

Lab 1: Basic Unit Testing

Software Testing 2022

2022/02/24

Overview

Lab and Homework

1. 10 ~ 11 Lab

- a. *Unit Testing*
- b. *Continuous Integration*
- c. *Web Applications Testing*
- d. *Control Flow Graph*
- e. *Behavior-Driven Development*
- f. *Fuzz Testing*
- g. *Symbolic Execution*

2. Homework: TBD

Github

- <https://github.com/a4865g/NYCU-Software-Testing-2022>

Group

1. Microsoft Teams

- a. Login by Office 365 (NYCU)
 - i. user@m365.nycu.edu.tw
- b. <https://reurl.cc/pWpWlx>
- c.



2. Line

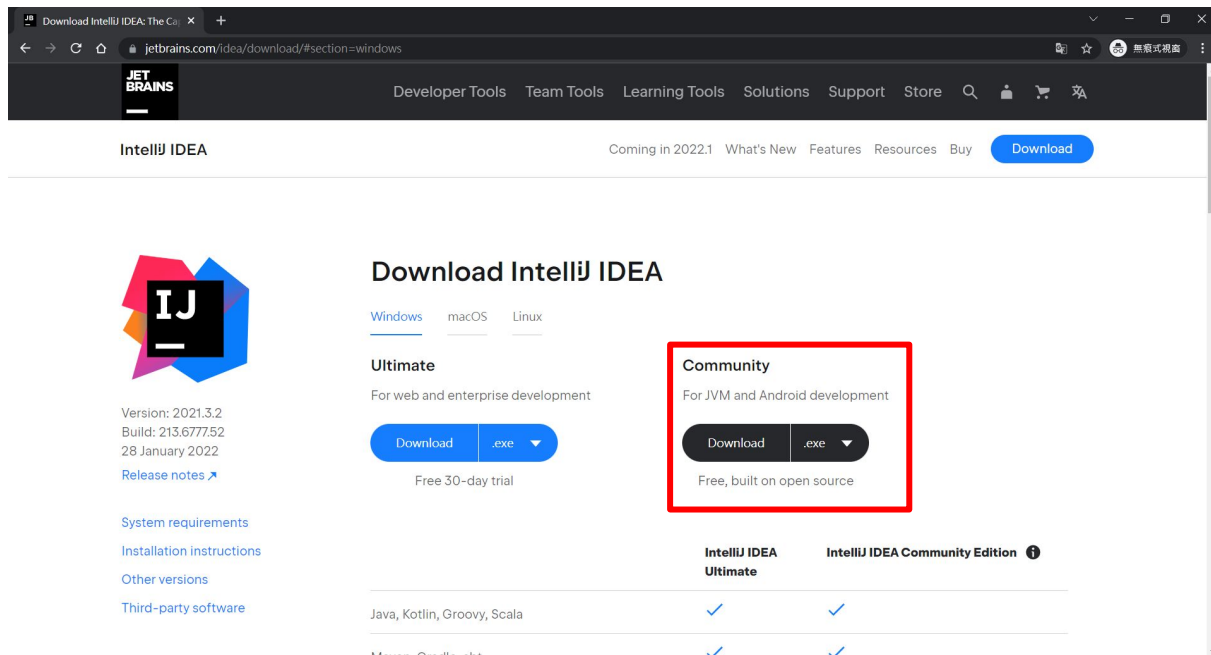
- a. <https://reurl.cc/xOKzYZ>
- b.



JAVA IDE

IntelliJ IDEA Community

→ <https://www.jetbrains.com/idea/download>



Download IntelliJ IDEA: The Community Edition

Developer Tools Team Tools Learning Tools Solutions Support Store

IntelliJ IDEA Coming in 2022.1 What's New Features Resources Buy [Download](#)

IntelliJ IDEA

Version: 2021.3.2
Build: 213.6777.52
28 January 2022
[Release notes](#)

[System requirements](#)
[Installation instructions](#)
[Other versions](#)
[Third-party software](#)

Download IntelliJ IDEA

[Windows](#) [macOS](#) [Linux](#)

Ultimate
For web and enterprise development

[Download](#) [.exe](#)

Free 30-day trial

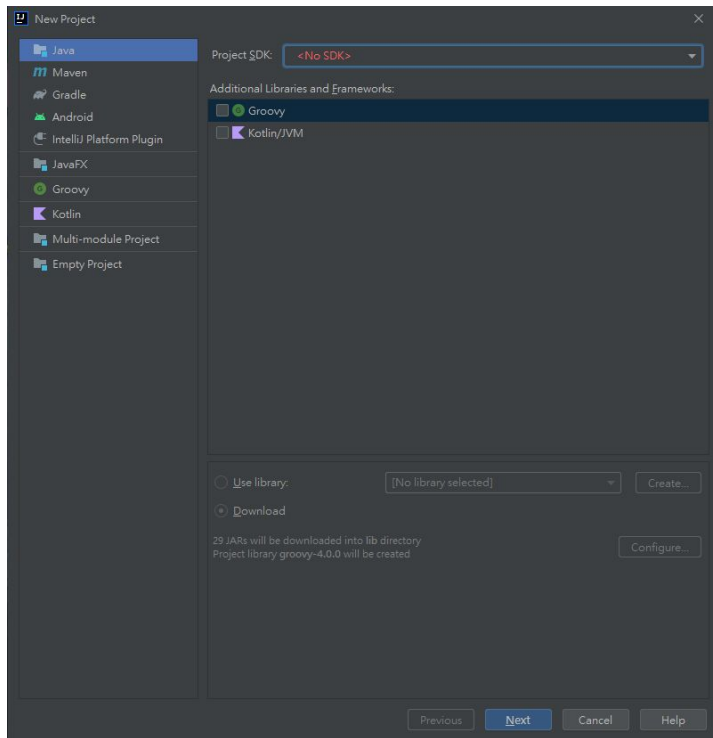
Community
For JVM and Android development

