JAVA 11

4. forduló

A kategória támogatója: IBM

RENDELKEZÉSRE ÁLLÓ IDŐ:

15:00

Ismertető a feladathoz

Fontos információk

A forduló után a megoldások publikálásával együtt iránymutatásként elérhetőek lesznek a **helyezéssel kapcsolatos információk**, látni fogod, hogy a kategóriában a játékosok 20%, 40% vagy 60%-a közé tartozol-e épp.

Felhívjuk figyelmedet, hogy a következő, **5. fordulótól az egyes kategóriák csak a kijelölt napokon lesznek megoldhatóak 7-22 óra között**, érdemes letöltened a naptárat a Kategóriáim menüpontban.

Felhasznált idő: 02:06/15:00 **Elért pontszám:** 0/50

1. feladat 0/5 pont

Melyek a terminális (terminal) operátorok az alábbiak közül?

Válasz

- allMatch, anyMatch, map, min
- reduce, count, filter, flatMap
- max, collect, count, noneMatch
- skip, peek, distinct, filter

Magyarázat

Az allMatch, anyMatch, noneMatch, min, max, reduce, count, collect terminal, a map, filter, flatMap, skip, peek, distinct pedig intermediate operátor.

2. feladat 0/5 pont

Melyek a köztes (intermediate) operátorok az alábbiak közül?

Válasz

- ○ allMatch, anyMatch, map, min
- ○ reduce, count, filter, flatMap
- ○ max, collect, count, noneMatch
- ◯ skip, peek, distinct, filter

Magyarázat

Az allMatch, anyMatch, noneMatch, min, max, reduce, count, collect terminal, a map, filter, flatMap, skip, peek, distinct pedig intermediate operator.

3. feladat 0/5 pont

Mit csinál az alábbi kódrészlet?

```
Stream.of("A:B", "C:D", "E:F")
  .map(a -> a.split(":"))
  .collect(Collectors.toMap(k -> k[0], v -> v[1]));
```

Válasz

- ○ Egy listából Map-et csinál és visszaad egy streamet
- ◯ Egy listából Map-et csinál és vissza is adja azt
- ○ Egy lista elemeit szétdarabolja és visszaad egy Stringet
- ○ NullPointer exceptiont dob

Magyarázat

A kódrészlet egy listát csinál, amelynek az elemein végrehajtja a szétdarabolást kettőspontokként, majd egy Map-be gyűjti az eredményt.

4. feladat 0/5 pont

Adott egy String input változó. Mi a különbség a következő két kódrészlet között?

1.

```
boolean isWhitespaceOnly = Stream.of(input)
  .map(String::trim)
  .anyMatch(String::isEmpty);
```

2.

```
boolean isWhitespaceOnly = Optional.ofNullable(input)
  .map(String::trim)
  .filter(String::isEmpty)
  .isPresent();
```

Válasz

- Semmi különbség nincs köztük
- A Stream.map() streamje nem adhat vissza objektumot
- Az Optional streamje nem adhat vissza objektumot
- Az Optional streamje nem dobhat NullPointerException-t

Magyarázat

Az Optional streamje úgynevezett null-safe collectiont hoz létre a streamekből, ezért nem dobhat NullPointerException-t.

5. feladat 0/5 pont

Melyik problémát célszerű megoldani a java.util.function.Predicate segítségével?

Válasz

- [ ] Eldöntjük egy Stringről, hogy yyyy-MM-dd formátumú dátumot tartalmaz-e
- [ ] Egy Singleton példányát adjuk vissza
- [ ] Az argumentumként kapott értéket kiírjuk a standard output-ra
- [ ] Egy String-et parse-olunk és dátummá konvertáljuk

Magyarázat

A Predicate egy olyan funkcionális interface, aminek a művelete egy adott típust fogad és boolean-nel tér vissza.

6. feladat 0/5 pont

Melyik problémát célszerű megoldani a java.util.function.Consumer segítségével?

Válasz

- [ ] Eldöntjük egy Stringről, hogy yyyy-MM-dd formátumú dátumot tartalmaz-e
- [ ] Egy Singleton példányát adjuk vissza
- [x] Az argumentumként kapott értéket kiírjuk a standard output-ra
- [ ] Egy String-et parse-olunk és dátummá konvertáljuk

Magyarázat

A Consumer egy olyan funkcionális interface, aminek a művelete egy adott típust fogad és void a visszatérési típusa.

7. feladat 0/5 pont

Melyik problémát célszerű megoldani a java.util.function.Supplier segítségével?

Válasz

- [ ] Eldöntjük egy Stringről, hogy yyyy-MM-dd formátumú dátumot tartalmaz-e
- [x] Egy Singleton példányát adjuk vissza
- [ ] Az argumentumként kapott értéket kiírjuk a standard output-ra
- [ ] Egy String-et parse-olunk és dátummá konvertáljuk

Magyarázat

A Supplier egy olyan funkcionális interface, aminek a művelete nem fogad argumentumot és egy adott típussal tér vissza.

8. feladat 0/5 pont

Melyik problémát célszerű megoldani a java.util.function.Function segítségével?

Válaszok

- [x] Eldöntjük egy Stringről, hogy yyyy-MM-dd formátumú dátumot tartalmaz-e
- [ ] Egy Singleton példányát adjuk vissza
- [ ] Az argumentumként kapott értéket kiírjuk a standard output-ra
- [x] Egy String-et parse-olunk és dátummá konvertáljuk

Magyarázat

A Function egy olyan funkcionális interface, aminek a művelete egy adott típust fogad és egy másikkal tér vissza - ami akár az argumentummal megegyezhet.

9. feladat 0/5 pont

Mire szolgál a Stream.reduce()?

Válasz

- ( ) Megkeresi a legkisebb elemet a listában
- ( ) Visszaadja a stream hosszát
- (•) Egy eredményt csinál a stream elemeiből
- ( ) Csökkenti a stream elemeinek a számát

Magyarázat

A Stream.reduce() lehetővé teszi, hogy egyetlen eredményt állítsunk elő egy stream elemeiből. Ez lehet veszteségmentes (pl. listává konvertálás) vagy veszteséges (pl. legnagyobb elem megkeresése).

10. feladat 0/5 pont

Mi a különbség az Optional.of() és az Optional.ofNullable() között, ha a függvény null értéket kap?

Válasz

- ○ Az Optional.of() csak @NotNullable annotációval ellátott paramétert fogad el.
- ○ Az of() üres Optional példányt ad vissza null érték esetén.
- ○ Nincs különbség köztük.
- ◔ Az ofNullable() üres Optional példányt ad vissza null érték esetén.

Magyarázat

Az Optional.ofNullable() ha null értéket kap, visszaad egy üres Optional példányt, ezáltal nem akasztja meg a program futását null érték esetén, míg az Optional.of() kivételt dob.