Prof. Dr. A. Weber
Dr. Hassan Errami

Universität Bonn
Institut für Informatik II
07.11.2016

Wintersemester 2016/2017

Übungen zur Vorlesung
Algorithmisches Denken und imperative Programmierung (BA-INF-014)
Aufgabenblatt 3

Zu bearbeiten bis: 12.11.2016

Aufgabe 1 *(Blackjack - 2+4=6 Punkte)*

In der Casino-Welt gehört Blackjack zu den beliebtesten Tischspielen. In dieser Aufgabe soll zunächst eine vereinfachte Version des Spiels in C implementiert werden. Das (hier vereinfachte) Spiel funktioniert wie folgt: Es steht ein Kartendeck mit den Karten von 2 bis 10 bereit. Die Wertigkeit der Karten gleicht ihrer Zahl. Ein einzelner Spieler tritt direkt gegen das Haus an. Das Spiel lässt sich in Phasen gliedern:

- Anfangs zieht das Dealer eine Karte vom Stapel und deckt diese auf.
- Anschließend kann der Spieler so viele Karten ziehen, wie er möchte. Sein Ziel sollte es sein, mit der Summe seiner Kartenwerte möglichst nah an 21 Punkte zu kommen, ohne sie zu überschreiten. Überschreitet er in der Summe die 21, hat er sofort verloren.
  Liegt er darunter und möchte keine weitere Karte ziehen („stay“), beginnt die nächste Phase:
- Der Dealer deckt so lange Karten auf, bis er inklusive der zuerst gezogenen Karte in der Summe 16 Punkte überschreitet. Danach darf er keine weitere Karte mehr aufdecken. Hat er dabei die 21 Punkte-Grenze überschritten oder hat er weniger Punkte als der Spieler, verliert er. Ansonsten gewinnt das Haus.

Aufgaben:

- Legen Sie eine neue C-Codedatei an und nutzen Sie folgende Vorlage:

```
#include <stdio.h>
#include <time.h>
#include <stdlib.h>

void initializegenerator() {
  srand(time(NULL));
}
int karteziehen() {
  // Sie koennen Zufallszahlen generieren, indem Sie die Funktion rand() aufrufen
  // z.B. int zufall = rand();
}
int main() {
  initializegenerator();
  // Code fuer den Spielablauf

  return 0;
}
```

- Implementieren Sie die Funktion „int karteziehen“, die die Funktion rand verwendet, um eine Zufallszahl zwischen 2 und 10 zu generieren und diese zurückgibt. **Tipp:** Nutzen Sie den Modulo-Operator %, um die Zahlen in ihrer Größe zu begrenzen.
- Schreiben Sie anschließend ein Programm, welches die oben beschriebene Form von Blackjack implementiert:
  - Nutzen Sie ihre eben erstellte Funktion „karteziehen“, um eine zufällige Karte für den Dealer zu ziehen und geben Sie die Wertigkeit in der Konsole aus. Lesen Sie anschließend wiederholt von der Kommandozeile. Jedes mal, wenn der Benutzer eine 1 eingibt, generieren Sie wie oben eine Zufallszahl zwischen 2 und 10 und addieren Sie sie auf. Wenn die Summe über 21 liegt, beenden Sie das Programm mit der Ausgabe „Spieler verliert“. Sobald der Benutzer eine 0 eingibt („stay“), fahren sie wie folgt fort:
  - Rufen Sie für den Dealer so oft „karteziehen“ auf, bis die Summe aus der ersten Karte des Dealers und den neu generierten Zahlen über 16 liegt. Liegt die Summe sogar über 21, beenden Sie das Programm mit der Ausgabe „Spieler gewinnt“. Andernfalls vergleichen Sie die Punkte von Spieler und Dealer und geben Sie den Gewinner aus.

Aufgabe 2 *(Statistiken auf Arrays - 5 Punkte)*

In dieser Aufgabe sollen Sie das arithmetische Mittel und die korrigierte Stichprobenvarianz eines Arrays von Zahlen bestimmen. Ist eine Stichprobe von n Werten $x_1, \ldots, x_n$ gegeben, so ist das arithmetische Mittel $\bar{x}$ definiert als:

$$\bar{x} := \frac{1}{n} \sum_{i=1}^{n} x_i$$

Die korrigierte Stichprobenvarianz s^2 ist definiert als:

$$s^2 := \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2$$

Schreiben Sie ein Programm, das:

1. Den Benutzer zunächst nach der Größe n (Integer) der Stichprobe fragt,
2. Die n Werte (double) einliest und in ein Array speichert.
3. Dann $\bar{x}$ sowie s^2 berechnet und ausgibt.

Aufgabe 3 *(Zahlen ausgeben - 5 Punkte)*

Schreiben Sie ein C-Programm, welches eine natürliche Zahl zwischen 20 und 69 einliest und anschließend die Zahl in Lautschrift ausgibt. Eine Ausgabe des Programms soll zum Beispiel wie folgt aussehen:

Eingabe der Zahl: 51
Ausgabe: ein-und-fünfzig

Eingabe der Zahl: 30
Ausgabe: dreißig

Beachten Sie dabei, wie Zahlen wörtlich zusammengesetzt sind. Programmieren Sie also **nicht** jede einzelne Zahl als String mit ein, sondern setzen Sie diesen zur Laufzeit (sofern möglich) zusammen, indem Sie die Ausgabefunktion „printf“ ggf. mehrfach hintereinander verwenden. Liegt die eingebene Zahl außerhalb des oben angegeben Bereichs, geben Sie eine Fehlermeldung aus.

Aufgabe 4 *(Gleitkomma-Arithmetik - 4 Punkte)*

Es seien $x_1 =$ `10000.0`, $x_2 =$ `-1.0e-3 / 9.0`, $x_3 =$ `25.0e2`, $x_4 =$ `1.0e-3 / 7.0` und $x_5 =$ `-12.5e3`. Im folgenden soll die Summe $\sum_{i=1}^{5} x_i$ auf verschiedene Art und Weise und mit verschiedenen Datentypen berechnet werden.

a) Berechnen Sie (von Hand) die (exakte) Summe $\sum_{i=1}^{5} x_i$.

b) Welches Ergebnis berechnet Ihr Programm, falls Sie ausschließlich Variablen vom Typ `float` verwenden?

c) Welches Ergebnis berechnet Ihr Programm, falls Sie ausschließlich Variablen vom Typ `double` verwenden?

d) Zur Berechnung einer Summe $S = \sum_{i=0}^{n} x_i$ wird folgendes Verfahren vorgeschlagen:

   (a) $S = 0$, $D = 0$
   (b) für $i = 1, \ldots, n$:
      i. $S_{\text{alt}} = S$
      ii. $S = S + x_i$
      iii. $D = D + (x_i - (S - S_{\text{alt}}))$
   (c) $S = S + D$

   Welches Ergebnis S berechnet C mit diesem Verfahren für $n = 5$ und obigen Werten für die x_i? Dabei sollen wieder nur Variablen des Typs `float` verwendet werden.

e) Weshalb ist diese Rechenvorschrift der einfachen Summation überlegen?

f) Was liefert das Verfahren aus (d), wenn Sie nur Variablen des Types `double` verwenden?