Testowanie aplikacji JAVA

Laboratorium (Wstęp do JUnit)

Wstęp do JUnit jest **tutaj**.
Projekt HelloJUnit jest **tutaj**.
Wiki do JUnit jest **tutaj**.

Zadanie 1. Napisz klasę NWD, która będzie zawierała metodę:

```
int nwd(int a, int b)
```

Napisz klasę NWDTest, która będzie klasą testującą dla klasy NWD. Zaplanuj i zaimplementuj odpowiednie przypadki testowe (sprawdzające poprawność wykonywanych operacji). Skorzystaj z różnych asercji. Metoda NWD ma liczyć największy wspólny dzielnik (algorytm Euklidesa).

Zadanie 2. Napisz klasę Calculator, która będzie dostarczała pięć publicznych metod:

```
int add(int a, int b)
int sub(int a, int b)
int multi(int a, int b)
int div(int a, int b)
boolean greater(int a, int b)
```

Napisz klasę CalculatorTest, która będzie klasą testującą dla klasy Calculator. Zaplanuj i zaimplementuj odpowiednie przypadki testowe (sprawdzające poprawność wykonywanych operacji). Skorzystaj z różnych asercji.

Zadanie 3. Rozszerz klasę CalculatorTest o przypadek testowy, który zakończy się błędem, gdy przy próbie dzielenia przez 0 nie wystąpi wyjątek typu ArithmeticException.

Zadanie 4. Napisz klasę Calculator, która będzie działała analogicznie do tej z zadania 1, ale będzie wykonywała operacje na liczbach typu double. Napisz klasę CalculatorTest (zwróć uwagę na możliwe błędy w zaokrągleniach, jak sobie z tym poradzić?).

Zadanie 5. Napisz klasę LiczbaRzymska, która będzie posiadała jedno prywatne pole zawierające liczbę (zainicjalizowane w konstruktorze) i metodę toString(), która będzie zwracała tę samą liczbę zapisaną w rzymskim systemie zapisywania liczb. Przed implementacją metody toString() zastanów się jak powinna zachować się ta klasa w różnych przypadkach np. gdy w konstruktorze podano liczbę ujemną. Zaimplementuj klasę testującą i odpowiednie przypadki testowe. W tym momencie testy oczywiście zakończą się niepowodzeniem. Zaimplementuj metodę toString() i uruchom ponownie testy.