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

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

Praktikum Programmieren
Übungsblatt 4
Termin: 09. November 2023

Übung 1 *„for-Schleifen“*

Geben Sie für die folgenden `for`-Schleifenköpfe an, wie oft diese durchlaufen werden und welche Werte die Zählervariable dabei annimmt:

1. `for (int zaehler = 0; zaehler != 10; zaehler = zaehler + 1)`
2. `for (int n = 10; n > 0; n = n - 1)`
3. `for (int x = 1; x <= 15; x = x + 3)`
4. `for (int x = 1; x != 15; x = x + 3)`
5. `for (int x = 1; x == 15; x = x + 3)`
6. `for (char c = '5'; c <= '9'; c = c + 2)`

Übung 2 *„Prüfzifferberechnung“*

Auf Verpackungen von Waren befinden sich häufig Strichcodes mit Code-Nummern. In der Breite wird die sogenannte Europäische Artikel-Nummerierung mit 13 Ziffern (EAN-13) eingesetzt. Bei Büchern wird seit vielen Jahren ein ähnlicher Code, die Internationale Standard-Buchnummer (ISBN), verwendet. Bis 2007 hatten die ISBN dabei 10 Stellen, seit 2007 werden für neu herausgegebene Bücher verpflichtend 13 Stellen eingesetzt.

Beide Codes beinhalten eine Prüfziffer, d.h. eine Ziffer wird gemäß einer definierten Formel aus den restlichen Ziffern abgeleitet.

Um zu prüfen, ob eine 13-stellige ISBN oder EAN korrekt ist, werden die Ziffern von vorne beginnend abwechselnd mit 1 bzw. 3 multipliziert und die Ergebnisse zusammenaddiert. Ist diese Summe durch 10 teilbar, ist die Nummer korrekt, ansonsten enthält sie einen Fehler.

Ein Beispiel: Die ISBN eines Buches laute 9783446464704. Die Summe ergibt sich zu $1*9+3*7+1*8+3*3+1*4+3*4+1*6+3*4+1*6+3*4+1*7+3*0+1*4 = 110$. Diese ist durch 10 teilbar, die ISBN ist somit korrekt.

Schreiben Sie ein Programm, das für eine 13-stellige ISBN oder EAN prüft, ob der Code korrekt ist, und eine entsprechende Meldung ausgibt. Legen Sie dafür den Code als Zeichenkette in einem Array passender Länge mit Elementen vom Typ `char` ab und nutzen Sie zur Berechnung eine `for`-Schleife. Testen Sie Ihre Lösung an Hand von „greifbaren“ Artikeln mit Code-Nummern.

Übung 3 *„Fibonacci-Zahlen“*

Leonardo da Pisa (Beiname: „figlio de Bonacci“, „Sohn des Bonacci“) untersuchte im dreizehnten Jahrhundert das Wachstum einer Kaninchenpopulation. Das Ergebnis war die sogenannte Fibonacci-Folge (f_k):

- $f_1 = f_2 = 1$
- $f_k = f_{k-1} + f_{k-2}$ für $k > 2$.

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

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

Die Zahl f_k gibt an, wie viele Kaninchenpaare im Monat k vorhanden sind, wenn jedes Kaninchenpaar jeden Monat ein weiteres Kaninchenpaar zur Welt bringt, aber die Kaninchen erst nach zwei Monaten geschlechtsreif sind. Unabhängig davon, wie realistisch diese Annahmen sind, hat die Folge eine interessante mathematische Eigenschaft: Der Quotient $\frac{f_k}{f_{k-1}}$ strebt mit steigendem k gegen den sogenannten Goldenen Schnitt $\frac{1+\sqrt{5}}{2} \approx 1,6180$.

Überzeugen Sie sich davon, indem Sie die ersten 20 Werte der Fibonacci-Folge berechnen und ausgeben. Geben Sie zusätzlich jeweils den Quotienten $\frac{f_k}{f_{k-1}}$ als Gleitkommazahl aus.