[Download](#) [.exe](#)

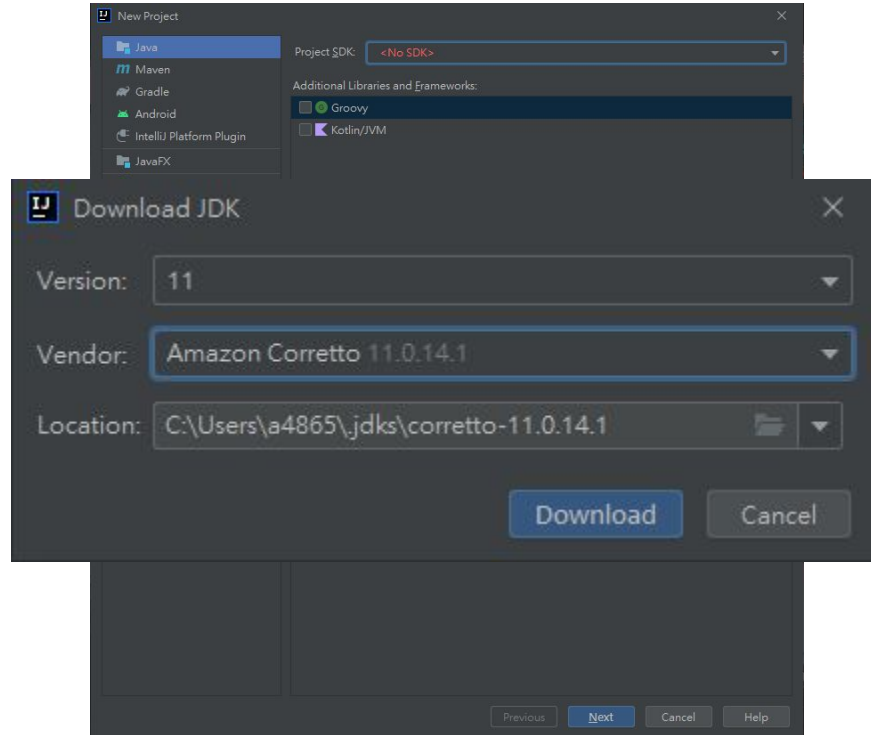
Free, built on open source

| | IntelliJ IDEA Ultimate | IntelliJ IDEA Community Edition |
|-----------------------------|------------------------|---------------------------------|
| Java, Kotlin, Groovy, Scala | ✓ | ✓ |
| Python, C++, Rust, etc. | ✓ | ✓ |

