zh-2023-06-06-gy-14h

Határidő Nincs megadva határidő **Pont** 40 **Kérdések** 1
Elérhető 2023. jún 6, 14:45 - 2023. jún 6, 17:50 körülbelül 3 óra **Időkorlát** Nincs
Engedélyezett próbálkozások Korlátlan

Ez a kvíz már nem érhető el, mivel a kurzus befejeződött.

Próbálkozások naplója

	Próbálkozás	Idő	Eredmény
LEGUTOLSÓ	1. próbálkozás	116 perc	0 az összesen elérhető 40 pontból *

* Néhány kérdés még nem lett értékelve

Ezen próbálkozás eredménye: **0** az összesen elérhető 40 pontból *
Beadva ekkor: 2023. jún 6, 16:43
Ez a próbálkozás ennyi időt vett igénybe: 116 perc

1. kérdés **Még nincs értékelve / 40 pont**

Programozási nyelvek Java ZH, 2023.06.06. 14:00

Feltételek

- A feladat megoldását önállóan, más segítsége nélkül kell elkészíteni.
  - Kommunikáció csak az oktatókkal megengedett.
  - Az elkészített megoldást nemcsak a ZH végéig, hanem egészen a ZH napjának végéig nem szabad megosztani mással (pl. fórumba vagy publikus verziókezelő rendszerbe felöltés).
  - A megoldás elkészítéséhez használható a Java API és a JUnit dokumentációja. Ez a Canvasből letölthető, kicsomagolható.
- Az elkészített megoldást **zip** formátumba csomagolva kell feltölteni a Canvasbe.
  - A `zip` tartalmazza a forrásfájlokat. A `jar` fájlok ne kerüljenek bele.

- A **ZH végén kb. 10 percet érdemes fenntartani** a kód tisztázására, fordíthatóvá tételére, tömörítésére, beküldésére.

Alapfeladat

Az UNO kártyajáték egy változatát készítjük el. A program szerkezete feleljen meg a **strukturális tesztelőben** (`UnoTestSuite` és hivatkozott osztályai) leírtaknak.

A játék során feltételezhető, hogy minden adat/bemenet helyes.

A kártyáknak (`Card`) adott a színe, a típusa és az értéke. Ez utóbbi csak akkor számít, ha a kártya típusa `VALUE`, különben feltételezhető, hogy `0`-ra van beállítva; az értékkártyák számai `1-9` közé esnek.

- A konstruktor egyszerűen átveszi ezeket az értékeket.
- Egy kártyát így kell kiírni, ha szám jellegű: `GREEN 9`, és így egyébként: `RED REVERSE`, `YELLOW SKIP`
- Két kártya akkor minősül egyenlőnek, ha mindegyik adattagjuk egyezik.
  - Tipp: ennek a műveletnek a társát is meg kell írni, amihez jól használható az `Objects.hash` metódus.
- Egy kártya akkor játszható ki egy másik kártyára (`isPlayableOver`), ha...
  - speciális (nem-érték) kártya csak megfelelő színű értékkártyára játszható ki.
  - értékkártya kijátszható egyező szín vagy egyező szám esetén.

Egy játékos (`Player`) eltárolja a nevét, a kártyáit sorban (`hand`) és a játszott játék referenciáját.

- Egy játékost így kell kiírni: `Player P2: 1=GREEN REVERSE 2=RED 4 3*=BLUE 5 4=RED 5 5=RED SKIP 6*=GREEN 3`
  - Itt a csillag azt jelöli, hogy a kártya kijátszható a játék aktív kártyájára.

A játék úgy működik, hogy a játékosok sorban megnevezik, melyik kártyájukat játsszák ki. Ha a kártya nem játszható ki, akkor húznak egyet a pakliból. Annak vezérléséről, hogy melyik kártyát választjuk, az `InputSource` osztály gondoskodik.

- A konstruktora megkapja, hogy interaktív módban működik-e.
  - Ha igen, akkor a sztenderd bemenetről olvas majd: `br = new BufferedReader(new InputStreamReader(System.in))`
  - Ha nem, akkor megkapja sorban a választott kártyák sorszámát.
- A `getNextInput` hívás kiadja a következő kártya sorszámát.
  - Interaktív módban beolvasunk `br` segítségével egy szöveget.
    - Ha a `done` szöveget kaptuk, akkor a `DONE` konstanst adja ki.
    - Különben feltételezhető, hogy egész szám jött, és ezt adja ki.

