Einführung in die Programmierung

Prof. Dr. Franziska Matthäus
Prof. Dr. Matthias Kaschube
Dr. Karsten Tolle
Kenan Khauto

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Übungsblatt ÜE-04

Ausgabe: 13.11.2024
Abgabe: 23.11.2024
13:00 Uhr

Schatzsucher-Spiel

Hinweis:

- *Es dürfen keine Lösungen aus dem Skript, dem Internet oder anderen Quellen abgeschrieben werden. Diese Quellen dürfen nur mit Quellenangaben verwendet werden und es muss ein hinreichend großer Eigenanteil in den Lösungen deutlich zu erkennen sein.*
- *Digitale Abgaben, die nicht im Format* ***.pdf*** *oder* ***.txt*** *für Texte oder* ***.py*** *für Code erfolgen, werden nicht bewertet. Bei Abgaben mehrerer Dateien müssen diese als* ***.zip*** *zusammengefasst werden.*
- *Achten Sie darauf die Variable* ***__author__*** *in allen Quellcode Dateien (.py) korrekt zu setzen (am Anfang des Quellcodes):* __author__ = "<Matr-Nr>, <Nachname>"
  Beispiel: __author__ = "1234567, Tolle"
  … Leerstellen vor und nach dem „=" und Leerstelle nach dem Komma beachten, sowie keine spitzen Klammern verwenden, die dienen bei der Schreibweise oben zum Definieren von Variablen.
- *Außerdem muss Ihr Name in jeder abgegebenen* ***.pdf*** *und* ***.txt*** *Datei zu finden sein.*
- *Abgaben der Dokumentation, die per Hand geschrieben und eingescannt werden, sind nur in zuvor abgesprochenen Ausnahmefällen erlaubt.*

Aktualisierung der Hinweise:

- *Datei- und Ordnernamen sollen keine Umlaute, diakritische oder Sonderzeichen, mit Ausnahme des Unterstrichs in der Namesmitte, enthalten! (Insb. MAC Nutzer sollten aufpassen, dass nicht zusätzliche Dateien im ZIP-Ordner hinzugefügt wurden! … diese ggf. löschen!)*
- *Beim Programmieren und Kommentieren halten Sie sich an die Regeln im Programmierhandbuch, siehe Moodle-Kurs (Programmierhandbuch WiSe 24/25 (Style Guide)). Im Zweifelsfall gilt PEP 8.*
- *Bitte auch darauf achte, dass Ihre .py Dateien lauffähig sind. Es ist den Tutoren nicht zuzumuten Abgaben zu Debuggen. Probleme, auch was vielleicht nicht geht, kann auskommentiert und mit Kommentaren versehen werden.*

∑ 10 Punkte

Aufgabe ÜE-04-1: Schatzsucher-Spiel (10 Punkte)

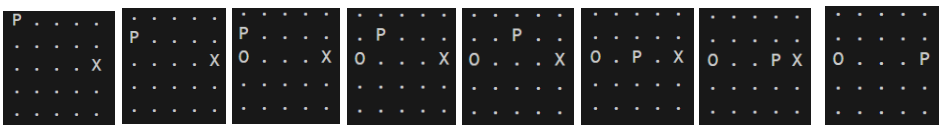
In dieser Aufgabe soll ein Konsolenspiel in Python implementiert werden, bei dem der Spieler in einer 5x5 Karte nach einem Schatz sucht. Der Spieler beginnt oben links und hat eine begrenzte Anzahl an Zügen (10 Züge), um den Schatz zu finden. Auf der Karte gibt es Hindernisse, die der Spieler nicht betreten kann. Ziel ist es, den Schatz zu finden, bevor die Züge aufgebraucht sind.

Vorgaben:

- **Spielfeld:**
  - Erstellen Sie ein 5x5 Gitter als Spielfeld.
  - Standardmäßig sind alle Felder leer (".").
  - Der Schatz ("X") und Hindernisse ("O") werden zufällig platziert.
  - Die Position des Spielers wird mit "P" markiert.
- **Spielmechanik:**
  - Der Spieler gibt Befehle ein ("o" für oben, "u" für unten, "l" für links, "r" für rechts`), um sich zu bewegen.
  - Der Spieler darf nicht außerhalb der Karte ziehen und nicht auf Felder mit Hindernissen gelangen.
  - Nach jedem Zug wird die Karte aktualisiert und angezeigt.
  - Das Spiel endet, wenn der Spieler den Schatz findet oder die Züge aufgebraucht sind.
  - Optional kann die Entfernung zum Schatz bei optimaler Schrittreihenfolge angezeigt werden, um die Suche spannender zu gestalten.
- **Abbruch des Spiels:**
  - Der Spieler kann das Spiel jederzeit mit der Eingabe eines bestimmten Befehls ("q") abbrechen.

