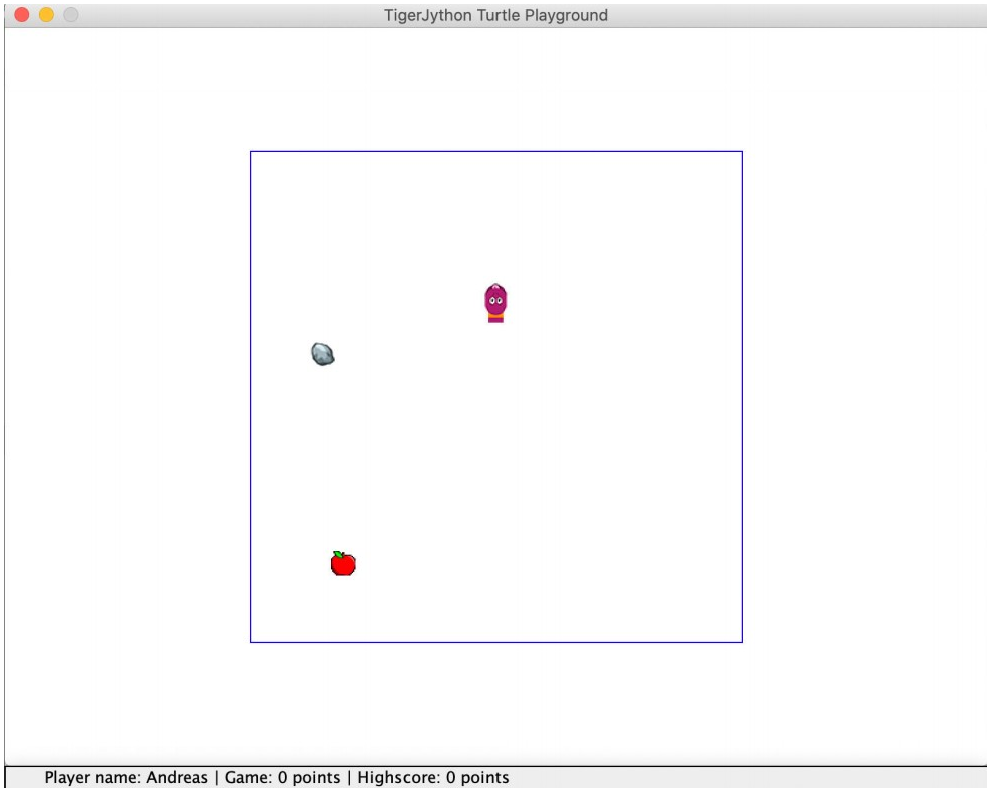
Auftrag 1: Programmieren eines Spiels

In diesem Auftrag entwickeln Sie ein eigenes Spiel in TigerJython. Es ist eine vereinfachte Version des früher auf Mobiltelefonen beliebten Spiels Snake.



Screenshot: Fenster des Spiels Snake in TigerJython

Das Spiel besteht aus einem blauen Quadrat, worin sich eine Schlange bewegt. Ziel des Spiels ist, möglichst viele Punkte zu ergattern. Um einen Punkt zu erhalten, muss die Schlange Futter (einen Apfel) fressen. Dabei muss sie aufpassen, dass sie nicht in die blaue Wand oder in ein Hindernis (einen Stein) läuft. Passiert dies, erklingt ein Ton und das Spiel ist zu Ende.

Hinweise

Mehr Details zu der Funktionsweise des Spiels finden Sie im Anhang am Ende des Dokuments.

Ein Demo-Video zum Spiel finden Sie hier: https://youtu.be/gQg7a6euWkQ.

1 Ziel

Im Auftrag werden Sie in einer praktischen Anwendung die in TigerJython im Kapitel 2 gelernten Themen anwenden, indem Sie das oben beschriebene Spiel «Snake» programmieren.

Sie versuchen, Ihr eigenes Programm in TigerJython zu entwickeln, sodass es dem oben beschriebenen Spiel in Funktionalität und Aussehen so ähnlich wie möglich kommt.

