D1049174 吳念澤

1.Triangle

2.Person

Github:https://github.com/kokorosawa/FCU_SQA

1.Triangle

設計說明:

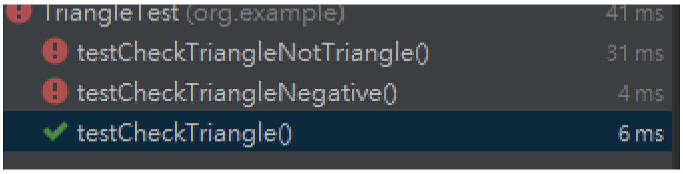
在class Triangle中有一個方法叫做 checkTriangle(), 三個邊長作為參數。首先, 它會檢查這三個邊長是否都大於 0, 如果有任何一個邊長小於 0, 就會拋出 TriangleException 例外, 再來檢查這三個邊長是否符合三角形的定義, 也就是兩邊的和必須大於第三邊。如果不符合, 就會拋出 TriangleException 例外。

最後, 它會根據三個邊長的關係來判斷三角形的種類。如果這三個邊長相等, 就是正三角形; 如果有兩個邊長相等, 就是等腰三角形; 否則, 就是普通的三角形。

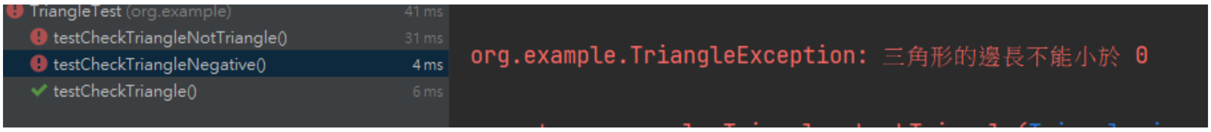
在JUnit測試中我先測試正常的輸入, 再來測試在輸入參數為負數時, 是否會拋出TriangleException 例外。

```
package org.example;

/**
* This is Triangle class.
* */
public class Triangle {
 /**
  * This is checkTriangle function.
  * */
 public String checkTriangle(int a, int b, int c) throws
TriangleException {
   if (a <= 0 || b <= 0 || c <= 0) {
     throw new TriangleException("三角形的邊長不能小於 0");
   }
   if (a + b <= c || a + c <= b || b + c <= a) {
     throw new TriangleException("兩邊和不可小於第三邊");
   }
   if (a == b && b == c) {
     return "正三角形";
   } else if (a == b || b == c || c == a) {
     return "等腰三角形";
   } else {
     return "三角形";
   }
 }
}
```

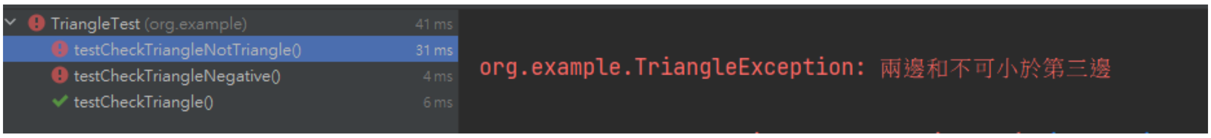
```
package org.example;

import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

class TriangleTest {
 @Test
 public void testCheckTriangle() throws TriangleException {
   assertEquals("正三角形", new Triangle().checkTriangle(3, 3,
3));
   assertEquals("等腰三角形", new Triangle().checkTriangle(3,
3, 5));
   assertEquals("三角形", new Triangle().checkTriangle(3, 4,
5));
 }

 @Test
 public void testCheckTriangleNegative() throws
TriangleException {
   new Triangle().checkTriangle(-1, 2, 3);
 }

 @Test
 public void testCheckTriangleNotTriangle() throws
TriangleException {
   new Triangle().checkTriangle(1, 2, 6);
 }

}
```



```
TriangleTest (org.example)            41 ms
  testCheckTriangleNotTriangle()      31 ms
  testCheckTriangleNegative()          4 ms
✓ testCheckTriangle()                  6 ms
```



```
TriangleTest (org.example)            41 ms
  testCheckTriangleNotTriangle()      31 ms
  testCheckTriangleNegative()          4 ms
✓ testCheckTriangle()                  6 ms

org.example.TriangleException: 三角形的邊長不能小於 0
```



```
TriangleTest (org.example)            41 ms
  testCheckTriangleNotTriangle()      31 ms
  testCheckTriangleNegative()          4 ms
✓ testCheckTriangle()                  6 ms

org.example.TriangleException: 兩邊和不可小於第三邊
```

