Aufgaben

Aufgabe 1 | *Dokumentation*

Ziehen Sie sich nochmals den im heutigen Unterricht geschriebenen Code zu Gemüte und erstellen Sie sich eine Zusammenfassung, welche die wichtigsten, heute behandelten Kerninhalte umfasst.

Aufgabe 2 | *Konsolenausgabe*

Erstellen Sie ein Programm, das Ihren Namen, Ihr Alter und Ihr Lieblingsessen in der Konsole einliest. Geben Sie diese Informationen dann in einem Satz aus.

Aufgabe 3 | *Variablen und Operatoren*

Schreiben Sie ein Programm, das eine als Variable hinterlegte Temperatur in Celsius in Fahrenheit umrechnet und dann anschließend das Ergebnis in der Konsole ausgibt. Die Umrechnungsformel lautet wie folgt:

$$F = 1.8 * C + 32$$

mit F: Temperatur in Fahrenheit; sowie C: Temperatur in Celsius

Aufgabe 4 | *Bit-Operatoren*

Schreiben Sie ein Python-Programm, das zwei binäre Zahlen vergleicht: Definieren Sie a = 0b1101 und b = 0b1011, berechnen Sie das bitweise UND, das bitweise ODER und das bitweise XOR (= Exklusives ODER) dieser beiden Zahlen. Prüfen Sie auch, ob das Ergebnis des bitweisen UND eine ungerade Dezimalzahl ist und geben Sie die Ergebnisse in binärer und dezimaler Form aus.

Aufgabe 5 | *if-Bedingungen*

Schreiben Sie ein Python-Programm, das eine Zahl vom Benutzer einliest und folgende Prüfungen durchführt:

1. *Falls die Zahl negativ ist → "Die Zahl ist negativ."*
2. *Falls die Zahl 0 ist → "Die Zahl ist null."*
3. *Falls die Zahl positiv und gerade ist → "Die Zahl ist positiv und gerade."*
4. *Falls die Zahl positiv und ungerade ist, dann:*
   (1) *Falls die Zahl größer als 100 ist → "Die Zahl ist positiv, ungerade und größer als 100."*
   (2) *Ansonsten → "Die Zahl ist positiv und ungerade."*
5. *Bonus: Prüfen Sie, ob die Zahl eine Primzahl ist (s. https://de.wikipedia.org/wiki/Primzahl)! Falls ja, ergänzen Sie die Ausgabe um: "Zusätzlich ist die Zahl eine Primzahl!"*

Aufgabe 6 | *Dokumentation*

Ziehen Sie sich nochmals den im heutigen Unterricht geschriebenen Code zu Gemüte und erstellen Sie sich eine Zusammenfassung, welche die wichtigsten, heute behandelten Kerninhalte umfasst.

Aufgabe 7 | *Schleifen*

Implementieren Sie das klassische FizzBuzz-Spiel. Schreiben Sie hierzu ein Programm, das die Zahlen von 1 bis 100 ausgibt, aber für Vielfache von 3 soll es „Fizz" ausgeben, für Vielfache von 5 „Buzz" und für Vielfache von sowohl 3 als auch 5 „FizzBuzz".

Aufgabe 8 | *Listen (Arrays)*

Erstellen Sie eine Liste mit fünf verschiedenen Obstsorten nach Wahl als Strings und führen Sie anschließend folgende Manipulationen durch:

1. *Fügen Sie der Liste eine „Kiwi“ hinzu.*
2. *Fügen Sie eine „Mango“ an zweiter Stelle der Liste ein.*
3. *Entfernen Sie das letzte Element der Liste.*
4. *Überprüfen Sie, ob eine „Banane“ in der Liste vorhanden ist und entfernen Sie diese ggf.*
5. *Drehen Sie die Reihenfolge der Liste um.*
6. *Geben Sie die Länge der Liste aus.*
7. *Erstellen Sie eine Kopie der Liste und geben Sie beide Listen sowie deren Speicheradressen aus.*

Aufgabe 9 | *Listen und Schleifen*

Schreiben Sie ein Python-Programm, das eine Liste von Temperaturen (in Grad Celsius) speichert und verschiedene Berechnungen und Manipulationen durchführt.