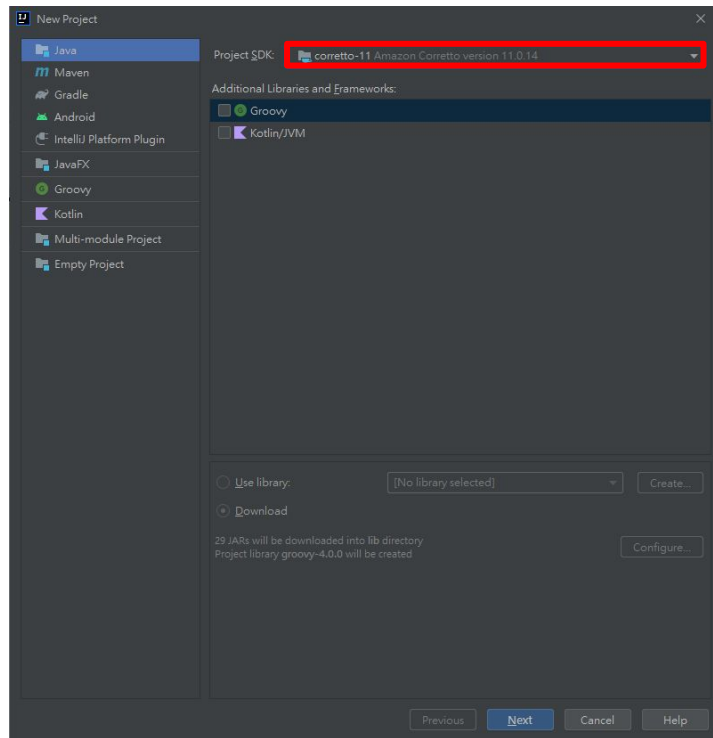
Guide - SDK



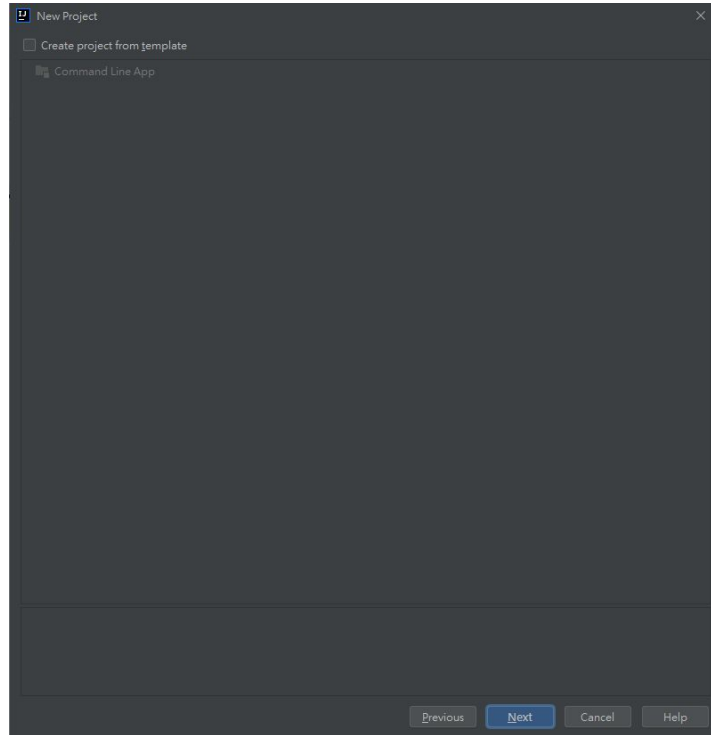
Guide - SDK



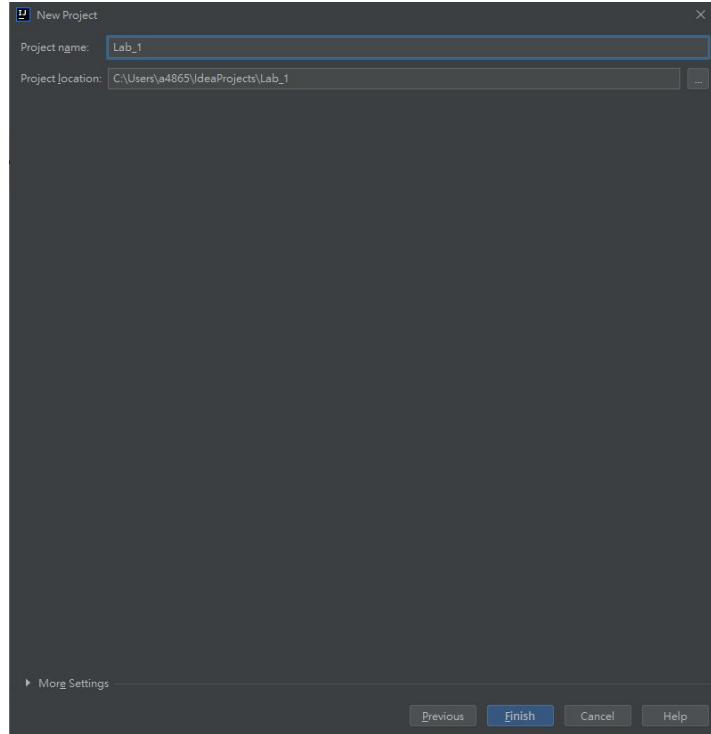
Guide - SDK



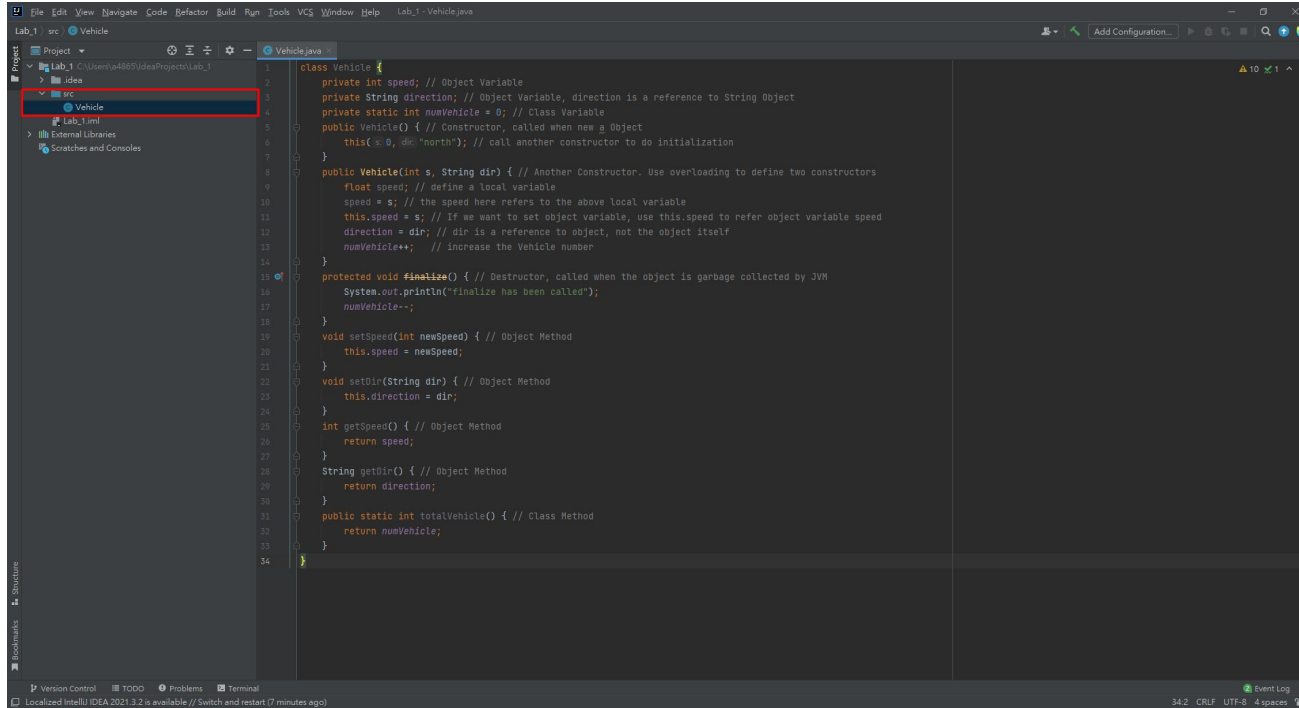
Guide - Template



Guide - Name



Guide - SRC

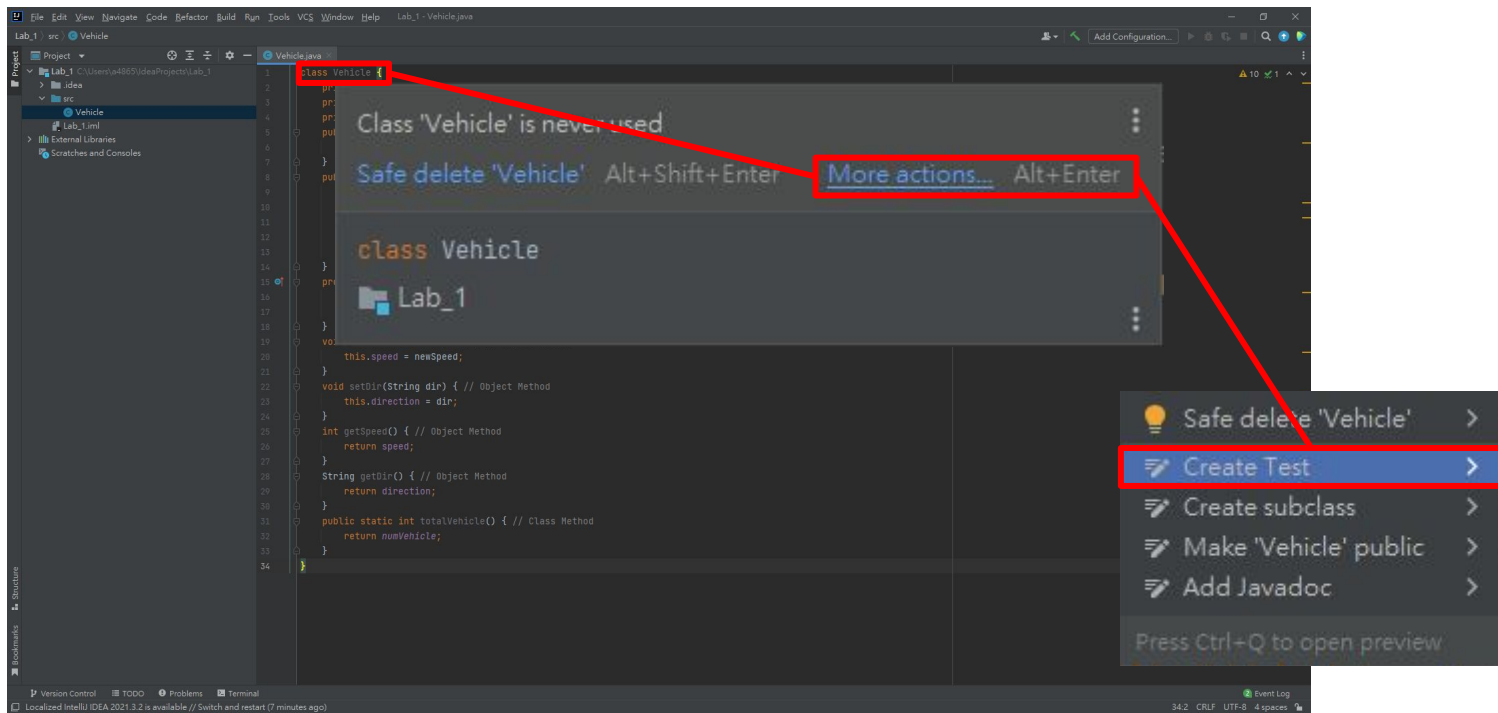


The screenshot shows an IDE window with a project named 'Lab_1'. The project structure on the left includes 'src' and 'Vehicle'. The 'Vehicle' class is selected, and its source code is displayed in the main editor. The code defines a 'Vehicle' class with attributes 'speed', 'direction', and 'numVehicle', and methods for initialization, setting/getting attributes, and a static method 'totalVehicle'.

