Übungsblatt 5 zur Vorlesung Grundlagen der Programmierung

Übung 1

Rätselraten

Schreiben Sie ein Spiel, welches nach folgenden Regeln funktioniert:

1) Der Nutzer wird gebeten sich eine Zahl zwischen 0 und 100 auszudenken und zu merken.
2) Das Programm versucht diese Zahl zu erraten.
3) Bei jedem Vorschlag des Programms muss der Nutzer bewerten, ob der Vorschlag für die geheime Zahl
   - korrekt (k),
   - zu hoch (h) oder
   - zu niedrig (n) ist.
4) Auf Basis des Feedbacks wird ein neuer (möglichst kluger) Vorschlag erzeugt

Bitte nutzen Sie zur Lösung die binäre Suche.

Beispiel-Ausgabe:

```
Denke Dir eine Zahl zwischen 0 und 100 aus und merke sie Dir!
Ist die geheime Zahl 50?
Tippe k für korrekt, n für zu niedrig, h für zu hoch! – h
Ist die geheime Zahl 25?
Tippe k für korrekt, n für zu niedrig, h für zu hoch! – n
[… weitere Fragen und neue Schätzungen …]
Ende. Deine geheime Zahl war 37
```

Übung 2

Quadrat- und Kubikwurzeln

In der Vorlesung haben wir den Wert der Quadrat- und Kubikwurzeln mithilfe der binären Suche bestimmt, jedoch die Lösung auf Werte ≥ 1 beschränkt.

Bitte passen Sie den Code für die Bestimmung der Quadrat- und Kubikwurzeln so an, dass er auch für Werte < 1 funktioniert.

Übung 3

Division von zwei Zahlen

Wir möchten in dieser Aufgabe zwei Zahlen a und b dividieren. Zur Vereinfachung nehmen wir an, dass $a \geq b > 1$.

Bitte implementieren Sie die Aufgabe mithilfe der binären Suche!

Übung 5

Vorkommen von Zeichen in sortiertem String

Schreiben Sie ein Programm, welches in einem alphabetisch sortierten String (z.B. „aabbbbcccfggghiklmnxz") mithilfe der binären Suche effizient prüft, ob ein bestimmtes Zeichen vorkommt.

Test Cases:

String	Zu suchendes Zeichen
aabbbbcccfggghiklmnxz	g
aaab	b
aaab	a
abbcddffhiz	d

Übung 4

Das Minimum von zufällig erzeugten Quadratischen Funktionen - Linear

Wir haben in der letzten Aufgabe die Kubikwurzel von beliebigen ganzzahligen Werten ermittelt, indem wir einfach alle möglichen Werte von Null startend geprüft haben. Wir wollen nun dieselbe Idee auf die Bestimmung des Wertes x, für den die quadratische Funktion $f(x) = ax^2 + bx + c$ ihr Minimum erreicht, übertragen. Die Werte für a, b und c werden zufällig erzeugt. Zur Vereinfachung sind die Werte für a auf das Intervall $a \in [0.5, 1.0]$ beschränkt. Abb. 1 zeigt drei Beispiele zufällig parametrisierter quadratischer Funktionen. Ziel ist es, jeweils den Wert für x zu bestimmen, an dem die auf diese Weise zufällig parametrisierten Funktion ihr Minimum im Intervall $x \in [-10, 10]$ erreicht. Der Wert soll zunächst mit der linearen Suche ermittelt werden.

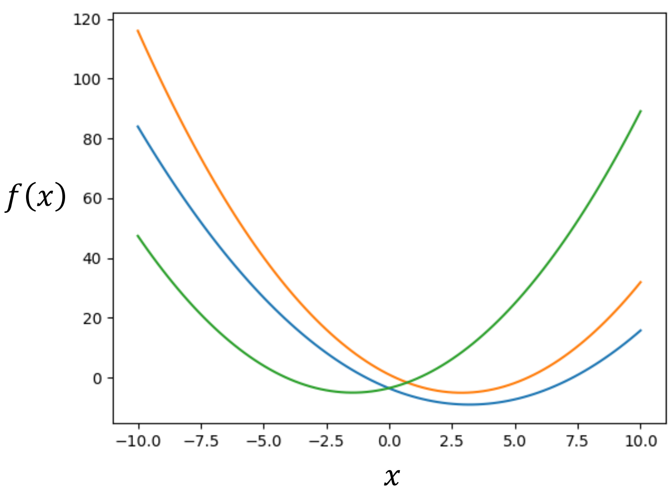

Abb. 1: Drei zufällig parametrisierte Quadratische Funktionen der Form $f(x) = ax^2 + bx + c$

