Human-Computer Interaction Group

Leibniz Universität Hannover

Übungsblatt 7

Programmieren 1 – WiSe 23/24

Prof. Dr. Michael Rohs, Jan Feuchter, M.Sc

Alle Übungen (bis auf die erste) müssen in Zweiergruppen bearbeitet werden. Beide Gruppenmitglieder müssen die Lösung der Zweiergruppe einzeln abgeben. Die Namen beider Gruppenmitglieder müssen sowohl in der PDF Abgabe, als auch als Kommentar in jeglichen Quelltextabgaben genannt werden. Plagiate führen zum Ausschluss von der Veranstaltung.

Abgabe bis Donnerstag den 30.11. um 23:59 Uhr über https://assignments.hci.uni-hannover.de/WiSe2023/Prog1. Die Abgabe muss aus einer einzelnen Zip-Datei bestehen, die den Quellcode, ein PDF für Freitextaufgaben und alle weiteren nötigen Dateien (z.B. Eingabedaten oder Makefiles) enthält. Lösen Sie Umlaute in Dateinamen auf.

Zum Bestehen der Studienleistung müssen Sie mindestens eine Aufgabe erfolgreich lösen. Sie dürfen natürlich gerne (zum Beispiel zum Erreichen des Klausurbonus) alle bearbeiten und abgeben.

Aufgabe 1: Gewichte umrechnen

In dieser Aufgabe werden Gewichtsangaben zwischen verschiedenen Einheiten umgerechnet. Die Struktur `Weight` repräsentiert ein Gewicht in einer bestimmten Einheit. Die möglichen Einheiten (Gramm g, Kilogramm kg, Tonne t, Pound lb) sind über die Aufzählung Unit definiert. Es gilt $1\,\text{lb} = 0.453\,592\,37\,\text{kg}$. Auch ist eine Testfunktion **test_within_weight** für Gewichte gegeben. Verwenden Sie die Template-Datei **weights.c** und bearbeiten Sie die mit **todo** markierten Stellen.

a) Implementieren Sie die Konstruktorfunktion **make_weight**.

b) Implementieren Sie die Funktion **print_weight**, die Gewichte in folgendem Format ausgibt:
```
1234.00 g
4.75 kg
3.10 t
5.40 lbs
```
Runden Sie auf maximal zwei Nachkommastellen.

c) Implementieren Sie die Funktion **to_unit**. Diese soll ein Gewicht in eine Zieleinheit konvertieren. Fügen Sie zunächst mindestens 5 sinnvolle Tests zu **to_unit_test** hinzu.

d) Implementieren Sie die Funktion **compare**. Diese soll zwei Gewichte vergleichen und `0` zurückgeben, wenn `w` und `v` das gleiche Gewicht repräsentieren, `-1` zurückgeben, wenn `w` ein kleineres Gewicht als `v` repräsentiert und `1` zurückgeben, wenn `w` ein größeres Gewicht als `v` repräsentiert. Fügen Sie zunächst mindestens 5 sinnvolle Tests zu **compare_test** hinzu.

Aufgabe 2: Volumen geometrischer Körper

In einem Programm sollen verschiedene Formen von geometrischen Körpern – Zylinder, Kugeln und Quader – repräsentiert werden. Entwickeln Sie Funktionen, die diese geometrischen Körper verarbeiten und das zugehörige Volumen berechnen. Orientieren Sie sich an den im Skript unter Recipe for Variant Data (Recipe for Variant Data) beschriebenen Schritten. Verwenden Sie die Template-Datei `volumes.c` und bearbeiten Sie die mit `todo` markierten Stellen.

a) Definieren Sie die Struktur `Body` zur Repräsentation eines geometrischen Körpers. Die möglichen Varianten sind Cylinder, Sphere und Cuboid. Nutzen Sie die entsprechenden Strukturen.

b) Wofür ist die Enumeration `TypeTag` wichtig? Wo und wie wird sie genutzt? Fügen Sie die Antwort als Kommentar in Ihren Quelltext ein.

c) Implementieren Sie die Konstruktorfunktionen `make_cylinder`, `make_sphere` und `make_cuboid`, die einen `Body` vom gegebenen Typen erzeugen und die jeweiligen Parameter setzen. Stellen Sie durch Preconditions sicher, dass keine negativen Werte übergeben werden können.

d) Implementieren Sie die Funktion `volume`, die das Volumen eines `Body` ausrechnet. Abhängig vom Typen sollen die jeweiligen Parameter und die jeweilige Volumenformel genutzt werden.