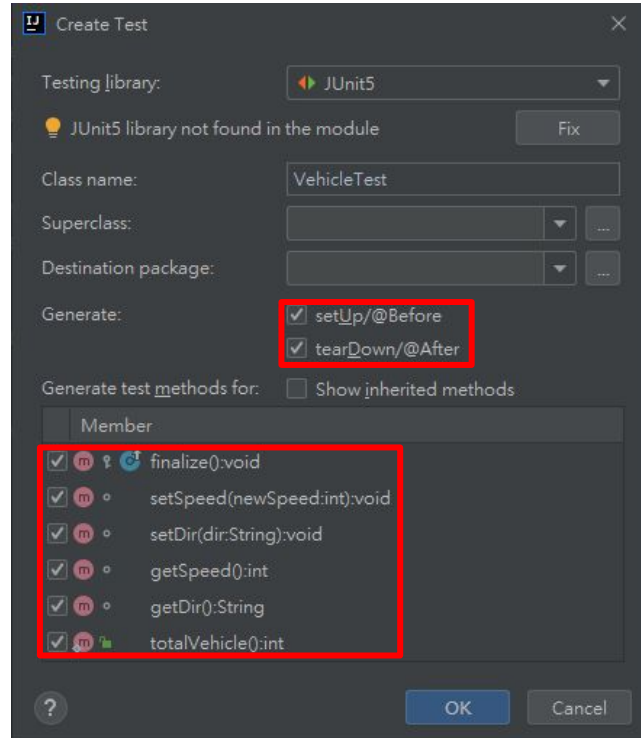
```
1 class Vehicle {
2     private int speed; // Object Variable
3     private String direction; // Object Variable, direction is a reference to String Object
4     private static int numVehicle = 0; // Class Variable
5     public Vehicle() { // Constructor, called when new a Object
6         this(0, "north"); // call another constructor to do initialization
7     }
8     public Vehicle(int s, String dir) { // Another Constructor. Use overloading to define two constructors
9         float speed; // define a local variable
10        speed = s; // the speed here refers to the above local variable
11        this.speed = s; // If we want to set object variable, use this.speed to refer object variable speed
12        direction = dir; // dir is a reference to object, not the object itself
13        numVehicle++; // Increase the Vehicle number
14    }
15    protected void finalize() { // Destructor, called when the object is garbage collected by JVM
16        System.out.println("finalize has been called");
17        numVehicle--;
18    }
19    void setSpeed(int newSpeed) { // Object Method
20        this.speed = newSpeed;
21    }
22    void setDir(String dir) { // Object Method
23        this.direction = dir;
24    }
25    int getSpeed() { // Object Method
26        return speed;
27    }
28    String getDir() { // Object Method
29        return direction;
30    }
31    public static int totalVehicle() { // Class Method
32        return numVehicle;
33    }
34 }
```

