

Programmieren 1 – WS 2020/21

Prof. Dr. Michael Rohs, Tim Dünke, M.Sc.

Übungsblatt 7

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. Wenn Sie für Übungsblatt 1 noch keinen Gruppenpartner haben, geben Sie alleine ab und nutzen Sie das erste Tutorium dazu, mit Hilfe des Tutors einen Partner zu finden. Plagiate führen zum Ausschluss von der Veranstaltung.

Abgabe bis Donnerstag den 03.12. um 23:59 Uhr über <https://assignments.hci.uni-hannover.de/WiSe2020/Programmieren1>. Die Abgabe muss aus einer einzelnen zip-Datei bestehen, die den Quellcode, ein pdf bei Freitextaufgaben und alle weiteren nötigen Dateien (z.B. Eingabedaten oder Makefiles) enthält. Lösen Sie Umlaute in Dateinamen bitte auf.

Die Dokumentation der Prog1lib finden Sie unter: <https://hci.uni-hannover.de/files/prog1lib/index.html>

Aufgabe 1: Rechtecke

Das Template für diese Aufgabe ist die Datei `rects.c`. Diese Aufgabe ähnelt der Aufgabe aus der 5. Präsenzübung. In dieser Aufgabe sollen Sie Rechtecke auf der Konsole ausgeben. Diese werden als Struktur dargestellt und haben ein `x` und `y` Koordinate sowie eine Breite `width` und Höhe `height` und eine Fülleigenschaft. `x` beschreibt dabei die Anzahl an Leerzeichen vor dem Rechteck und `y` die Anzahl an leeren Zeilen über dem Rechteck. Es gibt 3 verschiedene Fülleigenschaften. Eine Füllung kann eine `SimpleFill` sein, dann besteht diese immer aus dem gleichen Zeichen:

```
+--+ // x = 3, y = 1, width = 4, height = 5, Füllung simple
|kk| // mit Zeichen 'k'
|kk|
|kk|
+--+
```

Eine Füllung kann aus einem Bereich von Zeichen bestehen (`RangeFill`) bspw. 0 – 9, in diesem Fall soll sich die Füllung wiederholen, sobald das letzte Zeichen aus dem Range ausgegeben wurde:

```
+-----+ // x = 0, y = 0, width = 7, height = 7, Füllung range
|01234| // mit Zeichen zwischen '0' und '9'
|56789|
|01234|
|56789|
|01234|
+-----+
```

Eine Füllung kann aus einer beliebigen selbsterstellten Zeichenkette bestehen (PatternFill). Das Muster in der Zeichenkette soll entsprechend wiederholt werden, wenn das letzte Zeichen ausgegeben wurde.

```
+-----+ // x = 0, y = 0, width = 14, height = 7  
|:::~::~:~| // Füllung pattern: ":::~:"  
|:::~::~:~|  
|:::~::~:~|  
|:::~::~:~|  
|:::~::~:~|  
|:::~::~:~|  
+-----+
```

- Machen Sie sich mit den gegebenen Strukturen `Rect` und `Fill` sowie der Funktion `Fill make_pattern_fill(String s)` vertraut.
- Implementieren Sie die Funktionen `Fill make_range_fill(char lower, char upper)`. Diese soll eine Struktur `Fill` zurückgeben, die alle Informationen für eine `RangeFill` Füllung enthält. Stellen Sie über `Preconditions` sicher, dass `lower` immer kleiner oder gleich `upper` ist und beide nur im Bereich `[1, 127]` liegen.
- Implementieren Sie die Funktion `Fill make_simple_fill(char c)`. Diese soll eine Struktur `Fill` zurückgeben, die alle Informationen für eine `SimpleFill` Füllung enthält. Stellen Sie mit `Preconditions` sicher, dass `c` aus dem Intervall `[1, 127]` ist und gleichzeitig ein darstellbares (druckbares) Zeichen ist. Die Funktion `isprint` kann Ihnen behilflich sein.
- Implementieren Sie die Funktion `Rect make_rect(int x, int y, int width, int height, Fill fill)`. Diese soll ein mit den übergebenen Parametern initialisiertes Rechteck bereitstellen. Stellen Sie mit `Preconditions` sicher, dass die Parameter `x`, `y`, `width` und `height` nicht negativ sind. Prüfen Sie für `fill`, dass die übergebene Struktur eine der drei beschriebenen Varianten ist.
- Implementieren Sie die Funktion `void draw_rect(Rect rect)`, die ein `Rect` übergeben bekommt und dieses abhängig von seinen Eigenschaften auf der Konsole darstellt. Einige Beispiele sind hier gegeben, andere im Quelltext. Sie dürfen für diese Aufgabe beliebige Hilfsfunktionen erstellen.
- Geben Sie mindestens 6 verschiedene Beispielrechtecke auf der Konsole aus. Stellen Sie sicher, dass jede Füllungsvariante mindestens einmal vorkommt.

