

Uniwersytet im. Adama Mickiewicza w Poznaniu  
Wydział Matematyki i Informatyki



---

# Testowanie białoskrzynkowe i Testy mutacyjne

Raport pokrycia kodu

---

Autor:  
Jakub Przybyła

2 czerwca 2020

# 1. Temat zadania

Poniższe opracowanie tekstowe dotyczy projektu z przedmiotu „Wprowadzenie do testowania”. Zadanie polegało na zwiększeniu pokrycia testami jednostkowymi oraz przeprowadzenia testów mutacyjnych na zadanej aplikacji – kalkulator BMI.

## 2. Informacje ogólne

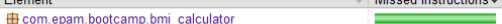

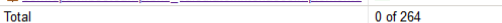
- Link do repozytorium: <https://github.com/JakubPrzybyla/bmi-calculator>
- *originalTests* - branch zawierający oryginalne testy
- *updatedTests* - branch zawierający uzupełnione testy
- *afterMutantsTests* - branch zawierający poprawione testy po przeprowadzeniu testów mutacyjnych

Raport dotyczący pokrycia testami został wytworzony dzięki pluginowi JaCoCo. Do przeprowadzenia testów mutacyjnych użyty został plugin PitTest.

## 3. Raport pokrycia kodu oryginalnymi testami

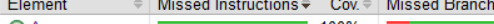



Oryginalne testy pokrywają kod w 92%.

### bmi-calculator

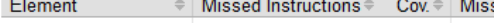








Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods	Missed Classes
com.epam.bootcamp.bmi_calculator		100%		92%	3 31	0 58	0 12	0 2
com.epam.bootcamp.bmi_calculator.interfaces.Implements		100%	n/a	n/a	0 6	0 8	0 6	0 2
Total	0 of 264	100%	3 of 38	92%	3 37	0 66	0 18	0 4

Wynik ten zaniża klasa **App**, a dokładniej jej dwie metody: *calculateBMI()* oraz *bmiResults()*.

### com.epam.bootcamp.bmi\_calculator

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods	Missed Classes
App		100%		83%	3 15	0 28	0 6	0 1
GuessTheUnits		100%		100%	0 16	0 30	0 6	0 1
Total	0 of 234	100%	3 of 38	92%	3 31	0 58	0 12	0 2




### App

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods
bmiResult()		100%		80%	2 6	0 7	0 1
calculateBMI()		100%		75%	1 3	0 11	0 1
ZeroChecker()		100%		100%	0 3	0 5	0 1
setWeight(double)		100%	n/a	n/a	0 1	0 2	0 1
setHeight(double)		100%	n/a	n/a	0 1	0 2	0 1
App()		100%	n/a	n/a	0 1	0 1	0 1
Total	0 of 117	100%	3 of 18	83%	3 15	0 28	0 6

## 4. Raport pokrycia kodu uzupełnionymi testami










Uzupełnione testy pokrywają kod w 97%.

### bmi-calculator

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods	Missed Classes
com.epam.bootcamp.bmi_calculator		100%		97%	1 29	0 58	0 12	0 2
com.epam.bootcamp.bmi_calculator.interfaces.Implements		100%	n/a		0 6	0 8	0 6	0 2
Total	0 of 254	100%	1 of 34	97%	1 35	0 66	0 18	0 4

Poprzez nowe testy udało się pokryć w całości kod metody *bmiResults()*.

### App

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods
calculateBMI()		100%		75%	1 3	0 11	0 1
bmiResult()		100%		100%	0 4	0 7	0 1
ZeroChecker()		100%		100%	0 3	0 5	0 1
setWeight(double)		100%	n/a		0 1	0 2	0 1
setHeight(double)		100%	n/a		0 1	0 2	0 1
App()		100%	n/a		0 1	0 1	0 1
Total	0 of 107	100%	1 of 14	92%	1 13	0 28	0 6

Jedynym, niepokrytym w pełni fragmentem jest metoda *calculateBMI()*.

```
33.     public double calculateBMI() throws Exception{ //végül kiszámoljuk a BMI-t
34.         ZeroChecker();
35.         GuessTheUnits gtu = new GuessTheUnits(this.height, this.weight);
36.         if(gtu.getUnitType().equals("US")){
37.             UsBMI ubmi = new UsBMI();
38.             ubmi.setBMI(gtu.getWeight(), gtu.getHeight());
39.             this.bmi = ubmi.getBMI();
40.         }else if(gtu.getUnitType().equals("metric")){
41.             MetricBMI mbmi = new MetricBMI();
42.             mbmi.setBMI(gtu.getWeight(), gtu.getHeight());
43.             this.bmi = mbmi.getBMI();
44.         }
45.         return this.bmi;
46.     }
```

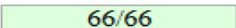
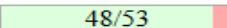
Nieuwzględniony w testach branch jest jednakże nieosiągalny. Logika aplikacji nie pozwala na osiągnięcie typu przyjętych danych, innych niż „US” bądź „metric”.

## 5. Raport z wykonania testów mutacyjnych

Testy mutacyjne pokrywają kod w 91%.

### Pit Test Coverage Report

#### Project Summary

Number of Classes	Line Coverage	Mutation Coverage
4	100% 	91% 

W tym przypadku obie klasy, **App** oraz **GuessTheUnits** zaniżają wynik ogólny.

