Dokumentation Snake Spiel Informatik

## Einleitung

Wir haben uns in den letzten Wochen im Informatikunterricht intensiv mit dem Erstellen eines Spiels in der Programmiersprache Python auseinandergesetzt. Wir sollten mit Hilfe des schon Gelernten aus den TigerJython-Skript-Kapiteln 2.1 – 2.11 das bekannte Spiel Snake programmieren. Die Snake sollte sich jedoch nicht verlängern, da wir dies noch nicht angeschaut haben. Ich habe jede Ergänzung jeweils direkt getestet, um zu schauen, ob die neuen Programmzeilen funktionieren. Einmal benutzte ich auch die Debugger Funktion, weil ich nicht verstand, wieso etwas nicht funktionierte.

## Funktion des Programms

Ich habe probiert meinen Code mithilfe der schon bekannten Befehlen zu schreiben. Dies ging in den meisten Fällen. Ansonsten habe ich im Skript nachgeschaut oder Sie haben es uns in der Stunde erklärt.

Als aller erstes habe ich die Turtle programmiert, welche den Rahmen zeichnet. Als nächstes habe ich die Steuerung der Snake programmiert. Dazu musste ich immer den Computer abfragen, welche Taste als letztes gedrückt wurde und dies dann in einer Variablen speichern. Je nachdem was in der Variable gespeichert wurde, steuert die Snake in eine andere Richtung. Damit sie wie verlangt nicht 180 Grad Drehungen machen kann, habe ich noch vorgesehen, dass sie nur nach links drehen kann, wenn die Richtung vorher auf- oder abwärts war; nicht aber wenn die Richtung rechts war. Dies habe ich dann mit allen Richtungen gemacht. Danach habe ich den Apfel programmiert. Dazu musste ich den Befehl setRandomPos(0.9\*frameSize, 0.9\*frameSize) verwenden, welcher eine zufällige Position innerhalb des Rahmens aussucht. Anschliessend habe ich programmiert, dass die Snake den Apfel essen kann. Dazu werden in jedem Durchlauf die Koordinaten des Apfels abgefragt und die Distanz zur Snake berechnet. Wenn diese kleiner als 15 ist, gilt der Apfel als gegessen und erhält eine neue zufällige Position. Zusätzlich ertönt ein hoher Ton und der Spieler erhält einen Punkt gutgeschrieben. Als nächstes habe ich programmiert, dass die Snake beim Berühren des Rahmens stirbt. Dazu werden die Koordinaten der Snake abgefragt: falls ein Wert höher als 250 oder tiefer als -250 ist, ertönt ein tiefer Ton und die Snake stirbt. Nun hatte ich alles Obligatorische gemacht und konnte mit dem optionalen Teil anfangen. Begonnen habe ich mit der Abfrage des Spielernamens, indem ich einen Befehl hinzugefügt habe, welcher ein Dialogfenster öffnet. In diesem Fenster muss man seinen Namen eingeben, welcher dann in einer Variabel abgespeichert wird. Als Nächstes habe ich den Restart programmiert, welcher sich aktiviert, wenn die Snake tot ist und die Entertaste gedrückt wird. Zusätzlich musste ich die Points zurücksetzen. Ausserdem startet das Spiel wieder. Danach habe ich die Steine programmiert. Dazu musste ich eine Liste erstellen, die automatisch ergänzt wird. Alle 100 Durchläufe der While Schlaufe in der Playfunktion wird ein neuer Stein an einer zufälligen Position erstellt und der Liste hinzufügt. Nun werden in jedem Umlauf überprüft, ob die Snake näher als 15 Pixel ist. Wenn dies bei einem Stein der Fall ist, stirbt die Snake und ein tiefer Ton ertönt. Da die Steine sich bei einem Restart auch verschwinden müssen, habe ich in der Restart Funktion noch alle Steine in der Liste unsichtbar gemacht und diese Liste geleert. Am Ende habe ich noch eine Statusbar erstellt, welche sich immer aktualisiert, wenn ein Apfel gegessen wird und damit die aktuelle Punktzahl angezeigt wird. Und wenn die Punkte höher als der Highscore sind, aktualisiert es auch den Highscore. Ausserdem habe ich noch einen Text hinzugefügt, welcher erst erscheint, wenn die Snake tot ist und darauf hinweist, dass man die Entertaste zum Restarten drücken muss.

## Programmcodes

Ich habe am Anfange des Codes alle Variablen, welche ich später im Code brauche, definiert. Ich habe so viel wie möglich in einzelne Funktionen geschrieben, so dass ich im Hauptteil am Ende des Programmes nur noch die Funktionen aufrufen muss. Ich habe die Funktionen probiert in einer logischen Reihenfolge zu definieren, also habe sie wie folgt aufgeschrieben: Zuerst habe ich das «snakeTurtleIsAlive», dann die Steuerung, dann das Erscheinen des Apfels, dann das Apfelessen und die Punktezählung, dann den Steine, dann den Statusbartext, dann denn Restart, dann die Namensabfragung, dann die Play Funktion und als letztes habe ich die Main Funktion aufgeschrieben.

## Erklärung der eatApple Funtion

Mit def eatApple() gibt man einer neuen erstellten Funktion einen Namen. Da in dieser Funktion das Apfel essen definiert sein soll, habe ich die Funktion eatApple() genannt.

Mit appleTurtle.getX() kriege ich die X Koordinaten des Apfels. Und speichere die Zahl in der Variable posAX . Dasselbe folgt anschliessend für die Y-Koordinate mit appleTurtle.getY() und posAY.

Die Funktion snakeTurtle.distance(posAX, posAY) berechnet die Distanz der Snake zu den abgespeicherten Koordinaten des Apfels, welche in der Variable dist abgespeichert wird.

Mit if dist < dangerZone: kontrolliere ich, ob die Distanz(dist) der Snake zum Apfel kleiner als die dangerZone ist. Der Variable dangerZone habe ich am Anfang des Programmes den Zahlenwert 15 zugewiesen. Wenn diese if Bedingung zutrifft (die Snake hat den Apfel berührt), geht es die Codes innerhalb dieser if Schlaufe durch.

Ich habe global points geschrieben, damit ich den Wert der Variable points für das gesamte Programm verändere und nicht nur für die Funktion.

Da der Apfel jetzt von der Snake gegessen(berührt)wurde, gebe ich mit appleTurtle.setRandomPos(0.9\*frameSize, 0.9\*frameSize) dem Apfel eine neue zufällige Position innerhalb des Rahmens.

Mit points = points +1 erhöhe ich den Zahlenwert der Variable points um eins (Ich kriege einen Punkt dazu).

Der Befehl playTone(700) spielt einen Ton in einer bestimmten Frequenz ab. In meinem Fall ist die Frequenz 700, also ein hoher Ton.

Mit barText() rufe ich eine von mir erstellte Funktion auf, welche den Text in der Status Bar mit den neuen points Wert überschreibt und falls der points Wert höher als der bisherige Highscore Wert ist, wird auch dieser Wert überschrieben.