1. *Erstellen Sie eine Liste, welche die folgenden Temperaturen für 7 Tage in °C enthält: [15, 18, 20, 22, 17, 19, 21].*
2. *Berechnen Sie die Durchschnittstemperatur und geben Sie diese in der Konsole aus.*
3. *Bestimmen Sie die höchste und die niedrigste Temperatur (max(), min()) und geben Sie die Werte entsprechend in der Konsole aus.*
4. *Ersetzen Sie alle Temperaturen unter 18°C durch den Wert 18.*
5. *Tauschen Sie nun die Temperatur des letzten mit der des ersten Tages.*

Aufgabe 10 | *Bonusaufgabe: Listen und Schleifen*

Schreiben Sie ein Python-Programm, das eine Liste von simulierten Aktienkursen erstellt und grundlegende Analysen durchführt.

1. *Erstellen Sie eine Liste mit 20 (= ca. 1 Handelsmonat) zufälligen Aktienkursen zwischen 75€ und 150€ für ein Unternehmen.*
2. *Berechnen Sie und geben Sie aus:*
   A. *Den durchschnittlichen Aktienkurs des Monats.*
   B. *Den höchsten und niedrigsten Kurs.*
   C. *Die Anzahl der Tage, an denen der Kurs gestiegen ist.*
   D. *Die Anzahl der Tage, an denen der Kurs gefallen ist*
3. *Finden Sie den Tag, an dem der Kurs am stärksten gestiegen ist.*
4. *Berechnen Sie den gesamten Kursanstieg oder -rückgang über den Monat hinweg.*

Aufgabe 11 | *Dokumentation*

Ziehen Sie sich nochmals den im heutigen Unterricht geschriebenen Code zu Gemüte und erstellen Sie sich eine Zusammenfassung, welche die wichtigsten, heute behandelten Kerninhalte umfasst.

Aufgabe 12 | *2D-Listen*

Erstellen Sie eine 4×4-Liste (= 4 Zeilen u. 4 Spalten) mit zufälligen Zahlen zwischen 1 und 9. Implementieren Sie danach folgende Funktionen:

1. *Geben Sie die Liste in einer schönen Tabellenform auf der Konsole aus.*
2. *Berechnen Sie die Summe aller Zahlen.*
3. *Finden Sie die größte Zahl in der Liste sowie deren Indizes.*

Aufgabe 13 | *Beispiel: Tic-Tac-Toe*

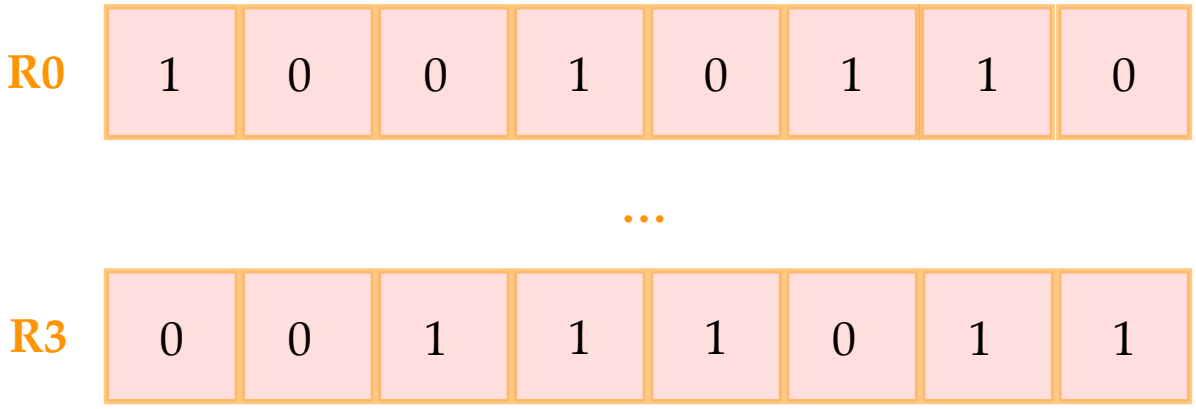
Schreiben Sie ein Tic-Tac-Toe-Spiel für zwei Spieler.

