Zadaci za vježbu iz teme 7 (Iznimke)

1. Utvrdite probleme koji se nalaze u metodama klase `hr.fer.oop.homework_07.e01.Problem`. Za svaki od problema predložite rješenja i implementirajte popravke po vlastitom nahođenju.

2. Napravite obradu iznimki za metodu `exceptionalMethod(String[] input)` na sljedeće načine:

    a. Ugraditi *try-catch* elemente na najspecifičnijim mjestima u kodu

    b. Ugraditi jedan *try* s više *catch* blokova

    c. Ugraditi jedan *try* s *multi catch* varijantom.

Obrada iznimki mora na standardni izlaz ispisati poruku koju je moguće dohvatiti pozivom `getMessage()` nad iznimkom.

```
private static void exceptionalMethod(String[] input) {
	String result = "";
	for (String string : input) {
		String upper = string.toUpperCase();
		result+=upper.toCharArray()[0];
	}
	System.out.println(result);
}
```

3. Napišite program koji prima točno dva ulazna argumenta te ih ispisuje na standardni izlaz. U protivnom će program izazvati odgovarajuću iznimku.

4. Dodajte `finally` blok u metodu `exceptionalMethod(String[] input)` (drugi zadatak, varijanta c s *multi catchom*). Blok treba ispisati „finally“. Predvidite ispis metode za ulaz `{ "a", "b", "c", null }`.

5. Napišite dvije vlastite iznimke: `UncheckedException` koja je neprovjeravana i `CheckedException` koja je provjeravana iznimka. Napišite metodu `m1` koja baca `UncheckedException` i `m2` koja baca `CheckedException`. Napišite metodu `main` koja poziva metode `m1` i `m2` te ne smije sadržavati *try catch*. Koja je razlika između provjeravane i neprovjeravane iznimke? Može li se provjeravati neprovjeravana iznimka?

6. Utvrdite ispis za sljedeći odsječak metode main:

```
Resource r1 = new Resource(1);
try {
	try (r1; Resource r2 = new Resource(2)) {
		System.out.println("try");
		Integer.parseInt("zero");
	} catch (NumberFormatException e) {
		throw new RuntimeException("wrapped exception", e);
	} finally {
		System.out.println("finally");
	}
} catch (Exception e) {
	System.out.println(e.getMessage());
	System.out.println(e.getCause().getMessage());
} finally {
	System.out.println("finally 2");
}
```

Klasa Resource implementirana je na sljedeći način:

```
public class Resource implements AutoCloseable {
	private int id;
	public Resource(int id) {
		this.id = id;
	}
	@Override
	public void close() {
		System.out.println("R" + id + " is now closed!");
	}
}
```

Rješenja zadataka dostupna su na sljedećoj poveznici:
https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-07

Komentari:

1. 5 metoda ima probleme, u rješenjima su dani dodatni komentari kako se ti problemi mogu riješiti.
2. S obzirom na to kako je u ovom slučaju obrada iznimki identična za sve tipove iznimki, posljednja varijanta je najprikladnija.
3. Odgovarajuća iznimka je `IllegalArgumentException`.
4. Ispis:
   finally
   finally
   finally
   Cannot invoke "String.toUpperCase()" because "string" is null
   finally
   ABC
5. Provjeravana iznimka nasljeđuje `Exception`, neprovjeravana iznimka nasljeđuje `RuntimeException`. Provjeravanu iznimku moramo obraditi dok neprovjeravanu ne moramo ali možemo. S obzirom na to kako zadatak nalaže kako se ne smije koristiti try catch u mainu, koristimo ključnu riječ `throws` u potpisu metode.
6. Ispis:
   try
   R2 is now closed!
   R1 is now closed!
   finally
   wrapped exception
   For input string: "zero"
   finally 2