Create Test

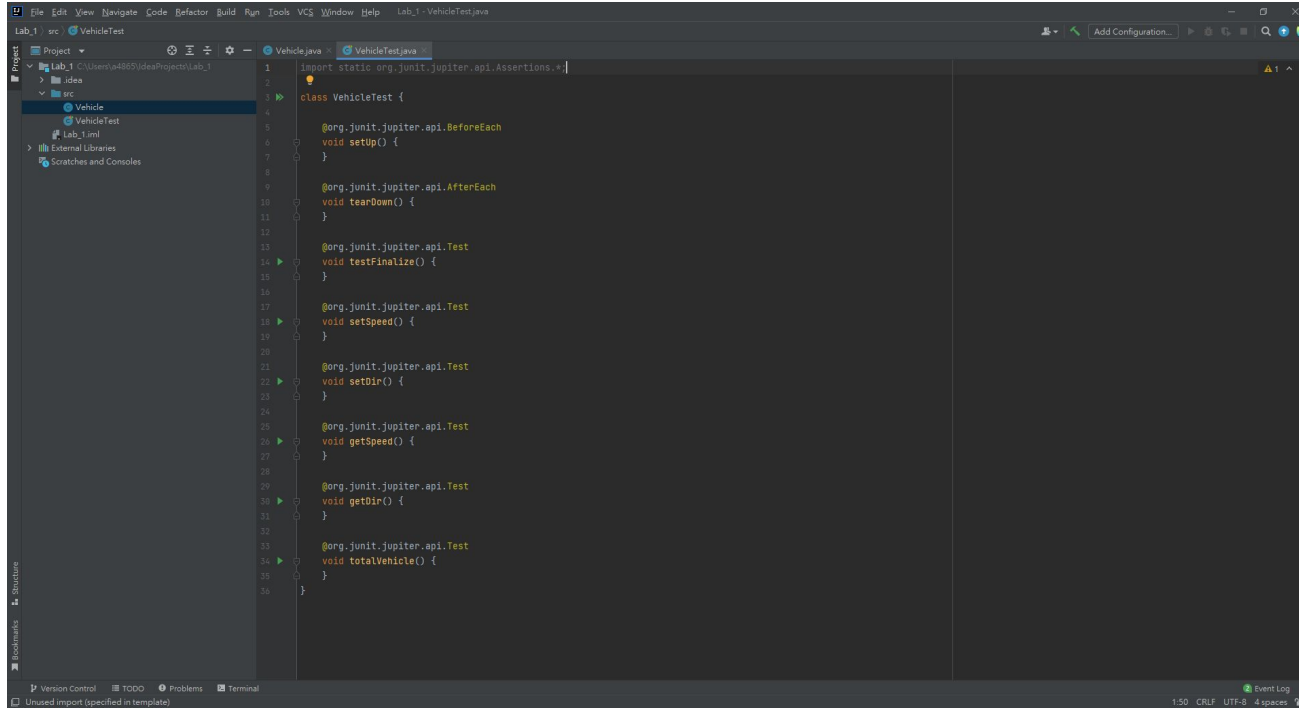
Create Test



Create Test - JUnit 5



Create Test - New File



The screenshot shows an IDE window with a project named 'Lab_1'. The 'src' directory is expanded, showing a package named 'Vehicle'. A new file named 'VehicleTest.java' has been created and is open in the editor. The code in the file is as follows:

```
1 import static org.junit.jupiter.api.Assertions.*;
2
3 class VehicleTest {
4
5     @org.junit.jupiter.api.BeforeEach
6     void setUp() {
7     }
8
9     @org.junit.jupiter.api.AfterEach
10    void tearDown() {
11    }
12
13    @org.junit.jupiter.api.Test
14    void testFinalize() {
15    }
16
17    @org.junit.jupiter.api.Test
18    void setSpeed() {
19    }
20
21    @org.junit.jupiter.api.Test
22    void setDir() {
23    }
24
25    @org.junit.jupiter.api.Test
26    void getSpeed() {
27    }
28
29    @org.junit.jupiter.api.Test
30    void getDir() {
31    }
32
33    @org.junit.jupiter.api.Test
34    void totalVehicle() {
35    }
36 }
```

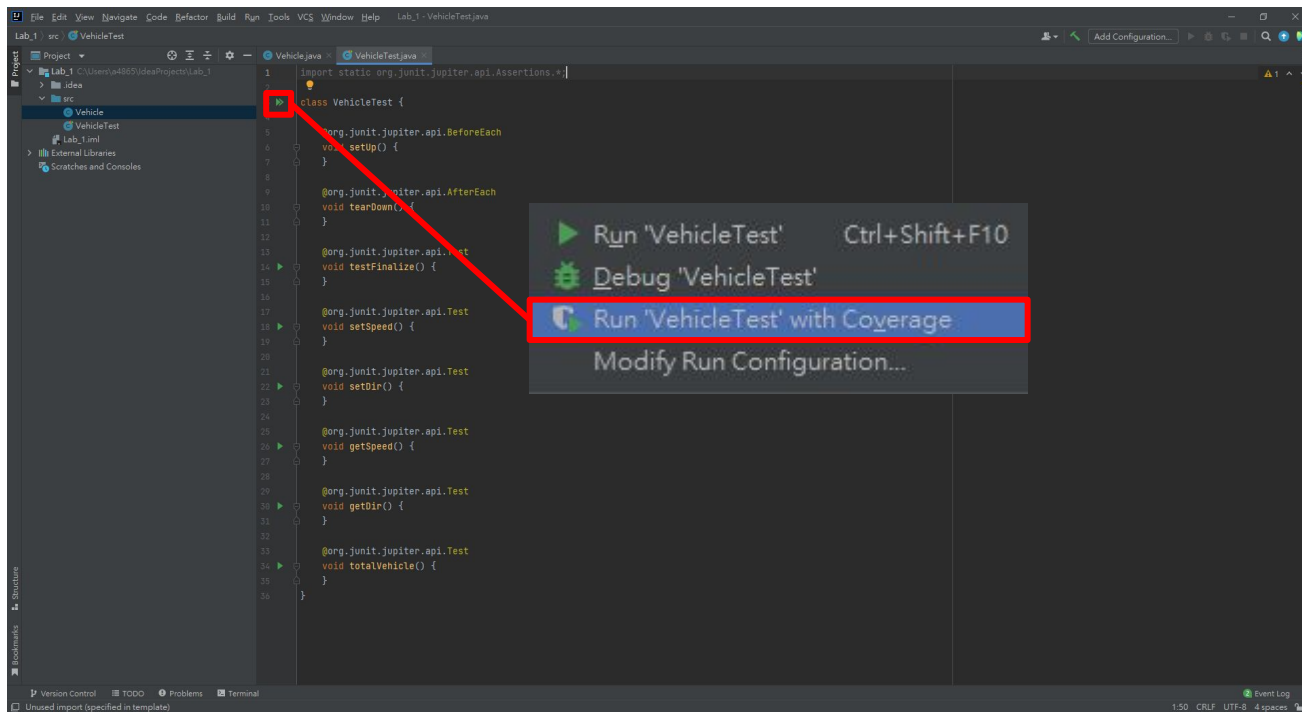
The IDE interface includes a menu bar at the top with options like File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, and Help. The left sidebar shows the project structure. The bottom status bar indicates the file encoding is UTF-8 and the line length is 150.

