Feladat – NagyZH – FONTOS INFÓK

- Alkalmazni kell a megoldás során a tanult objektum-orientáltsági elveket.
- A bemeneti fájlokat (**purchases.json**, **coupons-1.json**, **coupons-2.json**) teszteléskor a build könyvtárba kell másolni.
- Feltehetjük, hogy a bemenő/teszt adatok helyesek, érvényesek (kivéve, ha a feladat mást nem állít).
- A megoldás teljes forráskódját egyetlen ZIP fájlba csomagolva kell feltölteni.

A feladat leírása

A *main*-ben és ebben a leírásban szereplő osztályok és metódusok igény szerint **átnevezhetők**, az itteni nevek csak példák.

A feladatban vásárlásokra alkalmazható kuponok kezelése a cél. Minden kuponnak van egy értéke, és amikor használjuk, akkor ezzel az értékkel csökkenti egy vásárlás végösszegét. Ezen belül többfajta kupon is létezik: az egyszeri kupon csak egyszer használható, utána érvénytelen lesz. A többszöri kuponhoz tartozik egy darabszám, ahányszor használható, és ennyi darab használat után lesz érvénytelen. A havi kupon pedig dátum szerint, egy adott hónapban használható. Az aktuális dátum a programban legyen 2024-05-01, de ez legyen később átállítható.

Az ősosztályt az alapkód **Coupon**-nak hívja. A **Wallet** osztály kuponokat tudjon tárolni tetszőleges mennyiségben. A *main* függvényben a megfelelő helyen tölts fel egy **Wallet**-et egy 1000 Ft-os egyszeri kuponnal, egy 5-ször használható 2000 Ft-os többszöri kuponnal, és egy 2024 májusában érvényes 3000 Ft-os havi kuponnal.

Legyen egy **Purchase** osztály, ami egy vásárlás adatait tárolja. Konstruktorban JSON fájl nevét kapja meg, és innen olvassa be az adatokat. Minden egyes tétel esetén adott a termék neve, egységára, mennyisége és vételi egysége (ez utóbbi csak a kiírásnál fontos). A mennyiség lehet tört, de az árakat egészre kerekítjük. Legyen *print* metódus, ami kiír minden adatot.

Legyen a **Purchase** osztályban *getTotal* metódus, ami kiírja a vásárlás végösszegét. Legyen egy *useCoupons* metódus is, ami paraméterben egy **Wallet**-et kap. A *useCoupons* szintén visszaadja a végösszeget, de csökkentve a kuponok által meghatározott kedvezményekkel. Feltehetjük, hogy nem megyünk 0 alá. Csak az érvényes kuponokat szabad számolni, illetve a kuponok elhasználódnak a vásárlás során!

Legyen a **Wallet** osztályban egy *loadCoupons* függvény, ami egy JSON fájlból tölt be kuponokat a már meglévők mellé. A fájlban többféle kupon van tárolva, és mindegyik új (még nem használt).

Legyen a **Wallet** osztályban egy *countCoupons* **sablon** függvény, ami visszaadja, hogy hány adott típusú kupon van. Opcionálisan paraméterben legyen megadható, hogy csak az érvényes kuponokat számolja (**true**), vagy mindet (**false**, alapértelmezett).

Legyen a **Wallet** osztályban egy *exportCounts* metódus, ami egy egy fájlnevet kap, ahova JSON formátumban elmenti, hogy a háromféle típusból hány érvényes és érvénytelen darab van (lásd a két példa JSON fájlt).

Pontozás

1. Kuponok adatszerkezete, a használathoz szükséges virtuális függvények **(8 pont)**
2. Kuponok hozzáadása egy **Wallet**-hez a *main*-ben **(2 pont)**
3. **Purchase**: vásárlás tárolása, betöltése **(6 pont)**
4. *print* metódus **(4 pont)**
5. *getTotal* metódus **(4 pont)**
6. *useCoupons* metódus **(4 pont)**
7. A havi kuponokhoz az aktuális dátum globálisan átállítható **(2 pont)**
8. **Wallet**: *loadCoupons* metódus **(4 pont)**
9. *countCoupons* sablon metódus **(3 pont)**
10. *exportCounts* metódus **(3 pont)**

Összesen: **40 pont**.