## Pit Test Coverage Report

### Package Summary

**com.epam.bootcamp.bmi\_calculator**

Number of Classes	Line Coverage	Mutation Coverage
2	100% <div>58/58</div>	89% <div>41/46</div>

### Breakdown by Class

Name	Line Coverage	Mutation Coverage
<a href="#">App.java</a>	100% <div>28/28</div>	85% <div>17/20</div>
<a href="#">GuessTheUnits.java</a>	100% <div>30/30</div>	92% <div>24/26</div>

## 6. Analiza poszczególnych mutantów

Poniższe trzy mutacje zawarte są w jednej metodzie *bmiResults()* klasy **App** oraz są typu „Conditionals Boundary Mutator”. Zmieniają one operatory (<, <=, >, >=), odpowiednio dodając, bądź zabierając równość.

```
52         public String bmiResult(){
53 2         if(this.bmi < 18.5){ // Sovány
54 1             return "Thinness";
55 2         }else if(this.bmi <= 24.9){ // Normál testalkatú
56 1             return "Normal";
57 2         }else if(this.bmi <= 29.9){ // Túlsúlyos
58 1             return "Overweight";
59             }else{ // Erősen túlsúlyos
60 1             return "Heavily overweight";
61         }
62     }
```

Mutacje dotyczą sprawdzania wartości granicznych, sytuacji, gdy wartość zmiennej *bmi* wynosi 18.5, 24.9 albo 29.9. Byłem zaskoczony, że uzupełnione testy nie objęły tych przypadków. Po głębszej analizie dotychczasowych testów znalazłem w nich pewną nieścisłość, przez którą mutanty mogły przeżyć.

Przykładowy test mający w zamyśle testera pokryć ten przypadek wygląda następująco:

```
@Test
public void shouldReturnThatBMIisOverweight() throws Exception{
    app.setHeight(5.9);
    app.setWeight(207);
    assertEquals(app.calculateBMI(), actual: 29.9, delta: 1);
    assertEquals(app.bmiResult(), actual: "Overweight");
}
```

W teorii wszystko się zgadza, występuje asercja obliczonej wartości z liczbą 29.9, test przechodzi. Jednakże tester, aby ułatwić sobie pracę i uniknąć szukania dokładnych parametrów, dla których *bmi* wyniesie tyle ile chce, zastosował zaokrąglenie „delta: 1”.

Dokładna wartość *bmi* w powyższym teście to 29.030810112036765.

Mutant został zneutralizowany poprzez napisanie nowego testu, gdzie zaokrąglenie nie występuje, a wartość *bmi* wynosi dokładnie 29.9

```
@Test
public void shouldReturnThatBMINormalWhenEquals24_9() throws Exception{
    app.setHeight(2);
    app.setWeight(99.6);
    assertEquals(app.calculateBMI(), actual: 24.9, delta: 0.0);
    assertEquals(app.bmiResult(), actual: "Normal");
}
```

Pozostałe dwa mutanty o tej samej genezie zostały zabite w analogiczny sposób.

Innymi mutantami, które przetrwały są typu „Conditionals Boundary Mutator” w metodzie *guessUnit()* klasy **GuessTheUnits**.

```
52
53 1 if(this.unitType.equals("US")){ // ounces
54 2 if(this.weight > 1000){
55 1 convertUnit("ounces"); //átkonvertáljuk fontra
56 }
57 }else{
58 2 if(this.weight > 1000){
59 throw new Exception("Height and weight is in different metric.");
60 }
61 }
62 }
```

Oba dotyczą wartości zmiennej *weight*, w pierwszym przypadku, gdy *unitType* == „US”, a w drugim, gdy równy jest on „metric”.  
Dotychczasowe testy nie obejmowały przypadków, gdy waga wprowadzona w amerykańskim systemie wag jest równa 1000, oraz analogicznej sytuacji w systemie metrycznym.

Mutanty zostały zneutralizowane poniższymi testami:

```
69 //killing mutants
70 @Test
71 public void shouldReturn1000WeightInTypeUS() throws Exception{
72     GuessTheUnits gtu = new GuessTheUnits( height: 10, weight: 1000);
73     assertEquals(gtu.getUnitType(), actual: "US");
74     assertEquals(gtu.getWeight(), actual: 1000, delta: 0);
75 }
76
77 @Test
78 public void shouldReturn1000WeightInTypeMetric() throws Exception {
79     GuessTheUnits gtu = new GuessTheUnits( height: 1.7, weight: 1000);
80     assertEquals(gtu.getUnitType(), actual: "metric");
81     assertEquals(gtu.getWeight(), actual: 1000, delta: 0);
82 }
83 }
```

Po dodaniu odpowiednich testów pokryły one 100% kodu.

## Pit Test Coverage Report

### Project Summary

Number of Classes	Line Coverage	Mutation Coverage
4	100% 66/66	100% 53/53

### Breakdown by Package

Name	Number of Classes	Line Coverage	Mutation Coverage
<a href="#">com.epam.bootcamp.bmi.calculator</a>	2	100% 58/58	100% 46/46
<a href="#">com.epam.bootcamp.bmi.calculator.interfaces.Implements</a>	2	100% 8/8	100% 7/7