Programmieren mit Python: Schleifen

Digitalisierung und Programmierung

Übungen

1. Schreibe ein Programm, das die Summe aller Zahlen von 1 bis 100 berechnet und ausgibt. Verwende dazu eine Schleife!

2. Du hast in der Vorlesung im Zusammenhang mit dem **Traveling Salesman Problem** den Brute Force-Algorithmus kennengerlent. Nutze deine Pythonkenntnisse, um folgende Probleme zu lösen:

   a. Schreibe ein Programm zur Berechnung der Laufzeit des Brute-Force-Ansatzes beim Traveling Salesman Problem (TSP). Das Programm soll die Anzahl der Städte als Benutzereingabe entgegennehmen und die Anzahl möglicher Routen zusammen mit der Laufzeit in Sekunden und Jahren ausgeben. Die Laufzeit des Brute-Force-Ansatzes beträgt $O(n!)$, wobei n die Anzahl der Städte ist. Schreibe dazu deine eigene Funktion `factorial()` und verwende darin eine Schleife, um die Fakultät der Anzahl der Städte zu berechnen. Nimm an, dass ein Schritt eine halbe Nanosekunde dauert. Das entspricht einer Prozessortaktung von 2 GHz.

   b. Lass Dir von deinem Programm die Laufzeiten für 10, 15 und 20 Städte ausgeben. Wie viele Jahre würde es jeweils dauern, um alle Routen für die jeweilige Anzahl Städte zu berechnen?

   c. Untersuche welchen Einfluss die Prozessortaktung oder die benötigte Zeit in Nanosekunden für einen Schritt auf die Laufzeit hat. Was passiert, wenn die Prozessortaktung auf 3 GHz erhöht wird?

3. Du hast in der Vorlesung Algorithmen zur **Annäherung der Kreiszahl** π kennengelernt. Versuche dich nun in der Umsetzung mit Python:

   a. Schreibe ein Programm, das die Kreiszahl π mit Hilfe der Formel von Leibniz berechnet. Die Formel lautet: $\pi = 4 \cdot \left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - ...\right)$. Verwende dazu eine Schleife, die die ersten 1000 Glieder der Reihe berechnet. Gib die Annäherung von π nach 1000 Schritten aus.

b. Schreibe ein Programm, das die Kreiszahl π mit Hilfe einer Monte Carlo-Simulation annähert. Die Formel zur Berechnung der Kreiszahl lautet: $\pi = 4 \cdot \frac{N_C}{N_Q}$, wobei N_C die Anzahl der Punkte im Kreis und N_Q die Anzahl der Punkte im Quadrat ist. Verwende dazu eine Schleife, die 1000 Punkte zufällig im Quadrat verteilt und zähle, wie viele Punkte innerhalb des Kreises liegen. Gib die Annäherung von π nach 1000 Schritten aus.

   Hinweis: Du kannst feststellen, ob ein Punkt innerhalb des Kreises liegt, indem du die Entfernung des Punktes zum Kreismittelpunkt berechnest. Nutze dazu die Kreisformel $x^2 + y^2 = r^2$, die bei einem Kreis mit $r = 1$ entsprechend einfach zu überprüfen ist.