Funktionen:

1. **create_map() (1,5 Punkte)**:
   - **Beschreibung**: Erstellt das 5x5 Spielfeld und initialisiert es mit leeren Feldern (".").
   - **Parameter**: Keine.
   - **Rückgabewert**: Eine Liste von Listen (5x5 Gitter).
2. **place_element(game_map, element) (1,5 Punkte)**:
   - **Beschreibung**: Platziert ein Element (z. B. Schatz "X" oder Hindernisse "O") zufällig auf der Karte. Die Platzierung erfolgt nur auf freien Feldern.
   - Hindernisse sollen versteckt bleiben, bis der Spieler auf eins stößt. Danach werden sie auf der Karte angezeigt, damit der Spieler nicht erneut darauf stößt.
   - **Parameter**:
     - game_map: Das Spielfeld (Liste von Listen).
     - element: Das zu platzierendes Element (String).
   - **Rückgabewert**: Ein Tupel (x, y) mit den Koordinaten des platzierten Elements.
3. **print_map(game_map) (1 Punkt)**:
   - **Beschreibung**: Gibt die aktuelle Karte in der Konsole aus. Jede Zeile wird auf einer neuen Zeile angezeigt, und Felder werden durch Leerzeichen getrennt.
   - **Parameter**:
     - game_map: Das Spielfeld (Liste von Listen).
   - **Rückgabewert**: Keinen.



4. **move_player(position, direction, game_map, obstacles) (2 Punkte)**:
    - **Beschreibung**: Berechnet die neue Position des Spielers basierend auf der eingegebenen Richtung. Die Funktion verhindert Bewegungen außerhalb der Karte oder auf Hindernisse. Bei ungültigen Zügen bleibt der Spieler an seiner aktuellen Position.
    - Optional: Ungültige Züge reduzieren nicht die Zuganzahl. (siehe Hinweise zur Dokumentation)
    - **Parameter**:
        - position: Aktuelle Position des Spielers (x, y) (Tupel).
        - direction: Richtung ("o", "u", "l", "r") (String).
        - game_map: Spielfeld (Liste von Listen).
        - obstacles: Liste mit allen Positionen der Hindernisse (Tupel).
    - **Rückgabewert**: Ein Tupel (x, y) mit den neuen Koordinaten des Spielers.
5. **treasure_hunter_game() (4 Punkte)**:
    - **Beschreibung**: Steuert das gesamte Spiel:
        - Initialisiert das Spielfeld und platziert den Schatz und die Hindernisse.
        - Verarbeitet die Eingaben des Spielers und bewegt ihn entsprechend.
        - Zeigt nach jedem Zug die aktualisierte Karte an.
        - Optional: Zeigt die Entfernung zum Schatz an.
        - Beendet das Spiel, wenn der Schatz gefunden wurde oder die Züge aufgebraucht sind.
    - **Parameter**: Keine.
    - **Rückgabewert**: Keinen.

Hinweise zur Dokumentation:
Nutzen Sie Ihre Dokumentation und Docstrings, um Details zu den Regeln und Funktionen klar zu formulieren. Beispiele:

- **Wird die Zuganzahl bei ungültigen Zügen reduziert oder nicht?**
- **Werden Hilfestellungen wie die Entfernung zum Schatz angezeigt? (Im Fall, dass, der Schatz versteckt ist)**
- **Der Schatz kann sichtbar sein, oder die Suche fokussiert sich auf das Finden des Weges.**

Das Spiel kann nach Belieben erweitert werden.

Die Funktionen 1-4 sollten in einem eigenen Modul implementiert werden. Funktion 5 in einem anderen Modul.
Vergessen Sie bitte nicht, die Funktionen 1-4 unter if__name__ == „__main__“ zu testen.