**Blatt 05: Quartett**

1. **Analyse**

Das Programm wird gemäß des EVA-Entwicklungsmusters (Eingabe – Verarbeitung -Ausgabe) entworfen.

Das Programm wird in der Programmiersprache Python 3.9 verfasst.

Die Ein- und Ausgabe erfolgt über die Konsole.

Spezifikation:

* Implementierung eines Quartett Kartenspiels
* 32 Karten mit 8 x Quartetten (4 Karten)
* 2-8 Spieler
* bei 2 Spielern: jeder Spieler erhält 10 Karten, Stapel mit 12 Karten bleibt verdeckt
* bei 3+ Spielern: jeder Spieler erhält gleich viele Karten (wie möglich?), kein Stapel
* Spielablauf:
  + Überprüfen der Karten
  + Ablegen aller Quartette
  + Reihenfolge: Uhrzeigersinn
  + aktiver Spieler fragt beliebigen Spieler nach einer bestimmten Karte
  + wenn der Gefragte die Karte hat, gibt er sie ihm + Vorgang wiederholt sich
  + wenn der Gefragt keine Karte hat, zieht Fragender vom Stapel bzw. falls kein Stapel, dann ist nächster Spieler an der Reihe
  + bei Quartettbildung wird dieses abgelegt
  + Ende: wenn ein Spieler keine Karten mehr hat
  + Ziel: die meisten Quartette bilden

Ideen:

* Kartenset durch ein Dictionary implementieren
* Spielerprofile anlegen durch Funktion players().
  + min. 2, max. 8 Spieler durch if-else-Bedingung
  + Eingabe des Spielernamens
  + soll Spieler durch Computer oder Spieler gespielt werden
* Funktion player() implementiert Spielzug eines Spielers
  + Parameter a: allg. Kartenstapel
  + Parameter b: Kartenset eines Gegenspielers
  + nach Karte fragen können
  + Karten abgeben und aufnehmen können
  + return: eigenen Kartenstapel
* Funktion computer() implementiert Spielzug eines Computers
  + automatisierter Ablauf der Funktion player
* Spielimplementierung durch eigene Funktion game()
  + random.randint wählt zufälligen Spieler der beginnt
  + in for-Schleife (jeder Spieler ist ein Element) wird jeweils Funktion player oder computer ausgeführt
  + Aktualisierung des allg. Kartenstapels
  + Aktualisierung jedes Kartenstapels bei jedem Zug
  + es muss immer klar sein, wie viele Karten noch im Spiel sind und welche abgelegt wurden
* bei jeder Eingabe Neustart und Beenden ermöglichen
* robuste Eingabe durch while-Schleifen garantieren 🡪 Funktion erstellen

Algorithmus:

Algorithmus zum Spielen eines Quartettkartenspiels mit 32 Karten und 2-8 Spielern.

Ausgabe:

Gewinner des Kartenspiels

1. **Coding**

siehe Code:

1. **Test**

Positivtest = +

Negativtest = –

Funktion: rules()

keine Testfälle benötigt, da kein aktives Eingreifen durch den User geschieht

Funktion: players()

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | + – | Klasse | Eingabe | Soll-Wert | Ist-Wert | Kommentar |
| 1 | + | korrekte Eingabe | Computer or Human: 0 | Funktion soll weiterlaufen | Funktion läuft weiter |  |
| 2 | + | korrekte Eingabe | Another round: 0 | Funktion soll enden | Funktion endet  und returned Liste |  |
| 3 | - | falsche Eingabe | Computer or Human: C | erneute Abfrage soll erfolgen | erneute Abfrage erfolgt |  |
| 4 | - | falsche Eingabe | Another round: ja | erneute Abfrage soll erfolgen | erneute Abfrage erfolgt |  |
| 5 | - | 1 Spielerprofil | Funktion beenden nach einem Spielerprofil | nicht möglich | nicht möglich | durch if-Bedingung kein Enden der Funktion möglich |
| 6 | + | 4 Spielerprofile | Funktion beenden nach 4 Spielerprofilen | Beenden möglich | Beenden möglich | zwischen 2 und 8 Profilen ist selbstständigen Fortführen oder Beenden möglich |
| 7 | - | 9 Spielerprofile | Funktion weiterführen nach 8 Spielerprofilen | nicht möglich | automatischer Abbruch, somit nicht möglich | mehr als 8 Spielerprofile nicht möglich, da begrenz durch if-Bedingung |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | + – | Klasse | Eingabe | Soll-Wert | Ist-Wert | Kommentar |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | + – | Klasse | Eingabe | Soll-Wert | Ist-Wert | Kommentar |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | + – | Klasse | Eingabe | Soll-Wert | Ist-Wert | Kommentar |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |

1. **Dokumentation**

Das Programm ist in der Programmiersprache Python 3.9 verfasst.

Zur Ausführung des Programms ist ein Python-Interpreter 3.9 (oder neuer) mit Bibliotheken und Support-Programmen der Standard Installation Pythons erforderlich.

Bedienungsanleitung:

Implementierung der Spezifikation:

Fehler und Bugs: