Házi Feladat

Határidő:

2025.11.28. 23:59:59

Abstract

A feladat célja egy **Blackjack** játék modellezése objektumorientált programozással (OOP), amely megfelel a **SOLID** és **KISS** elveknek. A játékot a következő alapelvek szerint kell megvalósítani:

Feladatleírás:

Készíts egy **Blackjack** játékot, amelyben a játékos és a dealer (osztó) játszanak. Az osztó és a játékos lapokat kap, és a cél, hogy a kezük értéke 21-hez minél közelebb legyen, anélkül hogy meghaladná azt. A kártyák értéke: 2–10 a számértékük szerint, J, Q, K mind 10 pontot ér, és az A kártya 1 vagy 11 pontot érhet.

Alapkövetelmények:

1. **OOP és SOLID elvek:**
    - **Single Responsibility Principle (SRP)**: Minden osztály csak egyféle felelősséggel rendelkezzen. Például a `Deck` osztály feleljen meg a kártyák kezeléséért, míg a `Player` osztály a játékos (vagy dealer) kezeléséért.
    - **Open/Closed Principle (OCP)**: Az osztályoknak úgy kell működniük, hogy bővíthetőek legyenek új játékszabályokkal vagy játékmenettel, de ne szükséges módosítani őket.
    - **Liskov Substitution Principle (LSP)**: Az alosztályoknak helyettesíteniük kell a szülőosztályokat a helyes működés biztosításához.
    - **Interface Segregation Principle (ISP)**: Az osztályoknak csak a szükséges interfészeket kell implementálniuk.
    - **Dependency Inversion Principle (DIP)**: Magas szintű modulok (pl. a játék logikája) ne függjenek alacsony szintű implementációktól (pl. lapkezelés), inkább mindkettő egy közös absztrakciótól függjön.

2. **KISS elv:**
    - A kód legyen egyszerű és könnyen érthető. A játék mechanikáját ne bonyolítsd túl.
    -

Osztályok és funkciók:

1. **Card**: Reprezentálja a kártyát, tartalmazza a kártya értékét és típusát (szín, szám).

2. **Deck**: A kártyapakli, amely kártyákat tartalmaz, képes keverni és osztani kártyákat.

3. **Player**: Reprezentálja a játékost, beleértve az osztókat is. Kezelni kell a játékos kezét és a kártyák összesített értékét.

4. **Game**: A játék logikáját irányító osztály, amely kezeli a játékmenetet, az osztást, és meghatározza a győztest.

5. **BlackjackGame**: Fő játékosztály, amely az összes szükséges interakciót kezeli (felhasználói input, játékmenet folytatása, végkimenetel).

Követelmények:

- Az osztó és a játékos egyszerre kap 2-2 lapot.
- A játékos választhat, hogy kér-e új lapot ("hit") vagy marad (stand).
- A játék akkor ér véget, amikor a játékos vagy az osztó 21 pontot ér el, vagy túllépi azt.
- Az osztó a játék végén is végrehajtja a saját lépéseit (pl. "hit" amíg 17 pont alatt van).
- A győztes az, aki a 21 pontot legjobban megközelíti anélkül, hogy meghaladná azt.

Bónusz funkciók:

- A játék több játékosra is kiterjeszthető.
- A kártya tulajdonságai (például szín) külön kezelésre kerülhetnek.
- A játék végén tájékoztató üzenet a nyertest illetően.

Gyakorló feladatok

1. **JSON: Random személyek**

Generálj 10 személyt, ahol a nevek tartalmaznak véletlenszerű kezdőbetűt és számot, az életkorok pedig egy megadott tartományból származnak. Mentsd el JSON fájlba.

Az előző JSON fájlból olvasd be az adatokat, és szűrd ki azokat, akik 30 év alattiak, majd mentsd új fájlba.

Egészítsd ki a JSON adatokat véletlenszerű email-címek generálásával. Az email legyen a névből és egy véletlen domainből generálva.

---

2. **CSV: Városok adatai**

Adj hozzá egy "kontinens" oszlopot, ahol a kontinens véletlenszerűen van kiválasztva az "Európa", "Ázsia", "Amerika" opciók közül.

