Prof. Dr. A. Weber
Dr. Hassan Errami

Universität Bonn
Institut für Informatik II
14.11.2016

Wintersemester 2016/2017

Übungen zur Vorlesung
Algorithmisches Denken und imperative Programmierung (BA-INF-014)
Aufgabenblatt 4

Zu bearbeiten bis: 19.11.2016

Aufgabe 1 *(Binomialkoeffizient - 1+1+1=3 Punkte)*

a) Implementieren Sie in C eine rekursive Funktion, die zu einem Eingabeparameter n den Wert $n! = n \cdot (n-1) \cdot (n-2) \cdot ... \cdot 1$ zurückgibt *(Fakultätsfunktion)*.

b) Die für nicht-negative ganze Zahlen n und k definierte Funktion

$$\left(\begin{array}{c} n \\ k \end{array} \right) := \begin{cases} \frac{n!}{k!(n-k)!} & \text{für } 0 \leq k \leq n \\ 0 & \text{für } 0 \leq n < k \end{cases}$$

(gesprochen „n über k“) heißt *Binomialkoeffizient*. Implementieren Sie die Binomialkoeffizienten-Funktion. Nutzen Sie Ihre Funktion aus Aufgabenteil a).

c) Kombinieren Sie nun die beiden Funktionen zur Lösung des *modifizierten Lottoproblems*: Aus n Zahlen lassen sich – bei Berücksichtigung der Anordnung – k Zahlen ohne Zurücklegen auf

$$\left(\begin{array}{c} n \\ k \end{array} \right) k!$$

Arten auswählen.

Aufgabe 2 *(Rekursive Funktionen - 2+2+2=6 Punkte)*

a) Betrachten Sie folgende Vorgabe:

```
#include <stdio.h>

// Gibt den Nachfolger des übergebenen Wertes zurück
int succ(int x) {
   return ++x;
}

// Gibt den Vorgänger des übergebenen Wertes zurück
int pre(int x) {
   return --x;
}

// ...
```

Implementieren Sie ausschließlich unter Nutzung von Vergleichsoperatoren und den oben gegebenen Funktionen die nachfolgenden Funktionen rekursiv, wobei $x \in \mathbb{Z}$ und $y \in \mathbb{N}_0$ (Sie müssen also **keine** negativen Werte für y beachten):

- *int add(int x, int y)*, welche die Summe von x und y zurückgibt
- *int sub(int x, int y)*, welche die Differenz von x und y zurückgibt
- *int mult(int x, int y)*, welche das Produkt von x und y zurückgibt. Hier können Sie Ihre zuvor geschriebenen Funktion *add* verwenden.

Testen Sie ihre Funktionen jeweils für folgende Wertepaare: (2,0) und (-3,2).

Aufgabe 3 *(Trigonometrische Funktionen und Exponentialfunktion - 2+2+2=6 Punkte)*

Schreiben Sie ein Programm, welches eine Gleitkommazahl x von der Konsole einliest. Approximieren Sie anschließend

- sin(x) nach der Formel $\sum_{n=0}^{5} (-1)^n \frac{x^{2n+1}}{(2n+1)!}$
- cos(x) nach der Formel $\sum_{n=0}^{5} (-1)^n \frac{x^{2n}}{(2n)!}$
- exp(x) nach der Formel $\sum_{n=0}^{10} \frac{x^n}{n!}$

Testen Sie ihr Programm mit $x \approx \frac{\pi}{4}$ und $x \approx \frac{\pi}{2}$ bei Sinus und Kosinus sowie mit 1 bei der Exponentialfunktion. **Tipp:** Nutzen Sie n als Schleifenvariable. Sie können für diese Aufgabe ihre Fakultätsfunktion aus Aufgabe 1 und die **Potenzfunktion *pow* aus math.h** nutzen.

Aufgabe 4 *(Pythagoras-Triplet - 2 Punkte)*

Ein Pythagoras-Triplet ist eine Menge von drei natürlichen Zahlen, für die gilt:

- $a < b < c$
- $a^2 + b^2 = c^2$

Schreiben Sie eine Funktion, welche nach und nach Pythagoras-Triplets findet und ausgibt. **Tipp:** Sie können ihre auf Aufgabenblatt 2 geschriebene Potenzfunktion oder die Funktion *pow* aus math.h nutzen.

Aufgabe 5 *(Münzen - 3 Punkte)*

Schreiben Sie ein Programm, welches folgendes leistet: Gegeben seien die Münzwerte 50, 20, 10, 5, 2 und 1. Ihr Programm gibt einen zufälligen ganzzahligen Betrag aus (vgl. Übungsblatt 3, Aufgabe 1). Anschließend liest ihr Programm von der Konsole einen ganzzahligen Wert ein, der höher als der zufällige Betrag sein muss. Das Programm berechnet die Differenz zwischen den Beträgen und zerlegt diese in die gegebenen Münzwerte, und zwar so, dass es jeweils möglichst viele hohe Münzwerte zurückgibt. Geben Sie die entsprechenden Münzwerte in der Konsole aus und testen Sie ihr Programm mit verschiedenen Beträgen.