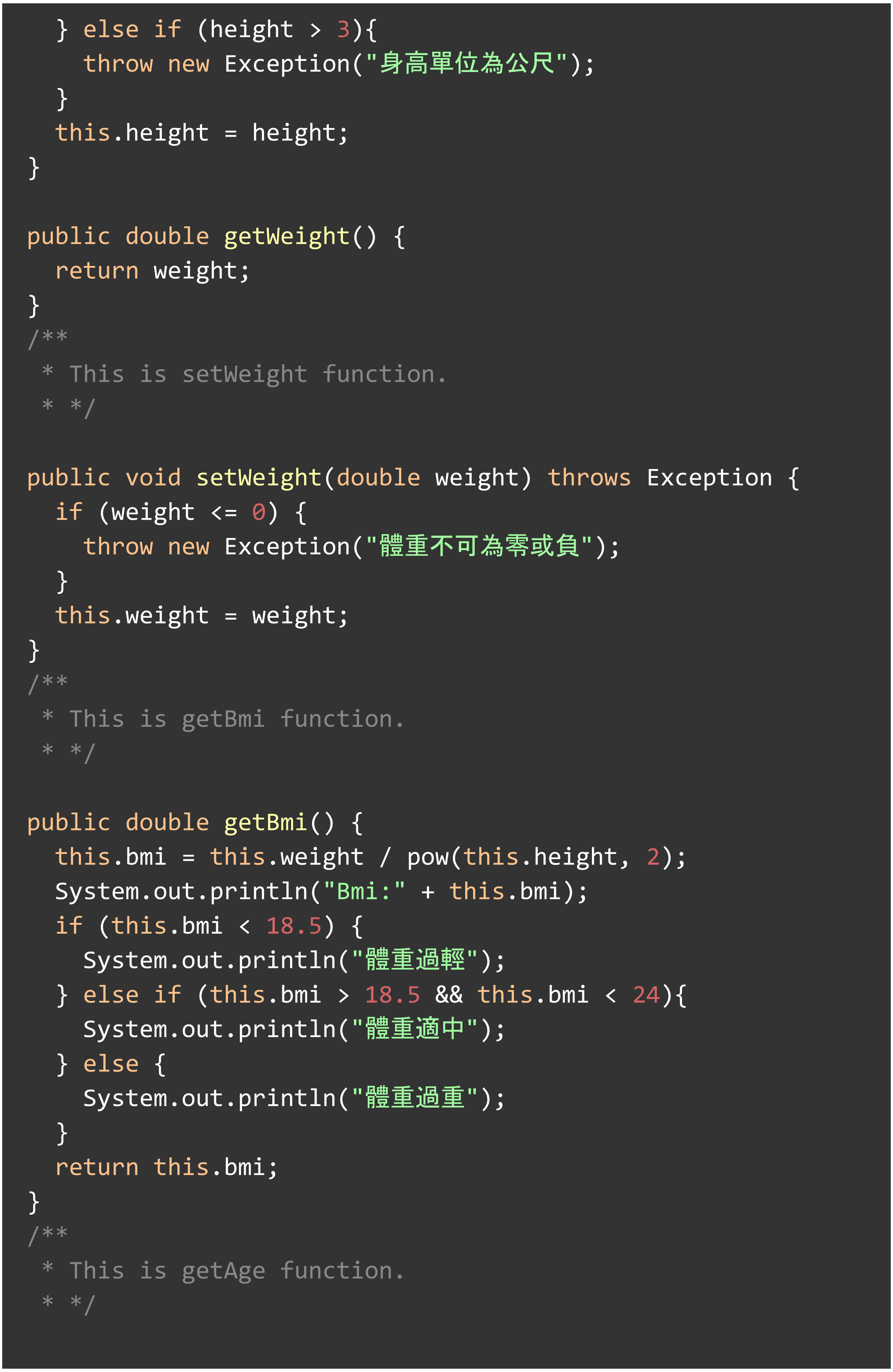
2.Person

設計說明:

在class Person中有建構子可使用，設定身高體重我用public getter、setter方法來存取private屬性，和計算bmi與年齡，並在其中有例外事件發生時，拋出Exception。

在JUnit測試中利用BeforeEach確認每次測試不會相互干擾，在依次測試，首先測試項目為:輸入身高為公分時是否抱錯、計算年齡、計算BMI

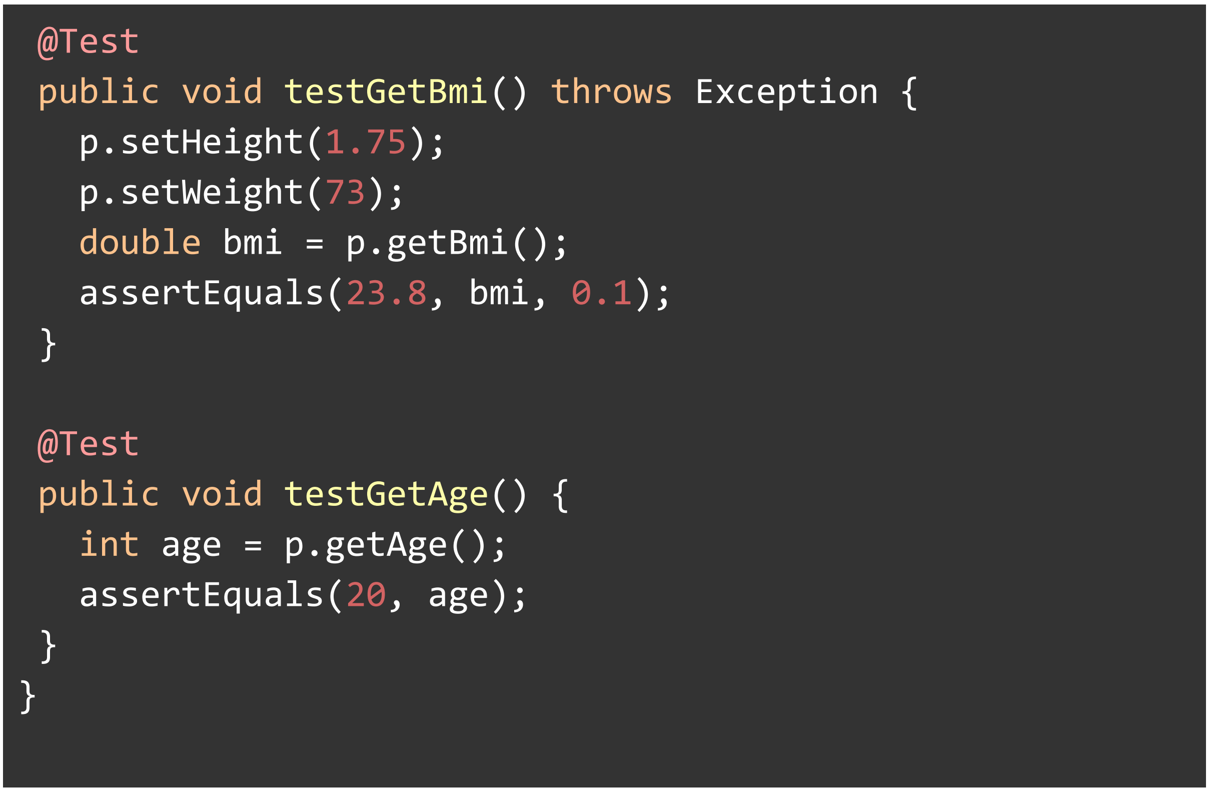
```
package org.example;

import static java.lang.Math.pow;
/**
* This is Person class.
* */

public class Person {
 private String name;
 private int birthYear;
 private double height;
 private double weight;
 private double bmi;
 private int age;
 /**
  * This is Person constructor.
  * */

 public Person(String name, int birthYear) throws Exception {
   this.name = name;
  if (birthYear <= 0) {
    throw new Exception("出生年不可小於0");
  }
  this.birthYear = birthYear;
}

public String getName() {
  return name;
}

public void setName(String name) {
  this.name = name;
}

public int getBirthYear() {
  return birthYear;
}
/**
 * This is setBirthYear function.
 * */

public void setBirthYear(int birthYear) throws Exception {
  if (birthYear <= 0) {
    throw new Exception("出生年不可小於0");
  }
  this.birthYear = birthYear;
}

public double getHeight() {
  return height;
}
/**
 * This is setHeight function.
 * */

public void setHeight(double height) throws Exception {
  if (height <= 0) {
    throw new Exception("身高不可為零或負");
```



