IT-Grundlagen und Einführung ins Programmieren

Bewertete Abgabe 3

Beschreibung

Entwickeln Sie ein Programm, um das **Rommé** Kartenspiel zu simulieren. Falls Sie das Spiel nicht kennen oder schon länger nicht mehr gespielt haben, finden Sie hier einen guten Überblick (Variante "**Rommé mit Auslegen**"):

https://de.wikipedia.org/wiki/Romm%C3%A9

Lösen Sie die Aufgaben jeweils in nur einem Skript, die Teile innerhalb einer Aufgabe bauen so aufeinander auf, dass das problemlos geht. Folgen Sie den Anweisungen zur Ausgabe, es muss für jede Aufgabe die entsprechende Ausgabe vorhanden sein, wenn das Skript ausgeführt wird.

Aufgabe 1 (60 Punkte):

Teil 1 (20)

Wir beschränken uns auf eine feste Anzahl von drei Spielern. Zunächst sollen Datenstrukturen und wichtige Funktionen vorbereitet werden.

Erstellen Sie Listen für das 104er Rommé-Kartenblatt (zwei 52er Blätter ohne Joker), die Hände der Spieler, den Ablagestapel und den Stapel zum Nachziehen von Karten. Die Karten sollen als Tupel der Form (FARBE[1], WERT[2]), also z.B. ("Karo", "Bube") oder ("Herz", "7") dargestellt werden. **Wir spielen ohne Joker!**

Schreiben Sie eine Funktion `karten_austeilen()`, die das Blatt erstellt, mischt, dem ersten Spieler 14 und den beiden anderen Spielern 13 Karten gibt und die verbleibenden Karten auf den Zugstapel legt. Sie dürfen dazu **globale Variablen** verwenden.

Geben Sie alle Datenstrukturen mittels `print()` aus:

```
Aufgabe 1, Teil 1:
Ablagestapel: []
Hand 1: [("Herz", "Ass"), ("Karo", "4"), …
Hand 2: …
Hand 3: …
Zugstapel: [("Kreuz", "7"), ("Pik", "Bube"), …
```

(Alle Beispiele sind gekürzt. Ihr Programm soll natürlich alle Karten ausgeben. Die Karten sind jede Runde neu gemischt)

[1] Farbe bezieht sich nicht auf rot oder schwarz, sondern auf die vier Spielkartenfarben Kreuz, Pik, Herz und Karo.
[2] Die Werte im 52er Blatt sind 2, 3, 4, 5, 6, 7, 8, 9, 10, Bube, Dame, König, Ass

Teil 2 (10)

Schreiben Sie eine Funktion `karte_ablegen(hand, auswahl)`, die eine ausgewählte Karte auf den Ablagestapel legt. Der Parameter `hand` ist eine Liste, die die Karten enthält. Der Parameter `auswahl` ist eine Zahl, die die Nummer der Karte in der Hand angibt, die abgelegt werden soll.

Zeigen Sie die Hand des ersten Spielers mit den 14 Karten so an, dass man eine Auswahl treffen kann, und lassen Sie die abzulegende Karte per Eingabe bestimmen. Zeigen Sie danach die Hand und den Ablagestapel wieder an:

```
Aufgabe 1, Teil 2:
Hand 1:
Karte 1: (“Herz”, “Ass”)
Karte 2: (“Karo”, “4”)
Karte 3: …
…
Welche Karte ablegen (1-14)? 1
Ablagestapel: [(“Herz”, “Ass”)]
Hand 1: [(“Karo”, “4”), …
```

Teil 3 (10)

Schreiben Sie die Funktion `karte_ziehen (hand, quelle)`, die eine Karte vom Ablage- oder vom Zugstapel nimmt und der Hand hinzufügt. Der Parameter `hand` ist eine Liste, die die Karten enthält, der Parameter `quelle` eine der beiden Stapel-Listen.

Geben Sie den Ablagestapel und die Hand des zweiten Spielers aus und lassen Sie per Eingabe bestimmen, ob die neue Karte vom Ablagestapel oder Zugstapel genommen werden soll. Zeigen Sie die Hand danach an und lassen Sie eine Karte ablegen (s. Teil 2).

```
Aufgabe 1, Teil 3:
Ablagestapel: [(“Herz”, “Ass”)]
Hand 2: [(“Pik”, “8”), …
Karte vom Ablege- oder Zugstapel nehmen (A/Z)? Z
Hand 2:
Karte 1: (“Pik”, “8”)
Karte 2: (“Kreuz”, “Dame”)
Karte 3: …
Welche Karte ablegen (1-14)? 1
Ablagestapel: [(“Herz”, “Ass”), (“Pik”, “8”)]
Hand 2: [(“Kreuz”, “Dame”), …
```

Teil 4 (20)

Schreiben Sie Funktionen

```
ist_satz(hand, auswahl)
ist_sequenz(hand, auswahl)
```

die prüfen, ob eine Auswahl von Karten als Satz bzw. Sequenz ausgelegt werden kann. Der Parameter `hand` erwartet eine Liste mit Karten, der Parameter `auswahl` eine Liste mit Zahlen, die mindestens drei Einträge hat. Die Funktionen sollen einen booleschen Wert (also `True` oder `False`) zurückliefern[3].

Zeigen Sie im Programm nun die Karten des dritten Spielers und die oberste Karte des Ablagestapels an, lassen entscheiden, ob vom Zug- oder Ablagestapel eine neue Karte gezogen werden soll und ziehen diese (s. Teil 3).

