabenblatt an. Sie müssen keine Begründung angeben.

		Ja	Nein
(i)	Es gibt ein Registermaschinen-Programm `HALT`, welches ein beliebiges Registermaschinen-Programm `A` und dessen Eingabe `E` zur Eingabe hat, und welches in endlicher Zeit entscheidet, ob `A` auf `E` terminiert.	☐	☐
(ii)	Eine Registermaschinen-Sprache kann auch komplexe Instruktionen wie zum Beispiel das Quadrieren von Zahlen unterstützen.	☐	☐
(iii)	Mit `float` Fliesskommazahlen (IEEE754) kann man Zahlen beliebig nahe an 0 darstellen.	☐	☐
(iv)	Um einen Algorithmus in `C++` implementieren zu können, muss er mit den Operationen der rechnereigenen Registermaschinen-Sprache formuliert sein.	☐	☐
(v)	Jedes Programm setzt einen bestimmten Algorithmus um.	☐	☐

b) Betrachten Sie das folgende Registermaschinen-Programm und charakterisieren Sie genau, was das Verhalten des Programmes sein wird. Dazu gehört sowohl, dass Sie Aussagen darüber treffen, welche Ausgabe in Abhängigkeit der Eingabe zu erwarten ist, als auch, dass Sie beschreiben, welche Probleme bei bestimmten Eingaben auftreten können. Beachten Sie, dass in diesem Registermaschinen-Modell beliebige reelle Zahlen als Eingabe erlaubt sind.

```
1. Read into Register(1)
2. Read into Register(2)
3. Register(3) ← 1
4. Register(4) ← 1
5. If Register(2) = 0, then go to line 9
6. Register(3) ← Register(3) * Register(1)
7. Register(2) ← Register(2) - Register(4)
8. Go to line 5
9. Output ← Register(3)
10. End
```

Aufgabe 2. (18 Punkte) Geben Sie zu den folgenden Ausdrücken jeweils den Typ und den Wert an. Spezifizieren Sie auch, ob der Ausdruck ein R-Wert oder ein L-Wert ist. Es gibt keine Punkte für die Teilschritte der Auswertung. Sie brauchen diese nicht hinzuschreiben.

Die Variable `x` ist vom Typ `int` und hat den Wert `1` (dies gilt zu Beginn jeder der Teilaufgaben). Bei der Konversion von `bool` zu `int` wird `true` zu `1` und `false` zu `0` umgewandelt. Umgekehrt wird `0` zu `false` und jeder andere `int`-Wert zu `true` umgewandelt.

a) `1 + true == 2`

b) `3 % 2 + 1 * 4`

c) `x = 10 / 2 / 5 / 2`

d) `x / 2.0`

e) `1 + x++`

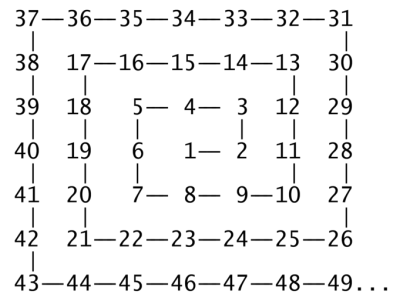
f) `++x`

Aufgabe 3. (22 Punkte) Schreiben Sie eine C++ Funktion, die den Binomialkoeffizienten $\binom{n}{k}$ anhand der folgenden Formel berechnet, wobei $n, k \in \mathbb{N} \cup \{0\}$.

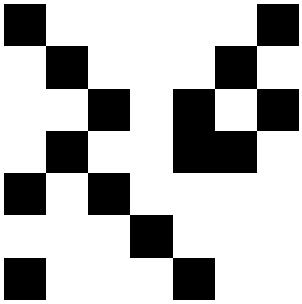
$$\binom{n}{k} = \begin{cases} 0 & \text{falls } n < k \\ 1 & \text{falls } n = k \text{ oder } k = 0 \\ \frac{n}{k}\binom{n-1}{k-1} & \text{sonst} \end{cases}$$

Überlegen Sie sich dazu welche Datentypen Sie verwenden wollen und ergänzen Sie Ihre Implementierung mit geeigneten Vor- und Nachbedingungen.

