FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Einführung in die Programmierung mit C – WS24/25

9. Übung

Biemann Nikbakhsh Gibietz

13. Januar 2025

Aufgabe 1

Erinnern Sie sich noch an die Münzgeldrückgabe?[1] Versuchen Sie nun den Quelltext noch kompakter zu gestalten, indem Sie die verfügbaren Münzen in einem Array speichern.

Gestalten Sie nun weiterhin die Textausgabe nutzerfreundlicher:

- Die Anzahlen 1–5 sollen als Text ausgegeben werden (Eins, Zwei, Drei, …), ansonsten als Zahl (6, 7, 8,…).
- Unnötige Münzen (mit Anzahl 0) werden nicht erwähnt.
- Es erfolgt eine Unterscheidung zwischen Singular („Eine Münze“) und Plural („Zwei Münzen“).

Ein möglicher Programmablauf sei wie folgt:

```
Wie gross ist der Euro-Betrag, welcher in moeglichst wenig
   Muenzen aufgeteilt werden soll? 8.97
Vier 2-Euro-Muenzen
Eine 50-Cent-Muenze
Zwei 20-Cent-Muenzen
Eine 5-Cent-Muenze
Eine 2-Cent-Muenze
```

[1] Wenn nicht, schauen Sie auf den älteren Aufgabenblättern nach!

Aufgabe 2

Sie haben sicherlich schon bemerkt, dass scanf() einige Nachteile hat. Nicht nur beim Einlesen von Zeichenketten sondern auch beim Einlesen von Zahlen gibt es einige nervige Probleme, die wir nun loswerden wollen. Ab jetzt möchten wir beim Einlesen von ganzen Zahlen auch kein scanf() mehr verwenden.

Für Zeichenketten haben wir uns selbst die Funktion readtext() programmiert. Es fehlt nun aber noch etwas für ganze Dezimalzahlen. Programmieren Sie daher analog zur Funktion readtext() eine Funktion readint(), welche eine Dezimalzahl von der Tastatur einliest und als Integerwert zurückliefert.

```
1  int zahl;
2  printf("Bitte geben Sie eine Zahl ein: ");
3  zahl = readint();
4  printf("Die Zahl lautet %d.\n", zahl);
```

Folgendes soll gelten:

- Ihre Funktion soll unabhängig von scanf() sein.
- Ihre Funktion soll kein Enterzeichen (ASCII-Code 10) im Tastaturpuffer hinterlassen (wie scanf es manchmal tut).
- readint() soll sowohl positive als auch negative ganze Zahlen einlesen können.
- Positive Zahlen haben entweder kein Vorzeichen oder das Pluszeichen (ASCII-Code 43), negative Zahlen das Minuszeichen (ASCII-Code 45) vorangestellt.
- Alle Zeichen außer Ziffern, Plus- und Minuszeichen führen dazu, dass Ihre Funktion die Benutzereingabe nicht weiter auswertet. Die Eingabe `123abc456` liefert die Zahl 123 zurück und *nicht* die Zahl 123456.
- Gibt der Benutzer zu Beginn Unsinn oder nichts ein, so wird die Null als Ergebnis zurückgeliefert.
- Lagern Sie Ihre Funktion wie bei readtext() in einer eigenständigen Datei aus, so dass diese per include-Anweisung einfach in beliebige andere Programme eingebunden werden kann.

Folgende Benutzereingaben sollen wie folgt interpretiert werden:

- `42` sei 42.
- `+42` sei 42.
- `-42` sei -42.
- `42a` sei 42.
- `-42!vf32g` sei -42.

- `h42` sei 0.
- `d` sei 0.
- `4+2` sei 4.
- `-4Fd#2` sei -4.

Schreiben Sie anschließend ein Hauptprogramm, welches readint() beispielhaft demonstriert.

Optional: Programmieren Sie auch eine Funktion readdouble.c, welche Fließkommazahlen einlesen kann.

Aufgabe 3

Erstellen Sie eine Struktur zur Speicherung eines dreidimensionalen Vektors (also mit einem Zahlentripel) für die drei Raumkoordinaten x, y und z.

a) Programmieren Sie eine Funktion zur Ausgabe eines Vektors. Als Parameter wird ein Vektor übergeben. Die Ausgabe des Zahlentripels erfolgt zwischen zwei runden Klammern getrennt von Kommas. Bsp: `(1,-2,3)`

b) Programmieren Sie eine Funktion zur Addition zweier Vektoren.

$$\vec{a} + \vec{b} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} + \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix} = \begin{pmatrix} a_x + b_x \\ a_y + b_y \\ a_z + b_z \end{pmatrix}$$

c) Programmieren Sie eine Funktion in welcher der Benutzer aufgefordert wird drei Raumkoordinaten einzugeben.[2] Der Rückgabewert Ihrer Funktion soll ein Vektor sein.

Aufgabe 4

Sie sollen ein Programm erstellen, welches einzelne Personen verwalten kann. Zum Beispiel zur Erstellung eines Adressbuchs. Jede Person zeichnet sich durch folgende Eigenschaften aus:

- Vorname in Form einer beliebig langen Zeichenkette
- Nachname in Form einer belibig langen Zeichenkette
- Alter in Form einer ganzen Zahl
- Geschlecht (Datentyp bleibt Ihnen überlassen)

Programmieren Sie folgendes:

1. Erstellen Sie zuerst eine Struktur zur Speicherung einer Person.

[2] Falls möglich bereits mit der Funktion readint().

2. Erstellen Sie eine Funktion zum Einlesen einer Person.

3. Erstellen Sie eine Funktion zur Ausgabe einer Person. Dabei soll das Format der Ausgabe wie folgt sein:

   a) Je nach Geschlecht wird eine passende Anrede vorangestellt (z. B. Frau oder Herr).

   b) Dann wird der Vorname ausgegeben.

   c) Im Anschluss wird der Nachname ausgegeben.

   d) Zum Schluss wird in runden Klammern das Alter hinten angestellt.

   Beispiel 1: `Frau Martina Muster (22)`
   Beispiel 2: `Herr Martin Mustermann (24)`

4. Erstellen Sie eine Funktion zum Initialisieren einer Person. Programmieren Sie es so, dass Personen, welche noch keinen Namen haben, identifiziert werden können. Wie lässt sich dies am einfachsten realisieren?

5. Ändern Sie die Ausgabe einer Person dahingehend ab, dass Personen mit fehlendem Vor-, Nachnamen oder Geschlecht korrekt ausgegeben werden. Hierbei soll eine kleine Information bei der Bildschirmausgabe erscheinen, ob und welcher Name oder das Geschlecht fehlen; beispielsweise direkt nach dem Initialisieren einer Person.

6. Erstellen Sie eine Funktion zum Löschen von Personen, so dass reservierter Speicherplatz vor Programmende freigegeben werden kann. Funktioniert die Ausgabe einer Person nach dem Löschen immer noch korrekt oder führt es bei Ihnen zu einem Absturz?

7. Erstellen Sie eine Funktion zum Vergleich zweier Personen, welche älter oder jünger ist.

8. Erstellen Sie ein Hauptprogramm, welches Ihre Funktionen in sinnvoller Weise verwendet.