Aufgabe 3.0 Wechsel der Entwicklungsumgebung

Lade die Datei `skorbut.jar` herunter und starte sie unter Windows per Doppelklick. Linux oder Apple Macintosh? Öffne ein Terminal im Download-Verzeichnis und gib ein:

```
java -jar skorbut.jar
```

Aufgabe 3.1 Summen

Tippe das folgende Programm ab:

```
void aufgabe_3_1()
{
    double a;
    double b;
    double sum;

    printf(" Erste Zahl? ");
    scanf("%lf", &a);

    printf("Zweite Zahl? ");
    scanf("%lf", &b);

    sum = a + b;
    printf("%f + %f = %f\n", a, b, sum);
}

int main()
{
    aufgabe_3_1();
    return 0;
}
```

Diskutiere mit Deinem Betreuer folgende Fragen:

- Was bedeutet `double a;`
- Was bedeutet `scanf("%lf", &a);`
- Kann man `sum = a + b;` auch weiter oben hinschreiben?
- Was bedeutet `printf("%f + %f = %f\n", a, b, sum);`
- Wie viele Variablen werden innerhalb von `aufgabe_3_1` definiert?
- Wie viele Funktionsaufrufe befinden sich innerhalb von `aufgabe_3_1`?
- Wie viele Argumente werden beim letzten `printf`-Aufruf übergeben?

Aufgabe 3.2 Quadratwurzeln

Das sog. Newton-Raphson-Verfahren kann zum Berechnen von Quadratwurzeln genutzt werden. Dieses startet mit einem geschätzten Wert g_0 (z.B. 1 oder x) und verfeinert diesen Schätzwert nach der Formel „g_{n+1} ist der Mittelwert aus: g_n sowie x geteilt durch g_n“.

Hier ein Beispiel zum Berechnen der Quadratwurzel von x = 100:

g0 = 1

g1 = (1 + 100/1) / 2 = 50,5

g2 = (50,5 + 100/50,5) / 2 = 26,24

g3 = (26,24 + 100/26,24) / 2 = 15,03

g4 = (15,03 + 100/15,03) / 2 = 10,84

g5 = (10,84 + 100/10,84) / 2 = 10,03

Wende das Verfahren handschriftlich für mindestens zwei weitere Zahlen an.

Schreibe eine Funktion `double root(double x)`, welche die Quadratwurzel von x mit einer festen Anzahl von Schritten (z.B. 5 wie im obigen Beispiel) annähert.

Schreibe eine Funktion `aufgabe_3_2`, die folgendes Verhalten an den Tag legen soll:

```
Herzlich willkommen zum Berechnen von Quadratwurzeln!

Radikand? 100
  Wurzel: 10.032579
```

(Die Eingabe des Benutzers ist fett hervorgehoben.)

Aufgabe 3.3 Rechtwinklige Dreiecke

Schreibe eine Funktion `aufgabe_3_3`, die folgendes Verhalten an den Tag legen soll:

```
Herzlich willkommen zum Berechnen rechtwinkliger Dreiecke!

   Ankathete? 3.0
Gegenkathete? 4.0
  Hypotenuse: 5.000023
```

(Die Eingaben des Benutzers sind fett hervorgehoben.)

Die Länge der Hypotenuse kannst Du mit dem Satz des Pythagoras bestimmen. Um die Wurzel einer Zahl zu berechnen, verwende die Funktion `root` aus der vorigen Aufgabe.

Aufgabe 3.4 Quadratische Gleichungen

Schreibe eine Funktion `aufgabe_3_4`, die folgendes Verhalten an den Tag legen soll:

```
Herzlich willkommen zum Lösen quadratischer Gleichungen!


+---------------------+
| a*x*x + b*x + c = 0 |
+---------------------+


a? 4.0
b? 5.0
c? -6.0


x1: -2.000000
x2: 0.750000
```

(Die Eingaben des Benutzers sind fett hervorgehoben.)

ACHTUNG: Nicht jede Gleichung der Form $ax^2+bx+c = 0$ hat genau 2 Lösungen! Welche Sonderfälle kannst Du identifizieren? Das Programm sollte sinnvoll darauf reagieren!

Wenn Du den kompletten Code in `aufgabe_3_4` packst, wird der Code sehr unübersichtlich. Es empfiehlt sich, die Sonderfälle in eigene Funktionen auszulagern.