    - Különben a megkapott értékeket adja vissza a metódus sorban.
        - Ha ezek elfogytak, akkor a `DONE` konstanst adja ki.

A játék

A játék megkezdésekor (`Game` osztály főprogramja) példányosítjuk a `Game` osztályt: játékosonként 6 kártya, interaktív `InputSource`, a játékosok nevei pedig a parancssori paraméterek.

- Ha nincsen legalább két játékos, a konstruktor váltson ki `NotEnoughPlayersException` kivételt.
    - A szülő osztály felé ez adjon át ilyen szöveget: `Only 1 players were given`
- A konstruktor első paramétere megadja, kezdetben hány kártyát kapnak a játékosok.
- A konstruktor állítson elő új paklit (`initDeck`).
    - Minden színre: minden számkártyából (`1-9` értékekkel) legyen egy darab.
    - Minden színre: minden speciális kártyából legyen két darab.
    - Keverjük meg a paklit: `Collections.shuffle(deck)`.
    - Ha nem interaktív a játék, úgy keverjük meg a paklit, hogy az mindig ugyanúgy álljon be: `Collections.shuffle(deck, new java.util.Random(12345))`.
- A konstruktor állítsa be a játékosokat (`initPlayers`).
    - A játékos kártyáit a `drawCards` metódussal kell felhúzni a pakliból.
- Az aktív kártya (`currentCard`) legyen felcsapva a pakliból.
    - Ehhez is a `drawCards` metódust kell használni.
- A `currentPlayerIdx`, `isForward`, `isOn` változók álljanak be értelemszerűen.

A játék addig zajlik, amíg `isOn` igaz. Egy lépést a `playNext` hívással teszünk meg. Ez a következőképpen működik.

1. Meghatározza, ki a következő játékos (`getNextPlayerIdx`).
    - Figyelembe kell venni, merre megy a kör, és hogy éppen "átlépünk-e a lista szélén".
    - Az `interactiveMsg` metódussal íródjon ki erről is, és minden másik fontos lépésről is információ.
        - Itt például írjuk ki a játékost (a kártyáival együtt), hogy a felhasználó eldönthesse, melyik kártyát választja.
        - Ez a metódus nem-interaktív módban nem ír ki semmit.
2. Az új játékos lesz az aktív (`currentPlayerIdx`).
3. Az `InputSource` használatával előáll a választott kártya sorszáma. a. Ha ez nem játszható ki, a játékos egy lapot húz fel a pakliból (`currentPlayerDrawCard`). a. Ha kijátszható, kikerül a játékos kezéből, és ez

lesz az aktív kártya. Továbbá, ha van speciális hatása, az kiváltódik (`useCardEffect`).
4. Ha a játékos kezében nem maradt kártya, ő nyert; `isOn` hamisra áll.

A kártyák speciális hatásai típus szerint:

`REVERSE`: a sorrend megfordul `SKIP`: a következő játékos kimarad `TAKE`: a következő játékos kimarad, és húz is egy kártyát

Nem-interaktív tesztelés

Az `uno.GameTest` osztályba kerüljön bele egy tesztelő metódus. Ez hozzon létre egy nem-interaktív `InputSource` által meghajtott játékot három játékossal, fejenként 2 kártyával, `1 0 1` lépésekkel. A következőket próbáljuk ki.

1. Kezdetben az aktív kártya `RED TAKE`.
2. Kezdetben a három játékos kártyái: `GREEN SKIP, RED 1`, `RED 3, BLUE 8`, `BLUE REVERSE, RED 8`
3. Háromszor hívjuk meg a `playNext` metódust, és mindegyik lépés után vizsgáljuk meg a következőt.
   - Az aktív játékos indexe megfelelő.
   - Az aktív kártya valóban a legutóbb kijátszott.
   - Az aktív játékos kártyái közül kikerült a kijátszott kártya.
   - A játék `isOn` értéke igaz.
4. Ha még egyszer meghívjuk a `playNext` metódust, `isOn` hamisra áll.

↓ **javazh.zip (https://canvas.elte.hu/files/2311727/download)**

Kvízeredmény: **0** az összesen elérhető 40 pontból