1. *Verwenden Sie eine 3×3-Liste, die zu Beginn mit leeren Zeichen gefüllt ist.*
2. *Die Spieler geben abwechselnd die Zeilen- und Spaltennummer ein, um ein 'X' oder 'O' zu setzen.*
3. *Nach jedem Zug soll das Spielfeld aktualisiert und ausgegeben werden.*
4. *Überprüfen Sie nach jedem Zug, ob ein Spieler gewonnen hat oder das Spielfeld voll ist.*

Aufgabe 14 | *Beispiel: Registerverwaltung in einem Prozessor*

Ein Prozessor speichert Daten in Registern, die als 2D-Liste dargestellt werden.

- *Die Liste besteht aus 4 Registern (R0 bis R3), die jeweils 8 Speicherzellen enthalten.*
- *Implementieren Sie Funktionen, um:*
    1. *Ein Register mit Zufallswerten (0 oder 1) zu füllen.*
    2. *Ein Bit in einem bestimmten Register zu setzen.*
    3. *Ein Register in eine Hex-Zahl umzuwandeln.*
    4. *Alle Register in Binärform auszugeben.*



Aufgabe 15 | *Dokumentation*

Ziehen Sie sich nochmals den im heutigen Unterricht geschriebenen Code zu Gemüte und erstellen Sie sich eine Zusammenfassung, welche die wichtigsten, heute behandelten Kerninhalte umfasst.

Aufgabe 16 | *Refactoring*

Werfen Sie einen Blick auf den im heutigen Unterricht gemeinsam erarbeiteten Lösungsansatz zu „Aufgabe 13“ und nehmen Sie folgende Änderungen / Optimierungen vor (Stichwort „Refactoring“)

1. *Die aktuelle Version sieht ein simples, Exception-Handling vor, sofern ein beliebiger Fehler innerhalb der Anwendung auftritt. Dabei bleibt allerdings die Reihenfolge der Spieler unberücksichtigt. Sorgen Sie dafür, dass bei einer fehlerhaften Eingabe eines bestimmten Spielers dessen Eingabe wiederholt wird, ohne dass unerwartet ein anderer Spieler zum Zuge kommt (Tipp: eine globale „State-Variable“ könnte Ihnen dabei durchaus behilflich sein).*
2. *„DRY“ (= „Don’t Repeat Yourself“) gilt als ein wesentliches Prinzip der modernen Softwareentwicklung. Lagern Sie sonach alle sich wiederholenden Code-Sequenzen in entsprechende Funktionen aus und / oder deklarieren und initialisieren Sie entsprechende Variablen.*
3. *Erweitern / optimieren Sie zuletzt den bestehenden Code ganz nach Ihrem Belieben - auch Python-spezifische Syntax ist hierbei erwünscht.*

Aufgabe 17 | *Bubblesort & Lineare Suche*

Sie arbeiten als Entwickler für einen Online-Shop, der Elektronikartikel verkauft. Kunden können sich Produkte in einer Liste anzeigen lassen, entweder aufsteigend oder absteigend nach Preis. Außerdem möchten sie ein bestimmtes Produkt schnell finden. Zusätzlich sollen sie die Möglichkeit haben, neue Produkte zur Liste hinzuzufügen. Die Anwendung startet mit folgenden Produkten:

[[‚Laptop‘, 1200], [‚Smartphone’, 800], [‚Tablet’, 500], [‚Monitor‘, 300], [‚Maus‘, 50]]

Ihre Aufgabe ist es, eine Python-Anwendung zu schreiben, die folgende Funktionen bietet:

1. *Sortierung der Produktpreise mit Bubblesort:*
   - *Implementieren Sie den Bubblesort-Algorithmus zur Sortierung der Produktliste.*
   - *Der Benutzer kann wählen, ob die Liste aufsteigend (günstigstes Produkt zuerst) oder absteigend (teuerstes Produkt zuerst) sortiert werden soll.*
   - *Der Algorithmus soll optimiert werden, sodass er abbricht, wenn die Liste bereits sortiert ist.*
2. *Lineare Suche nach einem Produkt:*
   - *Implementieren Sie eine Funktion, die mithilfe der linearen Suche nach einem Produktnamen sucht.*
   - *Falls das Produkt existiert, soll der Preis ausgegeben werden, ansonsten eine entsprechende Meldung erscheinen.*
3. *Hinzufügen neuer Produkte:*
   - *Der Benutzer soll ein neues Produkt mit Namen und Preis zur Liste hinzufügen können.*
   - *Die Liste soll danach automatisch sortiert werden (nach der zuletzt gewählten Sortiermethode).*