Halten Sie wie bisher im Unterricht Ihren Fortschritt in Ihrem Journal fest. Zusätzlich schreiben Sie eine Dokumentation, welche Ihr Programm im Detail beschreibt.

2 Vorbereitung

Lesen Sie im Skript von TigerJython das Kapitel 2.11 Turtleobjekte und lösen Sie mindestens zwei Aufgaben. Damit haben Sie das Kapitel 2. Turtlegrafik erfolgreich abgeschlossen!

Dieses Kapitel (ausgenommen Kapitel 2.9 Rekursionen und 2.12 Drucken) ist Grundlage für diesen Auftrag und für die erste Prüfung. Verschaffen Sie sich deshalb nochmals einen kurzen Überblick über das Gelernte.

3 Vorgehen

Gehen Sie in Microsoft Teams zum Team «GKG_Inf_2024_B». Laden Sie unter Kursmaterialien die Dateien im Ordner «2020-11-04 Auftrag 1» herunter. Der Ordner enthält bereits alle nötigen Dateien für den Auftrag (inkl. Programm und Dokumentation).

3.1 Programm

Öffnen Sie die Datei `snake.py` in TigerJython und studieren Sie die vorgegebene Struktur. Erweitern Sie das Programm solange, bis es die Funktionalität und das Aussehen hat, welche im Anhang definiert sind. Gewisse Teile des Programms dürfen Sie weglassen, wenn Ihnen die Zeit nicht mehr reicht.

Um das Programm fertigzustellen, können Sie die nachfolgenden Tipps verwenden. Die Tipps sind als «Guide» zu verstehen, falls Sie nicht mehr weiterkommen. Sie dürfen das Spiel selbstverständlich auch ohne diese Tipps programmieren.

Tipp: Globale Variablen verwenden

Gewisse Variablen werden immer wieder verwendet. Diese sind zuoberst im Programm bereits für Sie definiert.

Denken Sie daran, dass Sie diese Variablen nur in einer Funktion überschreiben können, wenn Sie am Anfang der Funktion auf einer Zeile `global ...` schreiben. Ein Beispiel wäre `global points` für die Variable `points`.

Tipp: Einzelne Schritte machen

Unterteilen Sie das Projekt in möglichst viele einzelne Schritte. Schreiben Sie für jeden Schritt eine einzelne Funktion. Beginnen Sie mit den Teilen, die Ihnen leichtfallen.

Folgende Schritte gehören zur Grundfunktionalität des Spiels:

a) Erstellen Sie zuerst die Turtle `frameTurtle` für den blauen Rahmen. Zeichnen Sie mit ihr den Rahmen des Spiels mit der Grösse `frameSize` (im Programm auf 500 gesetzt).

b) Erstellen Sie nun die Turtle `snakeTurtle` für die Schlange. Verwenden Sie den Befehl `snakeTurtle = Turtle(frame, "snake.png", keyPressed = onKeyPressed)`, um das Turtlebild `snake.png` zu verwenden und die Funktion `onKeyPressed` aufzurufen, wenn eine Taste gedrückt wird.

c) Versuchen Sie, mithilfe der vorgegebenen `while`-Schlaufe das Programm ewig laufen zu lassen und die Schlange gemäss der zuletzt gedrückten Taste zu drehen und automatisch zu bewegen. Berücksichtigen Sie, dass sich die Schlange nur um ± 90 Grad drehen kann und nicht um 180 Grad.

d) Ergänzen Sie nun die Funktion `snakeTurtleIsAlive()`, welche überprüft, ob Ihre Schlange noch lebt. Dies bedeutet, dass Sie überprüfen müssen, ob die Koordinaten Ihrer Schlange innerhalb des Rahmens sind. Die Koordinaten der Schlange erhalten Sie mit `snakeTurtle.getX()` und `snakeTurtle.getY()`. Falls die Schlange nicht mehr lebt, soll die `while`-Schlaufe verlassen und das Programm beendet werden. Spielen Sie einen tiefen Ton ab.

e) Erstellen Sie in einer neuen Funktion eine Turtle für den Apfel. Setzen Sie die Turtle mit `appleTurtle.setRandomPos(0.9*frameSize, 0.9*frameSize)` auf eine zufällige Position innerhalb des blauen Rahmens. Geben Sie der Turtle das Bild `apple.png`. Rufen Sie diese Funktion nun an geeigneter Stelle auf.

f) Schreiben Sie eine Funktion, welche überprüft, ob Ihre Schlange den Apfel berührt. Falls ja, verschieben Sie den Apfel an eine neue zufällige Position. Spielen Sie zudem einen hohen Ton ab. Rufen Sie auch diese Funktion an geeigneter Stelle auf.