Im Folgenden finden Sie wieder den Rahmen für Ihr Programm, welches Sie lediglich ab dem Kommentar "`Ihr Code kommt hier`" ergänzen müssen. Bitte öffnen Sie vor dem ersten Ausführen noch eine Kommandozeile und geben folgenden Befehl ein:

```
python3 -m pip install matplotlib
```

Hierdurch werden noch einige notwendige Bibliotheken nachgeladen, die Sie zur Visualisierung des Programmbeispiels benötigen.

Hinweise zur Lösung:

- Zur Bestimmung der Kubikwurzel haben wir stückweise alle ganzzahligen Werte von Null an als Ergebnis ausprobiert und mit dem erwarteten Ergebnis verglichen. Da wir das richtige Ergebnis hier nicht kennen: Wie können wir hier möglichst einfach prüfen, dass wir uns im tiefsten Punkt der Funktion befinden?

Das Minimum von zufällig erzeugten Quadratischen Funktionen - Binär

Wir wollen nun dieselbe Aufgabe mithilfe der *binären Suche* lösen. Sie können dieses Mal mit der Funktion

```
ableitungQuadratischeFunktion(x)
```

den Wert der Ableitung $f'(x) = 2ax + b$ am Punkt x bestimmen.

Hinweise zur Lösung:

- Durch die Einschränkung der Werte für a, b und c finden Sie im Intervall $x \in [-10,10]$ immer das Minimum der Funktion
- Weiterhin ist die Ableitung $f'(x)$ im Intervall $x \in [-10,10]$ stetig steigend. Den Wendepunkt der Funktion finden Sie, wenn die Ableitung den Wert Null annimmt.

```
# Damit der Code funktioniert, bitte auf der Kommandozeile ein Mal
# folgenden Befehl eingeben: python3 -m pip install matplotlib
import matplotlib.pyplot as plt
import numpy as np
import random

# Die Funktion wertQuadratischeFunktion gibt den Wert der
# Quadratischen Funktion der Form f(x) = ax^2 + bx + c zurück
def wertQuadratischeFunktion(x, a, b, c):
    return a * x**2 + b * x + c

# Die Funktion ableitungQuadratischeFunktion gibt die Ableitung an x zurück
def ableitungQuadratischeFunktion(x):
    '''
    x: Position
    returns: Wert der Ableitung an Position x
    '''
    return 2 * a * x + b

# Helfer-Funktion zur visuellen Ausgabe des Ergebnisses
def plotQuadratischeFunktion(minimumQuadratischeFunktion , a, b, c):
    x = np.linspace(-10, 10, 100)
    y = wertQuadratischeFunktion(x, a, b, c)
    plt.plot(x, y)
    plt.plot(minimumQuadratischeFunktion,
             wertQuadratischeFunktion(minimumQuadratischeFunktion, a, b, c),
             marker="x", markersize=10,markerfacecolor="blue")
    plt.show()

# erzeuge eine zufällige Quadratische Funktion der Form
# f(x) = ax^2 + bx + c
a = random.uniform(0.5, 1.0)
b = random.uniform(-5, 5)
c = random.uniform(-5, 5)

# Start des Programms
print("Finde das Minimum der Quadratischen Funktion f(x)=" +
      str(f"{a:.2f}") + "x²" + str(f"{b:+.2f}") + "x" +
      str(f"{c:+.2f}") + " im Intervall [-10,10]")

# Ihr Code kommt hier - das Ergebnis bitte in der
# Variable minimumQuadratischeFunktion ablegen:
minimumQuadratischeFunktion = 7

# Ausgabe Ihres angenäherten Ergebnisses
print("Angenähertes Ergebnis: " + str(f"{minimumQuadratischeFunktion:.2f}"))
# Ausgabe des mathematisch ermittelten Ergebnis zum Vergleich
print("Zum Testen: Mathematisch korrektes Ergebnis: " + str(f"{-b/(2*a):.2f}"))
# Visuelle Ausgabe des Ergebnisses
plotQuadratischeFunktion(minimumQuadratischeFunktion, a, b, c)
```