Zeigen Sie die Karten des dritten Spielers mit Nummern versehen an und fragen Sie ab, ob und - wenn ja - welche Karten ausgelegt werden sollen. Prüfen Sie, ob es sich dabei um einen Satz oder eine Sequenz handelt und geben Sie das Ergebnis der Prüfung aus.

Bei positiver Prüfung legen Sie die Karten aus (d.h. löschen Sie sie aus der Hand) und wiederholen Sie die Abfrage, ob ausgelegt werden soll. Lassen Sie am Ende eine Karte ablegen.

```
Aufgabe 1, Teil 4:
Oben auf dem Ablagestapel: (“Pik”, “8”)
Hand 3: [(“Pik”, “Ass”), …
Karte vom Ablage- oder Zugstapel nehmen (A/Z)? Z
Hand 3:
Karte 1: (“Pik”, “Ass”)
Karte 2: (“Karo”, “Ass”)
Karte 3: …
Soll ausgelegt werden (j/n)? j
Welche Karte auslegen (1-14, Ende zum Beenden der Auswahl)? 1
Welche Karte auslegen (1-14, Ende zum Beenden der Auswahl)? 2
Welche Karte auslegen (1-14, Ende zum Beenden der Auswahl)? 3
Welche Karte auslegen (1-14, Ende zum Beenden der Auswahl)? Ende
Ausgewählt: [1, 2, 3]
Die Karten bilden einen Satz.
Die Karten bilden keine Sequenz.
Ausgelegt wurden [(“Pik”, “Ass”), (“Karo”, “Ass”), (“Herz”, “Ass”)]
Hand 3:
Karte 1: (“Karo”, “5”)
Karte 2: (“Kreuz”, “3”)
Karte 3: …
Soll ausgelegt werden (j/n)? n
Welche Karte ablegen (1-11)? 1
Ablagestapel: [(“Herz”, “Ass”), (“Pik”, “8”), (“Karo”, “5”)]
Hand 3: [(“Kreuz”, “3”), …
```

[3] Testen Sie das korrekte Verhalten der Funktionen, indem Sie die Funktionen zunächst mit konstruierten Beispieldaten aufrufen.

Aufgabe 2 (40 Punkte):

Schreiben Sie ein Programm, das ein simples "Einzelspieler-Rommé" simuliert. Der Spieler erhält in jeder Runde eine neue Karte vom Zugstapel, er kann - soweit möglich - Sätze oder Sequenzen auslegen und muss eine Karte auf den Ablagestapel legen.

Die benötigten Funktionen haben Sie in Teil 1 zum Großteil schon erstellt. Natürlich dürfen nur gültige Sequenzen und Sätze ausgelegt werden. Zusätzlich sollten Sie für diese Aufgabe die ausgelegten Karten speichern, um diese ebenfalls jede Runde am Bildschirm anzeigen zu können. Konkret bietet sich dazu eine Liste an, in der wiederum Listen von Karten gespeichert werden.

Weiterhin soll ausgegeben werden, welchen Punktwert der ausgelegte Satz bzw. die ausgelegte Sequenz hat. Schreiben Sie dazu am besten eine generische Funktion `zaehle_punkte(liste)`, die für alle in der Liste enthaltenen Karten die Punkte addiert. Achten Sie dabei auf die Besonderheit, dass ein Ass in einem Satz Asse 11 Punkte, als Ende einer Sequenz (Dame - König - Ass) ebenfalls 11 Punkte, aber als Anfang einer Sequenz (Ass - 2 - 3) 1 Punkt zählt[4].

```
Aufgabe 2:
Bereits ausgelegt: []
Es wurde gezogen: (“Herz”, “8”)
Hand:
Karte 1: (“Pik”, “Ass”)
Karte 2: (“Karo”, “Ass”)
Karte 3: (“Herz”, “Ass”)
Karte 4: (“Karo”, “5”)
…
Soll ausgelegt werden (j/n)? j
Welche Karte auslegen (1-14, Ende zum Beenden der Auswahl)? 1
Welche Karte auslegen (1-14, Ende zum Beenden der Auswahl)? 2
Welche Karte auslegen (1-14, Ende zum Beenden der Auswahl)? 3
Welche Karte auslegen (1-14, Ende zum Beenden der Auswahl)? Ende
Ausgewählt: [1, 2, 3]
Die Karten bilden einen Satz.
Die Karten bilden keine Sequenz.
Ausgelegt wurden mit 33 Punkten:
[(“Pik”, “Ass”),  (“Karo”, “Ass”), (“Herz”, “Ass”)]
Hand:
Karte 1: (“Karo”, “5”)
Karte 2: (“Kreuz”, “3”)
Karte 3: …
Soll ausgelegt werden (j/n)? n
Welche Karte ablegen (1-11)? 1
```

(hier geht es dann wieder von vorne los)

[4] https://de.wikipedia.org/wiki/Romm%C3%A9#Die_Kartenwerte

Upload in Moodle

Laden Sie Ihre beiden Skript-Dateien als aufgabe-1.py und aufgabe-2.py in Moodle hoch.

Dies ist eine **Einzelabgabe** - bitte erarbeiten Sie Ihre individuelle Lösung. Abgaben, die sich zu sehr gleichen, können als Täuschungsversuch gewertet werden.