Zadaci - JUnit 5

1. Dopuniti klasu *CalculatorTest* sa testovima koji proveravaju metode klase *Calculator*. Slučajevi koje je potrebno proveriti za svaku metodu:

- Za metodu sabiranja – *add(int a, int b)* napisati jedan **parametrizovan test** koji proverava:
  - sabiranje bilo koja dva **pozitivna** broja – proveriti komutativnost,
  - sabiranje bilo koja dva **negativna** broja – proveriti komutativnost,
  - sabiranje **pozitivnog** i **negativnog** broja – proveriti komutativnost,
  - sabiranje dve **nule**,
  - sabiranje bilo kog broja **sa nulom** – proveriti za negatine i pozitivne brojeve
- Za metodu oduzimanja – *subtract(int a, int b)* napisati jedan **parametrizovan test** koji proverava:
  - oduzimanje dva **pozitvna** broja – proveriti slučajeve kada je $a > b$ i $a < b$,
  - oduzimanje dva **negativna** broja – proveriti slučajeve kada je $a > b$ i $a < b$,
  - oduzimanje **pozitivnog** i **negativnog** broja – proveriti slučajeve kada je $|a|>|b|$ i $|a|<|b|$,
  - oduzimanje bilo kog broja **sa nulom** – proveriti za negatine i pozitivne brojeve
- Za metodu množenja – *multiply(int a, int b)* – napisati jedan **parametrizovan test** koji proverava:
  - množenje bilo koja dva **pozitivna** broja – proveriti komutativnost,
  - množenje bilo koja dva **negativna** broja – proveriti komutativnost,
  - množenje pozitivnog i **negativnog** broja – proveriti komutativnost,
  - množenje dve **nule**,
  - množenje bilo kog broja **sa nulom** – proveriti za negativne i pozitivne brojeve
- Za metodu deljenja – *divide(int a, int b)* napisati jedan **parametrizovan test** koji proverava:
  - deljenje bilo koja dva **pozitivna** broja – proveriti slučajeve kada je $a > b$ i $a < b$,
  - deljenje bilo koja dva **negativna** broja – proveriti slučajeve kada je $a > b$ i $a < b$,
  - deljenje **pozitivnog** i **negativnog** broja – proveriti slučajeve kada je $|a|>|b|$ i $|a|<|b|$,
  - deljenje bilo kog broja **sa jedinicom** – proveriti za negativne i pozitivne brojeve

  * dodatno za metodu deljenja napisati test koji proverava da li prilikom deljenja sa nulom dolazi do bacanja *ArithmeticException*-a.

Napomene:

- Pre svih testova potrebno je instancirati objekat klase *Calculator* korišćenjem neke od *lifecycle* metoda.
- Za svaki parametrizovani test koristiti drugačiju metodu prosleđivanja vrednosti parametara: *@CsvSource*, *@CsvFileSource*, *@MethodSource* kada se metoda koja obezbeđuje testne podatke nalazi u istoj klasi i *@MethodSource* kada se metoda koja obezbeđuje testne podatke nalazi u drugoj klasi – potrebno navesti punu putanju do metode.
- Postaviti izvršavanje testova tako da se izvršavaju po abecednom redosledu naziva test metoda.

2. Testirati metode *BankAccount* klase koja se nalazi u *bank* paketu. Proveriti uspešne, neuspešne, granične i izuzetne slučajeve (slučajevi kada se baca *IllegalArgumentException* – proveriti da li se dobije odgovarajuća poruka izuzetka). Pre izvršavanja svakog testa, stanje na računu (*balance*) treba biti postavljeno na 100. U testovima u kojima se ne proverava da li je došlo do bacanja *IllegalArgumentException*-a, postaviti da se **preskoči izvršavanje** testa ukoliko se prosledi vrednost koja bi mogla da izazove bacanje ovog izuzetka.

3. Testirati metode *ContactManager* klase koja se nalazi u *contact* paketu. Pre svih testova instancirati objekat *ContactManager* klase i dodati jedan kontakt preko *addContact* metode. Proveriti uspešne, neuspešne (slučaj kada se

dodaje kontakt koji već postoji staviti da se ***ponavlja 3 puta***), granične i izuzetne slučajeve (slučajevi kada se bacaju *AlreadyExistsException* i *NotFoundException*). **Isključiti izvršavanje** testova koji testiraju *removeContact* metodu, jer ova metoda trenutno nije implementirana.

4. Testirati metode *SortingAlgorithms* klase koja se nalazi u *util* paketu. Sve metode su sort algoritmi, tako da je za sve testove moguće koristiti iste testne podatke. Dovoljno je testirati 3 slučaja: sortiranje niza pozitivnih brojeva, sortiranje niza negativnih brojeva i sotiranje niza koji sadrži i pozitivne i negativne brojeve. Postaviti da je dozvoljeno vreme izvršavanja svakog testa 100ms.

5. Napraviti **test suite** kojim se pokreću svi testovi koje ste napisali. Samostalno izaberite strategiju za grupisanje testova. Testove koji proveravaju bacanje izuzetaka svrstati u grupu “exception-handling” i **isključiti** ih iz test suite-a.