Prof. Dr. Ralf Gerlich
Fakultät Informatik
ralf.gerlich@hfu.eu

Hochschule Furtwangen
Robert-Gerwig-Platz 1
D-78120 Furtwangen

Praktikum Programmieren
Übungsblatt 7
Termin: 30. November 2023

Übung 1 *„Taschenrechnerfunktion“*

Schreiben Sie eine Funktion `int berechne(char op, int v1, int v2)`, die das Ergebnis des Ausdrucks `v1 op v2` zurückliefert. Für den Parameter `op` sind dabei die Werte `'+'`, `'-'`, `'*'` und `'/'` zulässig.

Wird ein gültiger Operator übergeben, soll der Wert -999 geliefert werden. Testen Sie die Funktion mit unterschiedlichen Parameterwerten, aber mit mindestens einem Aufruf pro zulässigem Operator und zusätzlich einem Aufruf mit einem unzulässigen Operator.

Übung 2 *„Kommandozeilenrechner“*

Schreiben Sie ein Programm, das von der Kommandozeile aufgerufen werden kann und Berechnungen wie in Aufgabe 1 ausführt: Mögliche Aufrufe:

```
Rechner + 10 20
Rechner - 30 10
```

Verwenden Sie zur Berechnung des Ergebnisses die Funktion `berechne` aus Aufgabe 1. Prüfen Sie typische Fehlersituationen ab und geben Sie entsprechende Fehlermeldungen aus:

- Zu wenige oder zu viele Argumente, oder
- unzulässige bzw. inkorrekte Argumente.

Nutzen Sie zur Umwandlung der Zahlen-Strings in eine Zahl die Funktion `int atoi(const char* s)` aus der Bibliothek `<stdlib.h>`

Übung 3 *„Größter gemeinsamer Teiler“*

Schreiben Sie eine Funktion `long ggt(long x, long y)`, die den größten gemeinsamen Teiler für zwei natürliche Zahlen `x` und `y` nach dem Algorithmus von Euklid berechnet. Dieser Algorithmus ist wie folgt beschrieben:

- Falls $x = y$, ist das Ergebnis `x`.
- Falls $x > y$, ist das Ergebnis `ggt(x-y, y)`.
- Falls $x < y$, ist das Ergebnis `ggt(x, y-x)`.

Erstellen Sie zwei Varianten der Funktion `ggt`:

1. Verwenden Sie eine `while`-Schleife, um die Zahlenwerte `x`, `y` schrittweise so lange zu verkleinern, bis beide den gleichen Wert haben,
2. Erstellen Sie eine Lösung, bei der sich die Funktion `ggt` rekursiv selbst aufruft.

Prof. Dr. Ralf Gerlich
Fakultät Informatik
ralf.gerlich@hfu.eu

Hochschule Furtwangen
Robert-Gerwig-Platz 1
D-78120 Furtwangen

Übung 4 *„Primzahlen“*

Schreiben Sie eine Funktion `bool prim(int x)`, die genau dann `true` zurückgibt, wenn `x` eine Primzahl ist. Um zu prüfen, ob `x` eine Primzahl ist, durchlaufen Sie die Zahlen $y = 2, 3, \ldots$, und prüfen Sie, ob `x` durch `y` teilbar ist.

Welches ist die höchste Zahl `y`, für die Sie diese Prüfung durchführen müssen?

Übung 5 *„Worte und Zeichen 1“*

Schreiben Sie eine Funktion `void stat(const char* s)`, die eine Statistik für die Zeichenkette `s` erstellt und diese innerhalb der Funktion wie folgt ausgibt:

```
Anzahl der Zeichen = ...
Anzahl der Worte = ...
```

Worte sind durch eine oder mehrere Leerstellen getrennt. Leerstellen zählen nicht als Zeichen.

Nutzen Sie die Funktion `int isspace(char ch)` aus der Bibliothek `<ctype.h>`. Diese liefert genau dann einen Wert ungleich 0 zurück, wenn `ch` einer Leerstelle (Leerzeichen, Tabulator, Newline, Carriage Return) entspricht.

Testen Sie Ihre Funktion mit verschiedenen Eingabestrings, zumindest aber mit einem leeren String, einem String der keine Leerstelle enthält, einem String, der nur Leerstellen enthält und einem String, der mehr als ein Wort enthält.

Übung 6 *„Worte und Zeichen 2“*

Schreiben Sie eine Funktion `void stat(const char* s, int* z, int* w)`, die wie in Aufgabe 5 die Zahl der Zeichen und Worte im String `s` zählt. Statt diese auszugeben, sollen diese jedoch an die Stellen geschrieben werden, auf die Pointer `z` bzw. `w` zeigen.

Nach dem folgenden Beispielaufruf soll also `zeichen==9` und `worte==2` gelten:

```
1 int zeichen, worte;
2 stat("Hallo Welt", &zeichen, &worte);
```

Schreiben Sie ein Programm, das einen String einliest, `stat` entsprechend aufruft und die Zahl der Zeichen und Worte ausgibt.

Testen Sie Ihr Programm erneut wie in Aufgabe 5.