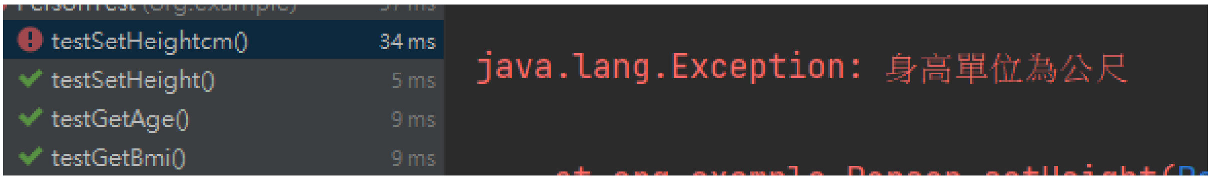
```
  } else if (height > 3){
    throw new Exception("身高單位為公尺");
  }
  this.height = height;
}

public double getWeight() {
  return weight;
}
/**
 * This is setWeight function.
 * */

public void setWeight(double weight) throws Exception {
  if (weight <= 0) {
    throw new Exception("體重不可為零或負");
  }
  this.weight = weight;
}
/**
 * This is getBmi function.
 * */

public double getBmi() {
  this.bmi = this.weight / pow(this.height, 2);
  System.out.println("Bmi:" + this.bmi);
  if (this.bmi < 18.5) {
    System.out.println("體重過輕");
  } else if (this.bmi > 18.5 && this.bmi < 24){
    System.out.println("體重適中");
  } else {
    System.out.println("體重過重");
  }
  return this.bmi;
}
/**
 * This is getAge function.
 * */
 public int getAge() {
   int thisYear = 2023;
   this.age = thisYear - this.birthYear;
   System.out.println("age:" + this.age);
   return this.age;
 }
}
```

JUnit:

```
package org.example;

import org.example.Person;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

import static
org.junit.jupiter.api.Assertions.assertEquals;

public class PersonTest {
 private Person p;

 @BeforeEach
 public void setUp() throws Exception {
   p = new Person("matt", 2003);
 }

 @Test
 public void testSetHeight() throws Exception {
   p.setHeight(1.75);
   assertEquals(1.75, p.getHeight());
 }

 @Test
 public void testSetHeightcm() throws Exception {
   p.setHeight(175);
 }
```