Készíts egy új CSV fájlt, amelyben csak azok a városok szerepelnek, ahol a lakosság meghaladja az 500,000 főt.

Írj egy programot, amely számolja és kiírja az átlagos városméretet (terület) az eredeti CSV fájl alapján.

---

3. **TXT: Véletlenszerű dátumok**

Generálj véletlenszerű időpontokat (óra és perc) a következő 24 órán belül, és írd őket egy TXT fájlba.

Olvass be egy TXT fájlt dátumokkal, és ellenőrizd, hogy vannak-e duplikált dátumok. Az egyedi dátumokat írd egy új fájlba.

Generálj dátumokat és hozzájuk tartozó szöveges megjegyzéseket (pl. eseményneveket) ugyanabba a TXT fájlba.

---

4. JSON: Eseménynaptár

Egészítsd ki az eseménynaptár JSON-t véletlenszerű eseménytípusokkal (pl. "meeting", "party", "deadline"), és mentsd új fájlba.

Olvasd be az eseménynaptárat, és számold meg, hány esemény van hétvégén (szombaton vagy vasárnap).

Generálj egy új JSON fájlt, amely az eseményekből külön "munka" és "szabadidő" kategóriákat készít, a dátumok alapján csoportosítva.

---

5. CSV: Átlaghőmérséklet

Írd ki, melyik hónapban volt a legnagyobb hőmérsékletingadozás (maximum - minimum).

Generálj egy új CSV-t, amely minden hónaphoz véletlenszerű időjárási eseményeket ad (pl. "eső", "hó", "száraz").

Számítsd ki, hogy az első 6 hónap vagy az utolsó 6 hónap volt melegebb, és az eredményt írd ki egy új CSV fájlba.

---

6. TXT: Naplóbejegyzések

Egészítsd ki a naplóbejegyzéseket időbélyegekkel (pl. "2024-11-25 14:32"), és írd őket egy új TXT fájlba.

Írj egy programot, amely megszámolja, hányszor szerepel egy bizonyos szó vagy kifejezés a naplófájlban, és az eredményt írd ki egy új fájlba.

Hozz létre egy új fájlt, amely csak azokat a sorokat tartalmazza, amelyek dátuma egy adott intervallumba esik.

---

7. JSON: Véletlenszerű időpontok

Generálj véletlenszerű időpontokat (óra és perc) a következő 3 napon belül, minden napra 3 eseményt, és mentsd JSON fájlba.

Olvasd be a JSON fájlt, és válaszd ki azokat az időpontokat, amelyek este 6 után vannak, majd mentsd új fájlba.

Készíts egy statisztikát, amely számolja, melyik napra esik a legtöbb esemény, és az eredményt írd ki JSON-ba.

---

8. CSV: Véletlenszerű dátumok az év elejéből

Egészítsd ki a CSV fájlt az év minden hónapjának átlagos hőmérsékletével, véletlenszerű értékeket használva.

Írj egy programot, amely összeszámolja, melyik napból van a legtöbb előfordulás a fájlban (pl. hétfőből vagy keddből).

Hozz létre egy új CSV fájlt, amely a hét napjaira csoportosítja a dátumokat, és minden naphoz hozzáad egy véletlenszerű esemény nevét.

---

9. TXT: Véletlenszerű nevek és születési dátumok

Olvass be neveket egy TXT fájlból, és rendelj hozzájuk véletlenszerű születési dátumokat, majd írd vissza őket egy új fájlba.

Készíts egy programot, amely a neveket és születési dátumokat átrendezi név szerinti ábécésorrendbe.

Egészítsd ki a TXT fájlt azzal, hogy minden személyhez hozzárendelsz egy véletlenszerű lakóhelyet egy előre definiált városlistából.

---

10. JSON: Termékek és átlagár

Számold ki, hány termék ára magasabb az átlagos árnál, és az eredményt mentsd egy új JSON fájlba.

Írd ki egy új JSON fájlba azokat a termékeket, amelyek ára egy bizonyos tartományban van (pl. 100 és 500 között).

Egészítsd ki a JSON fájlt minden terméknél egy "kedvezményes ár" mezővel, amely az eredeti ár véletlenszerű csökkentésével jön létre (pl. 5-20%).