Wenn Sie a) bis f) geschafft haben und Ihnen die Zeit reicht, dann können Sie noch die Abfrage des Namens, den Punktestand und die Hindernisse einbauen:

g) Erstellen Sie eine Funktion, die solange läuft, bis Sie ENTER drücken. Rufen Sie die Funktion auf, wenn die Schlange stirbt. Sobald Sie ENTER drücken, soll das Spiel neugestartet werden. Dazu können Sie die Funktion `play()` aufrufen.

h) Integrieren Sie eine Abfrage für den Namen mit `inputString(...)` und speichern Sie diesen in der vorgegebenen Variable `name`.

i) Ergänzen Sie die Funktion, wo Sie überprüfen, ob die Schlange einen Apfel frisst. Erhöhen Sie dort die Punktzahl (und allenfalls Höchstpunktzahl) des Spielers, indem Sie die Variablen `points` und `highScore` anpassen.

j) Bauen Sie an geeigneter Stelle eine Funktion ein, welche den Text in der Statusbar verändert. Sie können mit `snakeTurtle.setStatusText(...)` jederzeit in die Statuszeile des Fensters schreiben. Text und Zahlen können Sie mit folgendem Trick mischen: `text = "Hello " + name + ", you have " + str(points) + " points"`. Erklärung: Das `+` kombiniert mehrere Variablen oder Texte. Die Funktion `str(...)` wandelt eine Zahl in einen String (Text) um.

k) Um die Hindernisse zu programmieren, müssen Sie Arrays (Listen) verwenden. Eine leere Liste ist bereits erstellt, nämlich `stoneTurtles = [ ]`. Sie können für jedes neue Hindernis eine Turtle erstellen und diese dann an die Liste mit dem Befehl `stoneTurtles.append(turtle)` anhängen. Wenn Sie die Liste am Ende des Spiels wieder zurücksetzen möchten, können Sie wieder `stoneTurtles = [ ]` setzen.

l) Um herauszufinden, ob die Schlange irgendein Hindernis berührt, müssen Sie mit einer `for`-Schlaufe die Liste durchgehen und die Koordinaten zwischen der Schlange und dem Hindernis geeignet vergleichen.

Tipp: Bleiben Sie dran, fragen Sie nach

Wenn Sie nicht weiterkommen, versuchen Sie online zu recherchieren, helfen Sie sich gegenseitig, fragen Sie in der Stunde oder kontaktieren Sie mich via Teams.

3.2 Dokumentation

Ergänzen Sie das bestehende Journal in Ihrem Portfolio im Postfach jedes Mal, wenn Sie an Ihrem Spiel arbeiten. Dokumentieren Sie im Journal Ihren Fortschritt, Ihre Probleme, Ihre Ideen, usw.

Vergessen Sie jeweils nicht, in Ihrem Code die wichtigsten Stellen mit Kommentaren zu versehen. Sie können sich dabei an der Vorlage orientieren.

Zum Abschluss des Auftrages sollen Sie eine separate Dokumentation erstellen. In dieser beschreiben Sie zuerst in eigenen Worten, wie das Spiel für den Anwender funktioniert. Anschliessend erklären Sie, wie Ihr Programmcode aufgebaut ist. Für einen dritten Teil wählen Sie eine aus Ihrer Sicht wichtige Funktion in Ihrem Programm aus und erklären diese detaillierter.

Hinweise

Achten Sie bei der Dokumentation auf eine saubere Darstellung und verständliche Formulierungen.