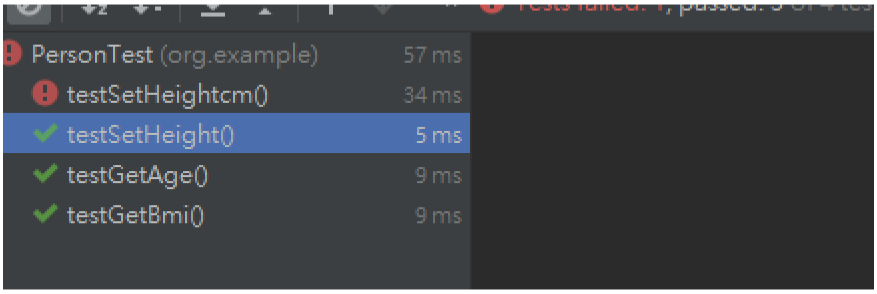


```java
@Test
public void testGetBmi() throws Exception {
   p.setHeight(1.75);
   p.setWeight(73);
   double bmi = p.getBmi();
   assertEquals(23.8, bmi, 0.1);
}

@Test
public void testGetAge() {
   int age = p.getAge();
   assertEquals(20, age);
}
}
```

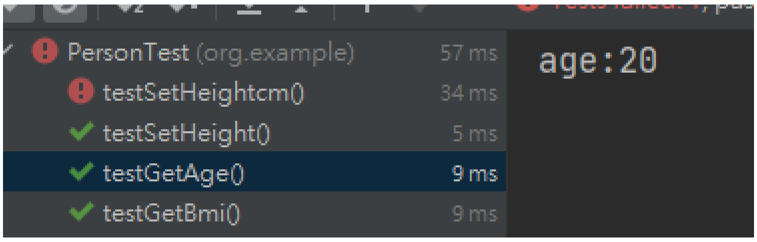


```
testSetHeightcm()   34 ms
testSetHeight()      5 ms
testGetAge()         9 ms
testGetBmi()         9 ms
```

```
java.lang.Exception: 身高單位為公尺
```

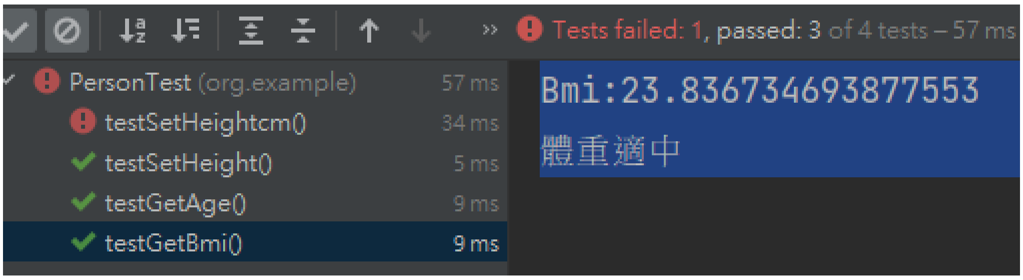


```
PersonTest (org.example)   57 ms
  testSetHeightcm()        34 ms
  testSetHeight()           5 ms
  testGetAge()              9 ms
  testGetBmi()              9 ms
```



```
PersonTest (org.example)   57 ms
  testSetHeightcm()        34 ms
  testSetHeight()           5 ms
  testGetAge()              9 ms
  testGetBmi()              9 ms
```

```
age:20
```



```
Tests failed: 1, passed: 3 of 4 tests – 57 ms
PersonTest (org.example)   57 ms
  testSetHeightcm()        34 ms
  testSetHeight()           5 ms
  testGetAge()              9 ms
  testGetBmi()              9 ms
```

```
Bmi:23.836734693877553
體重適中
```

心得:

在本次作業中我會開始揣摩使用者使用的例外場景，和自己寫的程式中是否有edge case，並使用Exception，及使用JUnit做單元測試，來實踐自己的想法或猜想是否正確、自己寫的Exception，在錯誤中是否會跳出錯誤。

在本次作業中我學習到軟體測試是軟體開發的重要環節，是確保軟體品質的關鍵也更了解軟體測試的流程、方法、工具等，並實踐設計和執行測試。