Aufgabe 2: Währungen umrechnen Bitcoin, Euro, Dollar & Dänische Krone

Die Template-Datei für diese Aufgabe ist `money.c`. Die Struktur `Money` repräsentiert eine Menge von Geld in einer bestimmten Währung. Die möglichen Währungen (Bitcoin, Dollar, Euro und Dänische Krone) sind über die Aufzählung `Currency` definiert. Die Umrechnungskurse sind in der Template-Datei angegeben. Gegeben sind außerdem eine Konstruktorfunktion `make_money` und eine Testfunktion `test_within_value` für `Money` Strukturen.

a) Implementieren Sie die Funktion `print_money`, die Geld in Abhängigkeit der Währung in folgendem Format ausgibt:

```
1234.12 $
4.75423414 Bitcoin // <- 8 Nachkommastellen
33.30 Euro
70.75 DKK
```

b) Implementieren Sie die Funktion `Money to_Currency(Money m, Currency target_currency)`. Diese soll Geld in einer bestimmten Währung in eine Zielwährung umrechnen. Fügen Sie zunächst mindestens 5 sinnvolle Tests zu `to_currency_test` hinzu.

c) Implementieren Sie die Funktion `int compare(Money m, Money v)`. Diese soll zwei Mengen an Geld vergleichen und `0` zurückgeben, wenn `m` und `v` den gleichen Wert repräsentieren (Toleranz 0.01), `-1` zurückgeben, wenn `m` einen kleineren Wert als `v` hat und `1` zurückgeben, wenn `w` einen größeren Wert als `v` hat. Fügen Sie zunächst mindestens 5 Tests zu `compare_test` hinzu.

Aufgabe 3: 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') (Typ `char`), und die letzte die Körpergrößen in Meter (Typ `double`). Ihr Programm soll alle enthaltenen Daten einlesen können. Berechnen Sie dann diese statistischen Angaben:

- Das durchschnittliche Geburtsjahr (gerundet auf ganze Jahre)
 - Die Anzahl Männer in der Tabelle
 - Die Anzahl Frauen in der Tabelle
 - Die durchschnittliche Körpergröße der Männer
 - Die durchschnittliche Körpergröße der Frauen
- a) Entwerfen Sie eine Struktur namens `Statistics`, die die genannten Angaben repräsentieren kann.
- b) Schreiben Sie eine Konstruktorfunktion `make_statistics`, welche die Komponenten der Struktur auf `0` bzw. `0.0` initialisiert.

- c) Schreiben Sie eine Funktion `print_statistics`, welche die Komponenten der Struktur in folgender Form auf der Konsole ausgibt:
- ```
mean year: 1921
number males: 12
number females: 28
mean height males: 1.28 m (zwei Nachkommastellen)
mean height females: 2.17 m (zwei Nachkommastellen)
```
- d) Schreiben Sie eine 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. Benutzen Sie zur Verarbeitung des Strings die Funktionen `s_length`, `s_get` und `s_sub`. 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`.

#### Aufgabe 4: Volumen geometrischer Körper

In einem Programm sollen verschiedene Formen von geometrischen Körpern – nämlich Zylinder, Kugeln und Quader – repräsentiert werden. Entwickeln Sie Funktionen, die diese geometrischen Körper verarbeiten kann und das zugehörige Volumen berechnet.

Verwenden Sie die im Skript unter [Recipe for Variant Data](#) beschriebenen Schritte. Verwenden Sie die Template-Datei `volume.c`.

- Implementieren Sie die Struktur `GeomObject` zur Repräsentation eines geometrischen Objekts. Die möglichen Varianten sind `Cylinder`, `Sphere` und `Cuboid`. Nutzen Sie die entsprechenden Strukturen.
- Implementieren Sie die Konstruktorfunktionen `make_cylinder`, `make_sphere` und `make_cuboid`. Nutzen Sie Preconditions um ungültige Eingaben (negative Zahlen) abzufangen.
- Implementieren Sie die Funktion `volume`. Nutzen Sie Pre- und Postconditions um die Funktion zu schützen. Stellen Sie so sicher, dass nur Strukturen mit gültigem Tag verarbeitet werden. Prüfen Sie am Ende, ob das Volumen korrekt berechnet wurde. Nutzen Sie dafür Postconditions und berechnen Sie aus den gegebenen Volumen abhängig von Tag die einzelnen Eingabegrößen. Bspw. für Sphere Volumen = 113.079 ->  $r = \text{pow}(113.079/M\_PI/4.0 * 3.0, 1/3.0)$ .
- Wofür ist die Enumeration `Tag` wichtig? Wo und wie wird sie genutzt? Fügen Sie die Antwort als Kommentar in Ihre Quelldatei ein.

Hinweise zum Editieren, Compilieren und Ausführen:

- mit Texteditor `file.c` editieren und speichern
- `make file` ← ausführbares Programm erstellen
- `./file` ← Programm starten (evtl. ohne `./`)
- Die letzten beiden Schritte lassen sich auf der Kommandozeile kombinieren zu:  
`make file && ./file`