Erstellen Sie die Dokumentation in Microsoft Word. Schreiben Sie eine bis zwei Seiten Text in A4, Schriftgrösse 12. Sie können dabei auch Bilder (z. B. Screenshots des Spiels oder Code) einfügen. Die minimale/maximale Anforderung an Seitenzahlen bezieht sich auf reinen Text.

Wenn Sie mit der Dokumentation fertig sind, speichern Sie diese als PDF ab. Ein PDF können Sie beispielsweise mit dem Befehl «Speichern Unter...» in Word generieren.

4 Abgabe

Die Abgabe des Auftrags inklusive Portfolio erfolgt via Microsoft Teams. Laden Sie Ihre Dateien bis spätestens am **Sonntag, 22. November 2020 um 23:59 Uhr** (Back-up Termin 29. November) in das Postfach des Teams GKG_Inf_2024_B hoch.

Achten Sie darauf, dass Ihr Postfach die folgende Struktur hat:

```
TigerJython
├── Aufgaben
│   ├── Kapitel 2.1
│   │   ├── aufgabe1.py
│   │   └── ...
│   ├── Kapitel 2.2
│   │   ├── aufgabe1.py
│   │   └── ...
│   └── ...
├── Auftrag 1
│   ├── apple.png
│   ├── Dokumentation.docx
│   ├── Dokumentation.pdf
│   ├── snake.png
│   ├── snake.py
│   └── stone.png
└── Journal.docx
```

5 Bewertung

Für diesen ersten Auftrag erhalten Sie eine Note, die aus folgenden Anteilen besteht:

a) 50 % Programm-Code des Auftrags: `snake.py`

b) 25 % Dokumentation des Auftrags: `Dokumentation.pdf`

c) 25 % Portfolio: Code und `Journal.docx`

Um Ihren aktuellen Stand abschätzen zu können, überprüfen Sie die Checkliste im Anhang.

Hinweise

Die drei geplanten Prüfungen während des Schuljahres werden als ganze Note gewichtet. Die Note für den Auftrag zählt als ganze Note. Sie werden wahrscheinlich im zweiten Semester einen weiteren Auftrag bearbeiten, der dieselbe Gewichtung wie dieser Auftrag erhält.

Sollte der Schulunterricht aufgrund der Coronavirus-Pandemie umgestellt werden müssen, werden wir die Modalitäten und Termine allenfalls anpassen.

Viel Erfolg!

Anhang 1: Funktionalität des Spiels

Ihr Spiel sollte so programmiert sein, dass es gemäss Beschreibung auf Seite 1 gespielt werden kann. Damit Sie das Spiel sinngemäss programmieren, sollen Sie die nachfolgend zusätzlich beschriebenen Punkte berücksichtigen:

Navigation der Schlange

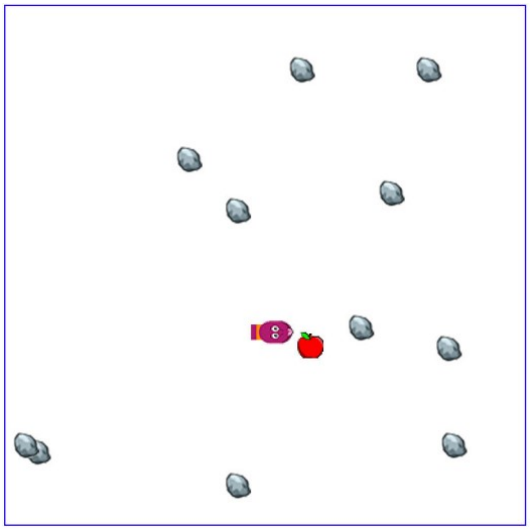
Der Benutzer des Spiels kann die Schlange mit den Pfeiltasten (alternativ, wenn Sie keine Pfeiltasten auf Ihrer Tastatur haben: mit den Buchstaben ASDW) navigieren. Die Geschwindigkeit der Schlange ist auf das Maximum gesetzt. Sie kriecht von alleine; die Pfeiltasten ändern nur die Richtung um jeweils ±90 Grad.