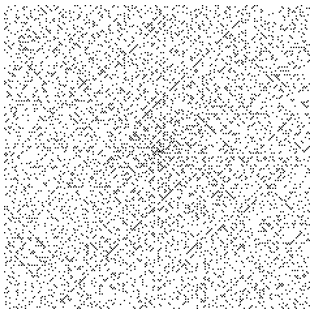
Aufgabe 4. (30 Punkte) Ihre Aufgabe ist es, die sogenannte Ulam-Spirale auf dem Bildschirm auszugeben. Das nach einem polnischen Mathematiker benannte Konstrukt trägt alle natürlichen Zahlen spiralförmig in ein Gitter ein, beginnend mit der 1 aufsteigend (siehe Abbildung 1(a)). Ersetzt man nun jede Primzahl durch einen schwarzen Gitterpunkt und lässt die anderen Zahlen weg, erhält man die Ulam Spirale, und es wird ersichtlich, dass sich die Primzahlen auf Diagonalen häufen (siehe Abbildung 1(c)).



(a) Die Grundidee ist eine spiralförmige Auflistung der natürlichen Zahlen.



(b) Das Gitter aus Abbildung 1(a), in dem die Zahlen durch Gitterpunkte ersetzt wurden.



(c) In dieser Abbildung ist ein grösseres Gitter zu sehen (ca. 200x200 Gitterpunkte)

Vervollständigen Sie folgendes Programmgerüst mit der Implementierung der Funktion `ulam_spirale` gemäss den Vor- und Nachbedingungen. Sie haben dazu zwei Funktionen zur Verfügung. Die Funktion `prim` entscheidet, ob eine Zahl prim ist. Die Funktion `print` färbt einen Gitterpunkt schwarz. Das Ausgabefenster können Sie als gegeben betrachten und es hat ausreichende Grösse. Zentrieren Sie die Spirale (d.h. den Gitterpunkt für die Zahl 1) am Ursprung (d.h. dem Punkt $(0,0)$). Der positive Quadrant des Koordinatensystems ist rechts oben.

```
// PRE: n > 0
// POST: Gibt true zurueck, wenn n eine Primzahl ist, false
//       andernfalls.
bool prim(unsigned int n);

// POST: Der Gitterpunkt (i,j) wird schwarz gefaerbt.
void print(int i, int j);

// PRE: n > 3
// POST: Es wird die Ulam Spirale als Gitterpunktbild auf dem Bildschirm
//       ausgegeben. Dabei werden die ersten n natuerlichen Zahlen
//       beruecksichtigt.
void ulam_spirale(unsigned int n) {
   // ERGAENZEN SIE IHRE IMPLEMENTIERUNG HIER.
}
```

Aufgabe 5. (14 / 6 Punkte)

a) Geben Sie für die folgende Funktion die Nachbedingung an!

```
// PRE: [b, e) und [o, o+(e-b)) sind zwei gueltige
//      und disjunkte Bereiche.
void f (int* b, int* e, int* o)
{
  while (b != e) *(o++) = *(--e);
}
```

Die Nachbedingung muss das Verhalten der Funktion für alle gültigen Eingaben *vollständig* beschreiben.

b) Geben Sie für die drei im folgenden Quelltext auftretenden Aufrufe der Funktion `f` an, ob es sich laut Vorbedingung um einen gültigen Aufruf handelt. Falls es sich bei einem Beispiel nicht um einen gültigen Aufruf handelt, begründen Sie Ihre Antwort.

```
int a[5] = {1,2,3,4,5};

f(a, a+5, a+5);
f(a, a+2, a+3);
f(a, a+3, a+2);
```

Aufgabe 6. (15 Punkte). In der Vorlesung haben wir gesehen, dass es Dezimalzahlen gibt (zum Beispiel 0.1), die keine endliche Darstellung als binäre Fliesskommazahl (Basis $\beta = 2$) haben. Kollege X. M. Plestudent behauptet, die Ursache sei, dass $\beta < 10$ gilt. Er schlägt vor, mit einem hexadezimalen Fliesskommazahlensystem ($\beta = 16$) zu arbeiten und meint, in einem solchen System habe auch 0.1 eine endliche Darstellung. Hat Herr Plestudent recht oder nicht?

Etwas formaler: gibt es eine natürliche Zahl p und Zahlen $d_1, \ldots, d_p$ mit $d_i \in \{0, \ldots, 15\}$ für alle i, so dass

$$\frac{1}{10} = \sum_{i=1}^{p} d_i 16^{-i} = \text{“}0.d_1 \ldots d_p\text{”}$$

gilt? Begründen Sie Ihre Antwort!