TECHNISCHE
UNIVERSITÄT
WIEN
Vienna University of Technology

Aufgaben Übung 12

16. Jänner 2013

Übungsbeispiel

Das heutige Übungsbeispiel soll ein einfaches Kartenspiel simulieren. Dieses Spiel soll an zwei bis vier Spieler (muss zu Beginn festgelegt werden) ein Kartenspiel mit 52 Karten austeilen (2-10, B, D, K, A).
Um zu spielen legen die Spieler jeweils die oberste Karte ihres Hand-Stapels auf den Tisch, die höchste Karte (ohne Rücksicht auf die Farbe) gewinnt, die Karten des Tisches gehen an den Besitzer dieser Karte.
Falls gleich hohe Karten zweier oder mehrerer Spieler den Rundensieg beanspruchen, werden zwei weitere Runden Karten darüber gelegt. Die letzte entscheidet dann die Runde (auch bisher unterlegene Mitspieler können noch gewinnen).

Die Aufgabe lässt sich bei näherer Betrachtung in einige Teilbereiche zerlegen, die in weiterer Folge beschrieben werden.

Die Grundklassen: Karten

- eine notwendige Klasse ist die Klasse Karte. Sie muss die Farbe, den Wert, und eventuell auch einen Zahlenwert für die Wertigkeit der Karte (damit z.B. K>B festgelegt wird) enthalten.
- für Vergleiche braucht man auch zwei Vergleichsoperatoren für Karten: < und ==
- Kartenstapel Die Menge aller im Spiel zur Verfügung stehenden Karten.
  - Konstruktor -- Karten im Stapel erstellen (sortiert)
  - Mischen (siehe am Ende...)
  - Teilen: gib eine Karte aus (z.B. als Rückgabewert), die dabei vom Stapel entfernt wird
  - Sammeln: füge eine Karte an den Stapel an. (zum Einsammeln der Karten nach dem Spiel)
  - zum Testen: Ausgabe des Stapels und der Zahl der Karten im Stapel

Die Grundklassen: SpielerIn

- weiters benötigt man die SpielerIn:
  - Name: Bezeichnung für die Ausgabe
  - Hand: Liste zum Halten der Karten
  - Tisch: Liste mit den auf den Tisch gelegten Karten
  - `Karte ausspiel(bool display);` : Karte von der Hand auf den Tisch, zum Debuggen mit Ausgabe der Karte
  - `Karte ausHand(bool display);` : Karte aus der Hand (zum Zurückgeben an den Kartenstapel)
  - `aufnehmen(Karte dazu);` : die Karte hinten an die Hand anfügen (damit die/der SiegerIn die Karten vom Tisch aufbehmen kann)
  - `zeigeHand();` Zeige alle Karten der Hand
  - `zeigeTisch();` Zeige alle Karten des Tisches
  - `int Ergebnis();` Gib die Gesamtzahl an Karten in der Hand zurück

Die Klasse Spiel

Die Klasse Spiel versammelt nun N SpielerInnen, den Kartenstapel und einen Zettel, der die Ergebnissummen der SpielerInnen enthält (ein Feld von N integer-Zahlen). Dazu eine Anzahl Methoden, die dann das Spiel durchführen.

- Konstruktoren (die vorgewählte Zahl Spieler, Defaultwert: 2 Konstruktor Kartenstapel, mischen)
- Karten austeilen (Stapel gibt die Karte, SpielerIn nimmt)
- Spielrunde (siehe nächste Seite)
- Alle Tische werden von SpielerInnen abgegeben und an die/den SiegerIn angehängt
- Ergebnis nach Spielende notieren (Zahl in der Hand zu den Ergebnissen am Zettel dazuzählen)
- Summen vom Zettel ausgeben
- reset(), das nach dem Spiel die Karten einsammelt, mischt und neu verteilt.

Spielrunde

- SpielerIn gibt die oberste Karte in den Tisch
- aus den rückgegebenen Kartenwerten wird der/die Siegerin ermittelt (auswerten)
- bei Gleichstand:
  - eine Runde in den Tisch, nicht auswerten
  - noch eine Runde in den Tisch, auswerten

Das Hauptprogramm

- Hier wird die Klasse Spiel mit $n \in \{2, 3, 4\}$ Spielern erstellt.
- einzelne Spielrunden werden gestartet
- Das Spiel endet, wenn die vorgegebene Zahl von Runden absolviert ist, bzw. wenn nur mehr ein/e SpielerIn Karten hat.
- Ausgabe des Ergebnisses.
- Durchführung mehrerer Spielrunden, Gesamtergebnis

Mischen:

Im Konstruktor des Kartenstapels wird auch ein Zufallszahlengenerator initialisiert.

```
#include <sys/time.h>
#include <cstdlib>
 ...
struct timeval tv;
gettimeofday(&tv, NULL);
srand(tv.tv_sec * tv.tv_usec);
```

Benötigt man danach eine ganze Zufallszahl $0 < i < 53$, so kann man diese aus

```
i = 1. + 52.*(rand()/(RAND_MAX+1.));
```

Mischen erfolgt dann am einfachsten, indem man der Reihe nach jede Karte mit einer zufällig aus dem Stapel gewählten austauscht. Einmal für jede Karte sollte reichen ...