Futter für die Schlange

Die Position des Apfels wird jeweils zufällig im blauen Quadrat gewählt. Sobald ein Apfel gefressen wurde, verschwindet er und ein neuer Apfel wird an einer anderen zufälligen Position gezeigt. Zeitgleich ertönt ein hoher Ton. Zudem wird die Punktzahl in der Statusbar aktualisiert (freiwillig).

Hindernisse für die Schlange (freiwillig)

Um die Schwierigkeit mit Dauer des Spiels zu erhöhen, entsteht alle paar Sekunden ein Hindernis an einer zufälligen Position. Somit füllt sich das blaue Quadrat immer mehr mit Hindernissen, um welche die Schlange navigieren muss.



Screenshot: Das Quadrat füllt sich mit Hindernissen

Abfrage eines Namens (freiwillig)

In einer Statusbar wird der Name des Spielers mit dem aktuellen Punktestand gezeigt. Damit Sie den Namen anzeigen können, geht vor Spielbeginn ein Dialogfenster auf, welches den Namen abfragt.

Screenshot: Abfrage des Namens in Dialogbox

Text in der Statusbar (freiwillig)

Während des Spiels wird in der Statusbar Ihr Name, die aktuelle Punktzahl und die Höchstpunktzahl angezeigt. Ist das Spiel vorbei, kann der Benutzer mit dem Drücken der ENTER-Taste das Spiel neu beginnen. Darauf wird er in der Statusbar hingewiesen.

Screenshot: Hinweis in der Statusbar

Anhang 2: Checkliste

Zur besseren Übersicht kann Ihnen die folgende Checkliste dienen:

Programm

Allgemein:
- ☐ Funktionen und wichtige Punkte sind im Programm kommentiert
- ☐ Das Spiel macht einen in sich abgeschlossen Eindruck

Grundfunktionalität:
- ☐ Blauer Rahmen wird in Fenstermitte gezeichnet
- ☐ Schlange mit Turtlebild wird in Mitte platziert
- ☐ Schlange kriecht von alleine geradeaus
- ☐ Schlange kann mit Pfeiltasten um ±90 Grad gedreht werden
- ☐ Schlange kann *nicht* eine 180-Grad-Drehung machen
- ☐ Apfel mit Turtlebild wird an zufälligem Ort eingefügt
- ☐ Apfel wird neu platziert, wenn die Schlange ihn berührt
- ☐ Hoher Ton ertönt, wenn die Schlange den Apfel berührt
- ☐ Spiel ist zu Ende, wenn die Schlange in den blauen Rahmen fährt
- ☐ Tiefer Ton ertönt, wenn die Schlange in den blauen Rahmen fährt

Motivation und Kreativität (optional):
- ☐ Mit ENTER kann ein neues Spiel begonnen werden
- ☐ Name des Spielers wird vor Spielbeginn abgefragt
- ☐ Punkte werden aktualisiert, wenn die Schlange einen Apfel berührt
- ☐ Statusbar zeigt jederzeit Name, Punktestand und Highscore
- ☐ Steine werden Schritt für Schritt hinzugefügt
- ☐ Spiel ist zu Ende, wenn Schlange einen Stein berührt
- ☐ Statusbar zeigt Hinweis, wenn das Spiel zu Ende ist
- ☐ Es wurden eigene Ideen umgesetzt

Dokumentation

Inhalt:
- ☐ Teil 1: Beschrieben, wie Ihr Programm funktioniert
- ☐ Teil 2: Erklärt, wie Ihr Code grob aufgebaut ist
- ☐ Teil 3: Eine wichtige Funktion ausgewählt und detailliert erläutert

Präsentation:
- ☐ Microsoft Word, 1–2 Seiten A4, Schriftgrösse 12, fliessender Text
- ☐ Dokumentation ist verständlich und korrekt formuliert
- ☐ Am Ende aus dem Word-Dokument ein PDF generiert

Portfolio

- ☐ Pro bearbeitetes Unterkapitel eine vollständig gelöste Aufgabe
- ☐ Journal ist (seit 4.11.2020) für jede Lektion ausgefüllt
- ☐ Journal dokumentiert Ihren Fortschritt im Projekt