Übungsblatt zur Programmierung: Funktionen

Übungsaufgaben zur Programmierung

Prof. Dr. Nicolas Meseth

Reflektionsfragen

1. Erkläre, wann es sinnvoll ist, Code in eine Funktion auszulagern.
2. Was solltest du bei der Bennennung einer Funktion beachten?
3. Funktionen können aus verschiedenen Quellen stammen. Erläutere die folgenden Arten:
   a. Built-in Funktionen
   b. Funktionen aus internen Modulen
   c. Funktionen aus externen Modulen
   d. Selbst geschriebene Funktionen
4. Warum ist es nicht sinnvoll, hinter einer `return`-Anweisung innerhalb einer Funktion weitere Anweisungen hinzuzufügen?
5. Erläutere die Begriffe *Parameter* und *Rückgabewert* im Zusammenhang mit Funktionen in der Programmierung und nenne jeweils ein Beispiel!

Programmieraufgaben

1. Schreibe eine Funktion, die prüft, ob eine Zahl größer als die andere ist. Die Funktion soll beide Zahlen als Parameter entgegennehmen und `True` zurückgeben, wenn die erste Zahl größer ist als die zweite, andernfalls `False`.
2. Definiere eine Funktion `celsius_to_fahrenheit`, die einen Celsius-Wert entgegennimmt und den entsprechenden Fahrenheit-Wert zurückgibt. Die Formel lautet: $°F = °C \times \frac{9}{5} + 32$
3. Schreibe die beiden Funktionen `square(x)` und `cube(x)`. Teste beide, indem du den Benutzer um einen Wert bittest und beide Ergebnisse auf der Konsole ausgibst.
4. Erstelle eine Funktion, die BMI berechnet. Frage den Benutzer nach den benötigen Eingaben und gib das Ergebnis aus.

5. Erstelle eine weitere Funktion, um auf Basis des BMI zu ermitteln, ob eine Person untergewichtig ($< 18{,}5$), normalgewichtig (zwischen 18,5 und 25) oder übergewichtig ist (25).

6. Erstelle eine Funktion, die prüft, ob ein String ein Palindrom ist. Sie soll `True` oder `False`zurückliefern. Groß- und Kleinschreibung soll ignoriert werden.

7. Schreibe eine Funktion `factorial(n)`, die die Fakultät einer ganzen Zahl berechnet. Prüfe, wie das Konzept der *Rekursion* dir dabei helfen kann!

8. Implementiere ein kleines Spiel, bei dem der Computer sich eine Zufallszahl zwischen 1 und 100 ausdenkt, und der Spieler solange raten muss, bis er sie trifft. Implementiere dazu die folgenden Funktionen:

    a. `generate_secret(min, max)` - erzeugt eine zufällige Zahl zwischen `min` und `max`
    b. `get_guess_from_player()` - fragt den Spieler nach seinem Tipp
    c. `check_guess(secret, guess)` - prüft, ob der Tipp zu hoch, zu niedrig, oder richtig war.
    d. **Hinweis**: Du wirst eine Schleife benötigen - lies nach wie das geht.

9. Schreibe eine Funktion `generate_password(length)` die ein Passwort der angegebenen Länge zurückgibt. Das Passwort soll aus Groß- und Kleinbuchstaben, Ziffern und Sonderzeichen bestehen. Hinweise: Schau dir das `string`-Modul in Python an und insbesondere die dort vordefinierten Listen wie `string.ascii_lowercase`, `string.ascii_uppercase`, `string.digits` und `string.punctuation`.