Hinweise

- Der Wert von π ist über die Konstante `M_PI` definiert.

Aufgabe 3: Verschachtelte Schleifen

Implementieren Sie Funktionen, die Ausgaben der angegebenen Form erzeugen. Der Parameter **n** beschreibt die Größe der Ausgabe. Erlaubte Werte für den Parameter **n** liegen im Intervall **[0, 9]**. In den Beispielen gilt **n = 5**. Verwenden Sie in dieser Aufgabe verschachtelte for-Schleifen. Die Template-Datei für diese Aufgabe ist **loops.c**.

a) Implementieren Sie die Funktion **void loops_a(int n)**, die Ausgaben der folgenden Form erzeugt:

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

b) Implementieren Sie die Funktion **void loops_b(int n)**, die Ausgaben der folgenden Form erzeugt:

```
        1
      1 2
    1 2 3
  1 2 3 4
1 2 3 4 5
```

c) Implementieren Sie die Funktion **void loops_c(int n)**, die Ausgaben der folgenden Form erzeugt:

```
    1
   1 2
  1 2 3
 1 2 3 4
1 2 3 4 5
```

d) Implementieren Sie die Funktion **void loops_d(int n)**, die Ausgaben der folgenden Form erzeugt:

```
0         0
  1     1
    2
  1     1
0         0
```

e) Implementieren Sie die Funktion **void loops_e(int n)**, die Ausgaben der folgenden Form erzeugt (für **n ≥ 3**). Sie dürfen beliebige Hilfsfunktionen implementieren.

```
    +--------+
   /        /
  /        /
 /        /
+--------+
```

f) **(OPTIONAL)** Implementieren Sie die Funktion **void loops_f(int n)**, die Ausgaben der folgenden Form erzeugt (für **n ≥ 3**). Sie dürfen beliebige Hilfsfunktionen implementieren.

```
n = 5:
    +--------+
   /0 1 2 3 /
  /4 5 6 7 /
 /8 9 0 1 /
+--------+

n = 9:
        +----------------+
       /0 1 2 3 4 5 6 7 /
      /8 9 0 1 2 3 4 5 /
     /6 7 8 9 0 1 2 3 /
    /4 5 6 7 8 9 0 1 /
   /2 3 4 5 6 7 8 9 /
  /0 1 2 3 4 5 6 7 /
 /8 9 0 1 2 3 4 5 /
+----------------+
```

Aufgabe 4: Dateien einlesen und verarbeiten

In dieser Aufgabe soll die Tabelle **people.txt** eingelesen und verarbeitet werden. Die Tabelle enthält eine Zeile mit Spaltenbeschreibungen und dann eine größere Anzahl Zeilen mit entsprechenden Werten. Die erste Spalte enthält Geburtsjahre (Typ int), die zweite die Geschlechter ('m', 'f', 'd') (Typ char), und die letzte Körpergrößen in Meter (Typ double).

Im Template ist bereits Code enthalten, der die Datei in einen String einliest. Gehen Sie diesen zeilenweise durch und errechnen Sie damit folgende statistische Angaben:

- Das durchschnittliche Geburtsjahr (gerundet auf ganze Jahre)
- Die Anzahl Personen pro Geschlecht
- Die durchschnittliche Körpergröße pro Geschlecht

Vervollständigen Sie die Funktion **compute_statistics**, welche die in einer Zeichenkette (Typ: String) vorliegende Tabelle verarbeitet und dafür eine Statistik berechnet und zurückgibt. Schauen Sie sich zunächst an, wie der Inhalt von **people.txt** formatiert ist. Ihre Funktion muss nur für dieses Format funktionieren. Eine Fehlerbehandlung soll nicht implementiert werden.

Hinweise:

- Benutzen Sie zur Verarbeitung des Strings die Funktionen `s_length`, `s_get` und `s_sub`.
- In der **people.txt** sind Zeilen durch ein `newline` ('\n') getrennt, einzelne Daten durch einen Tabulator ('\t').
- Benutzen Sie zur Konvertierung eines Strings in ein `int` die Funktion `i_of_s` und zur Konvertierung eines Strings in ein `double` die Funktion `d_of_s`.