Programozás 2

– 1. zárthelyi dolgozat –

Czier Norbert

2022-10-12

1. feladat – Játékos (1 pont)

Írjon egy Player nevű osztályt, amivel egy játékost tudunk reprezentálni. A Player osztályt egy `Player.java` állományba helyezze el. Az osztálynak egyetlen konstruktora legyen. A konstruktornak meg kell adni a játékos nevét és típusát(harcos, mágus, felderítő). A Player osztálynak négy attributuma legyen: name, cast, xp, level. A Player objektumok esetén a kezdeti szint 1 és a tapasztalati pont 0. A példányosítás után a name és a cast attributumokat ne lehessen módosítani.

```
Player p1 = new Player("Gamer123", "harcos");
Player p2 = new Player("Hacker1", "mágus");
Player p3 = new Player("Körte", "felderítő");
```

Példányosítás után az példányokat a következőképpen akarjuk használni. A megjegyzésekben az elvárt kimenet látható:

```
System.out.println(p1);                  // Level 1 mágus Gamer123
System.out.println(p1.getName());        // Gamer123
System.out.println(p2.getLevel());       // 1
System.out.println(p2.increaseXp(2));    // az xp-t 2-vel növeljük
System.out.println(p2.getLevel());       // 2
System.out.println(p3.getCast());        // felderítő
System.out.println(p1.compare(p2));      // -1
System.out.println(p1.compare(p3));      // 0
System.out.println(p2.compare(p1));      // 1
```

Magyarázat: Az `increaseXp()` metódus paraméterül kap egy egész számot és annyival növeli az xp-t. Ha az xp nagyobb vagy egyenlő mint az aktuális szint kétszerese, akkor növeljük a játékos szintjét. A `compare()` metódus két játékos szintjét hasonlítja össze. Ha a bal oldali játékos kisebb szintű, akkor -1-et adunk vissza. Ha a két játékos azonos szintű, akkor 0 a visszatérési érték. Ha a bal oldali játékos a nagyobb szintű, akkor +1 -et kell visszaadni.

2. feladat – Csomagolópapír (1 pont)

Az `dobozok.txt` file-ban dobozok méretei szerepelnek. Írjon egy programot, ami kiszámítja a dobozokhoz szükséges csomagolópapír méretét. Ehhez szükséges lesz a doboz felületének a kiszámítására, ami `2*l*w + 2*w*h + 2*h*l`.

Például tegyük fel, hogy az állomány a következőket tartalmazza:

```
2x3x4
1x1x10
```

Egy 2x3x4-es méretű doboz felülete: $2*6 + 2*12 + 2*8 = 52$. *Egy 1x1x10 méretű doboz felülete:* $2*1 + 2*10 + 2*10 = 42$. Ebben az esetben 94 négyzetméter csomagolópapírra van szükség.

Futási példa egy kisebb méretű input fájl esetén:

```
$ cat pelda.txt
2x3x4
1x1x10

$ java Main
94 m^2
```

3. feladat – pangram (1 pont)

Írjon egy programot, ami interaktív módon bekér egy sztringet a felhasználótól, majd írjuk ki, hogy az adott sztring pangram-e? A pangram egy olyan mondat, amely az angol ábécé mind a 26 betűjét tartalmazza. Az egyik legismertebb példa a pangramra a következő: "The quick brown fox jumps over the lazy dog."

A feladat megoldásához használjuk a `StringUtils.isPangram()` nevű metódust, ami eldönti, hogy az adott sztring pangram e.

Futási példa:

```
$ java Main
Szöveg: The quick brown fox jumps over the lazy dog.
A fenti szöveg pangram!

$ java Main
Szöveg: Ez egy roszz pelda.
A fenti szöveg nem pangram!
```

4. feladat – átalakítás (1 pont)

Írjon egy programot, ami parancssori argumentumként kap egy darab bináris számot, majd a képernyőre kiírja a szám decimális értékét. Ellenőrizzük le, hogy csak egy darab bináris szám lett megadva és tájékoztassuk a felhasználót. Feltételezhetjük, hogy érvényes bináris szám lett megadva és a bináris szám maximum 32 bit-ből áll.

Futási példa:

```
$ java Main
Hiba! Nem megfelelő számú argumentumot adott meg!
$ echo $?
1

$ java Main 0011 011
Hiba! Nem megfelelő számú argumentumot adott meg!
$ echo $?
1

$ java Main 011001
25

$ java Main 1111
15
```