Die Ausgabe sollte so aussehen:

```
f_3  =     2 2.000000000000000
f_4  =     3 1.500000000000000
f_5  =     5 1.666666666666667
f_6  =     8 1.600000000000000
f_7  =    13 1.625000000000000
f_8  =    21 1.615384615384615
f_9  =    34 1.619047619047619
f_10 =    55 1.617647058823529
f_11 =    89 1.618181818181818
f_12 =   144 1.617977528089888
f_13 =   233 1.618055555555556
f_14 =   377 1.618025751072961
f_15 =   610 1.618037135278515
f_16 =   987 1.618032786885246
f_17 =  1597 1.618034447821682
f_18 =  2584 1.618033813400125
f_19 =  4181 1.618034055727554
f_20 =  6765 1.618033963166706
```

Übung 4 *„Heron-Verfahren“*

Um etwa 100 n. Chr. beschrieb der griechische Mathematiker Heron von Alexandria ein iteratives Verfahren zur Berechnung der Quadratwurzel einer positiven Zahl. Das Verfahren wurde nach ihm als „Heron-Verfahren“ benannt, obwohl es bereits etwa 1000 Jahre früher den Babyloniern bekannt war.

Gegeben sei eine positive Zahl A und eine erste Näherung x_0 der Quadratwurzel von A. Nun wird wiederholt die Formel

$$x_{k+1} = \frac{x_k + \frac{A}{x_k}}{2} \tag{1}$$

angewendet, um jeweils aus der vorangegangenen Näherung x_k eine neue, bessere Näherung x_{k+1} der Wurzel von A zu bestimmen. Das Verfahren zeichnet sich dadurch aus, dass bereits nach wenigen Schritten eine recht gute Näherung des tatsächlichen Werts erreicht wird.

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

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

Für $A = 2$ und $x_0 = 1$ ergibt sich etwa die Folge:

$$x_1 = \frac{1 + \frac{2}{1}}{2} = 1,5 \tag{2}$$

$$x_2 = \frac{1,5 + \frac{2}{1,5}}{2} \approx 1,416667 \tag{3}$$

$$x_3 = \frac{1,416667 + \frac{2}{1,416667}}{2} \approx 1,414216 \tag{4}$$

$$x_4 = \frac{1,414216 + \frac{2}{1,414216}}{2} \approx 1,414214 \tag{5}$$

Der tatsächliche Wert von $\sqrt{2}$ (auf 6 Nachkommastellen gerundet) beträgt $1,414214$.

1. Schreiben Sie ein Programm, dass eine Zahl A einliest und mit Hilfe einer `for`-Schleife die Näherungswerte $x_1, x_2, \ldots, x_10$ ausgibt. Nehmen Sie als ersten Näherungswert $x_0 = 1$ an.

2. Ersetzen Sie die `for`-Schleife aus Teilaufgabe 1 durch eine `do-while`-Schleife. Es sollen so lange die Werte für $x_1, x_2, \ldots$ ausgegeben werden, bis der Abstand $x_{k+1} - x_k$ zweier aufeinanderfolgender Werte betragsmäßig kleiner als $0,000001$ ist.

Übung 5 *„Geschachtelte for-Anweisungen“*

Das folgende Programmfragment gibt eine Anzahl von „*“-Zeichen in Form eines Dreiecks aus. Ändern Sie das Programm so, dass dieses Dreieck auf dem Kopf steht.

```
1 int zeile, spalte;
2 for (zeile = 1; zeile <= 20; zeile = zeile + 1) {
3   for (spalte = 1; spalte <= zeile; spalte = spalte + 1) {
4     printf("*");
5   }
6   printf("\n");
7 }
```

Übung 6 *„Binärdarstellung einer Zahl“*

Lesen Sie eine ganze Zahl ein. Wurde eine negative Zahl eingegeben, geben Sie eine entsprechende Meldung aus. Ansonsten (bei einer nicht-negativen Zahl) – geben Sie die Binärdarstellung der Zahl in umgekehrter Reihenfolge (niederwertigstes Bit zuerst) aus.