Code Coverage

Code Coverage

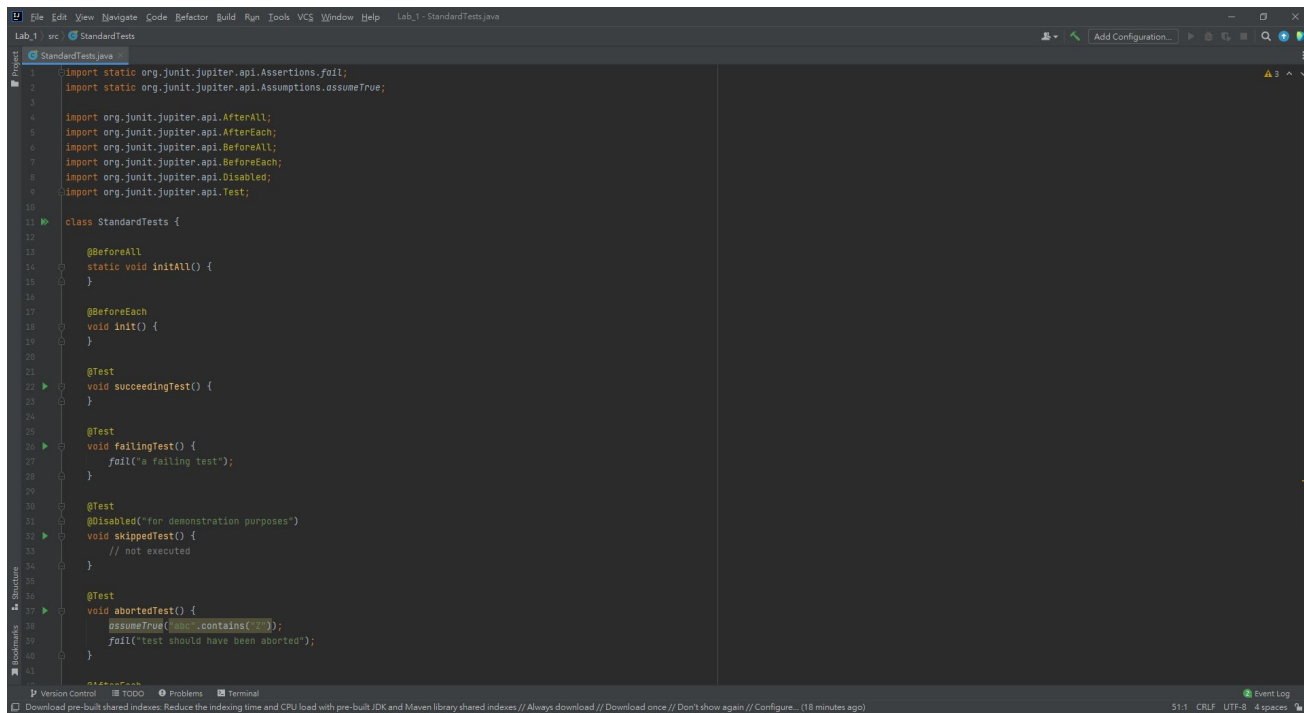
- In computer science, test coverage is **a measure (in percent) of the degree** to which the source code of **a program is executed** when a particular test suite is run.
- A program with high test coverage has more of its source code executed during testing, which suggests it has a lower chance of containing undetected software bugs compared to a program with low test coverage.

Get Code Coverage



JUnit 5

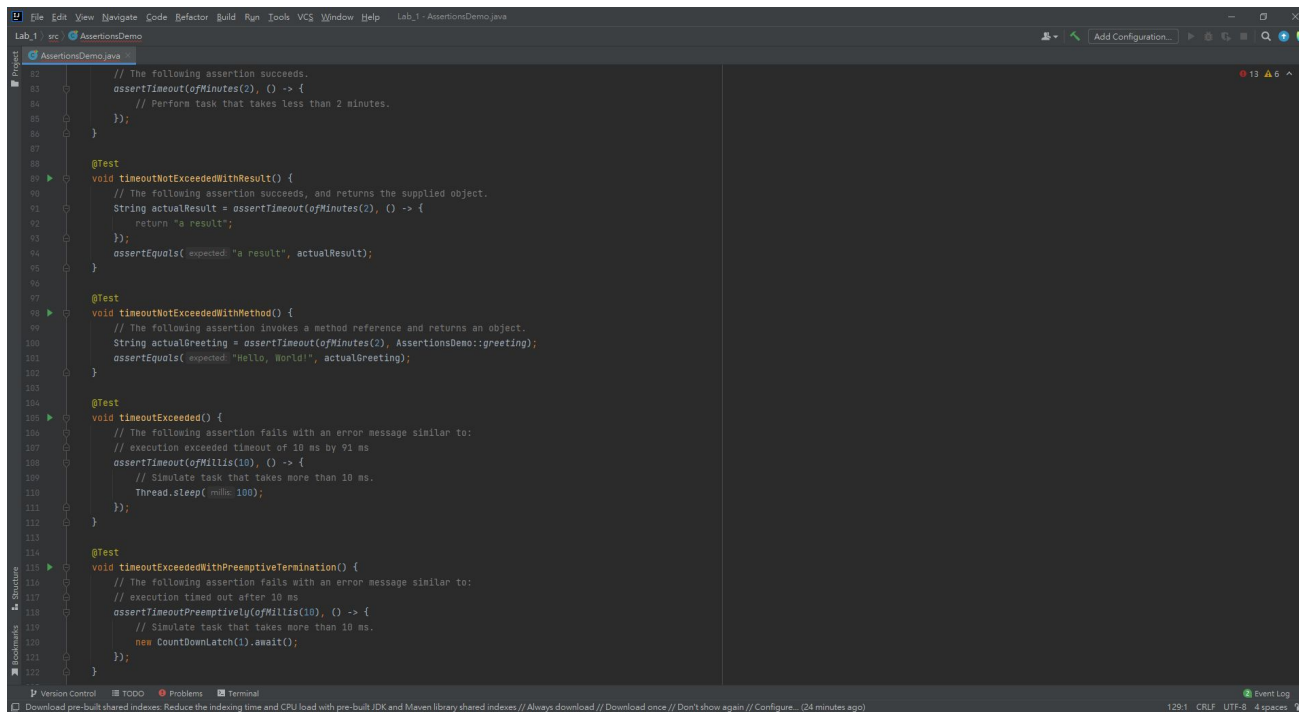
Test Classes and Methods



```
1 import static org.junit.jupiter.api.Assertions.fail;
2 import static org.junit.jupiter.api.Assertions.assertTrue;
3
4 import org.junit.jupiter.api.AfterAll;
5 import org.junit.jupiter.api.AfterEach;
6 import org.junit.jupiter.api.BeforeAll;
7 import org.junit.jupiter.api.BeforeEach;
8 import org.junit.jupiter.api.Disabled;
9 import org.junit.jupiter.api.Test;
10
11 class StandardTests {
12
13     @BeforeAll
14     static void initAll() {
15     }
16
17     @BeforeEach
18     void init() {
19     }
20
21     @Test
22     void succeedingTest() {
23     }
24
25     @Test
26     void failingTest() {
27         fail("a failing test");
28     }
29
30     @Test
31     @Disabled("for demonstration purposes")
32     void skippedTest() {
33         // not executed
34     }
35
36     @Test
37     void abortedTest() {
38         assertTrue("abc".contains("a"));
39         fail("test should have been aborted");
40     }
41 }
```

→ <https://junit.org/junit5/docs/current/user-guide/#writing-tests-classes-and-methods>

Assertions



```
1 // The following assertion succeeds.
2 assertTimeout(ofMinutes(2), () -> {
3     // Perform task that takes less than 2 minutes.
4 });
5
6
7
8
9 @Test
10 void timeoutNotExceededWithResult() {
11     // The following assertion succeeds, and returns the supplied object.
12     String actualResult = assertTimeout(ofMinutes(2), () -> {
13         return "a result";
14     });
15     assertEquals("expected: 'a result', actualResult");
16 }
17
18
19 @Test
20 void timeoutNotExceededWithMethod() {
21     // The following assertion invokes a method reference and returns an object.
22     String actualGreeting = assertTimeout(ofMinutes(2), AssertionsDemo::greeting);
23     assertEquals("expected: 'Hello, World!', actualGreeting");
24 }
25
26
27
28
29 @Test
30 void timeoutExceeded() {
31     // The following assertion fails with an error message similar to:
32     // execution exceeded timeout of 10 ms by 91 ms
33     assertTimeout(ofMillis(10), () -> {
34         // Simulate task that takes more than 10 ms.
35         Thread.sleep(100);
36     });
37 }
38
39
40
41
42 @Test
43 void timeoutExceededWithPreemptiveTermination() {
44     // The following assertion fails with an error message similar to:
45     // execution timed out after 10 ms
46     assertTimeoutPreemptively(ofMillis(10), () -> {
47         // Simulate task that takes more than 10 ms.
48         new CountDownLatch(1).await();
49     });
50 }
```

→ <https://junit.org/junit5/docs/current/user-guide/#writing-tests-assertions>



Lab

Lab

1. Download **Vehicle.java** from Github.
 - a. <https://github.com/a4865g/NYCU-Software-Testing-2022>
2. Cover all code with test cases.
3. Deliverables shall include the following:
 - a. Test Code: **VehicleTest.java**
 - b. Screenshot of test coverage: **Student_ID.png**

Hint: Assertion

Do not compress the files and plagiarism.

Screenshot - example

