Leonardo Rochinha Pedro

lr222qp@student.lnu.se

Github:
https://github.com/leonardopedro98/lr222qp_1DV600/releases/tag/Assignment3

Task 1 - Test Plan

Objectives
The objectives of testing this iteration is to find bugs, issues within the code that will then be fixed in the last iteration.

What to test?
We intend to test the UC2 ("Play Game")

How to test?
Manual Test Cases using the client application and Junit Testing.

Time Plan

Task	Estimated	Actual
Manual Test Case	45min	40min
JUnit test	1h 30min	1h 30min
Running test	40min	30min
Checking the code	30min	45min
Report of Testing	30 min	1h

Reflection

By the end of this iteration I intend to make sure every part of the code is running properly and with no problems while testing both manually and automatically.

And this serves as a guideline to the whole procedure of testing the game, so when I'm testing the program and fixing it, I still have a clear idea of what I need to be careful and what parts I need to cover for the testing part of the program in order to be working efficiently, so nothing gets forgotten and no features of the game get missed understood when testing the application

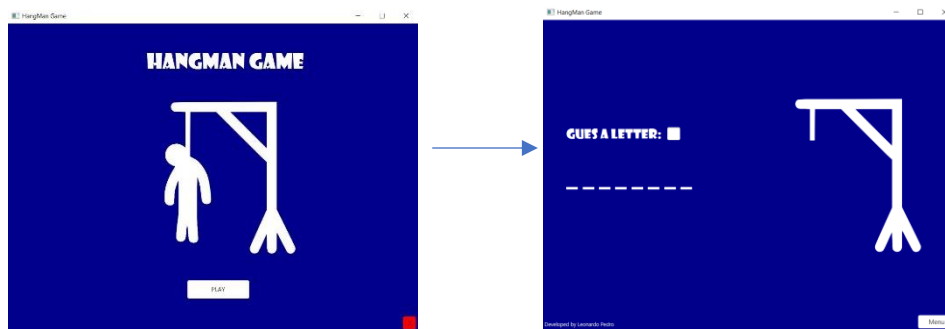# Task 2 – Manual Test Cases using the client application

TC2. 1 Win Game

Use Case Tested: UC2 ("Play Game")

Short Description: This test case is testing if the system shows the correct text when the player wins the game for UC2 , assuming all the user guesses are correct (perfect scenario).

Preconditions: UC1.
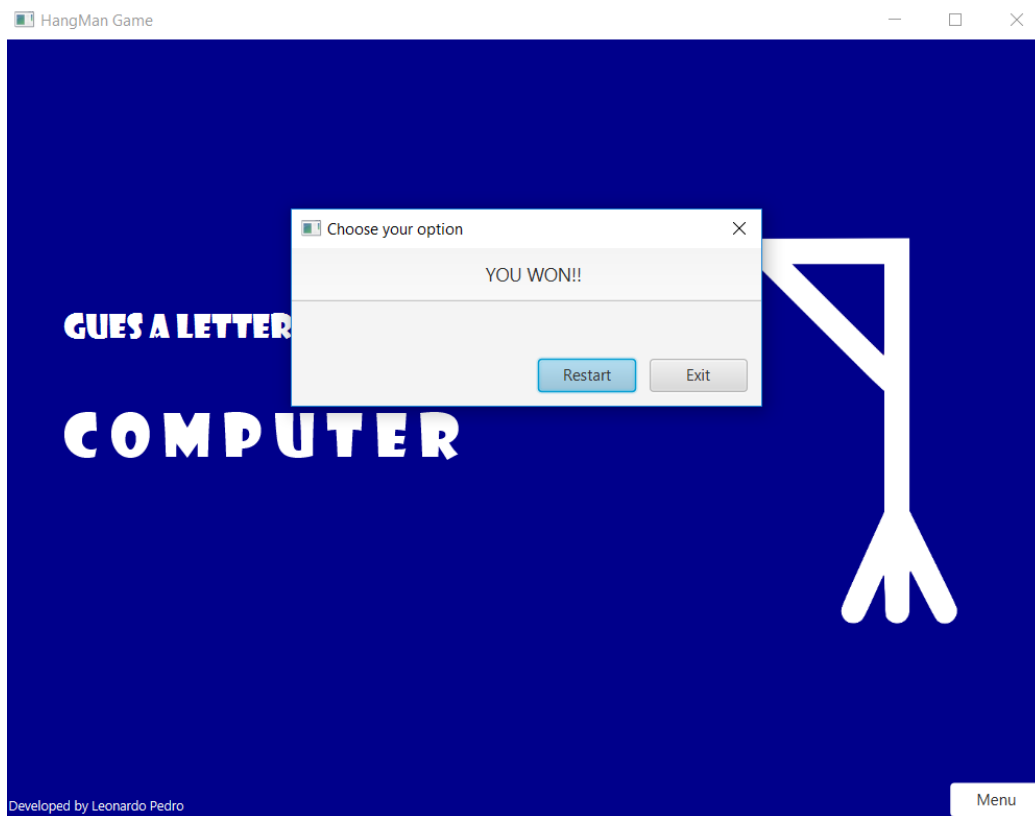
Test steps

- Press the "PLAY" Button.

- Write "c" and Press the Key "Enter".
- Write "o" and Press the Key "Enter".
- Write "m" and Press the Key "Enter".
- Write "p" and Press the Key "Enter".
- Write "u" and Press the Key "Enter".
- Write "t" and Press the Key "Enter".
- Write "e" and Press the Key "Enter".

Expected

- Write "r" and Press the Key "Enter".
- The system should show the text "Computer" and pop up a window saying "YOU WON!!" and asking if the user pretends to restart or exit.

TC2.2 Lose Game

Use Case Tested: UC2 ("Play Game")

Short Description: This test case is testing if the system shows the correct text when the player loses the game for UC2.
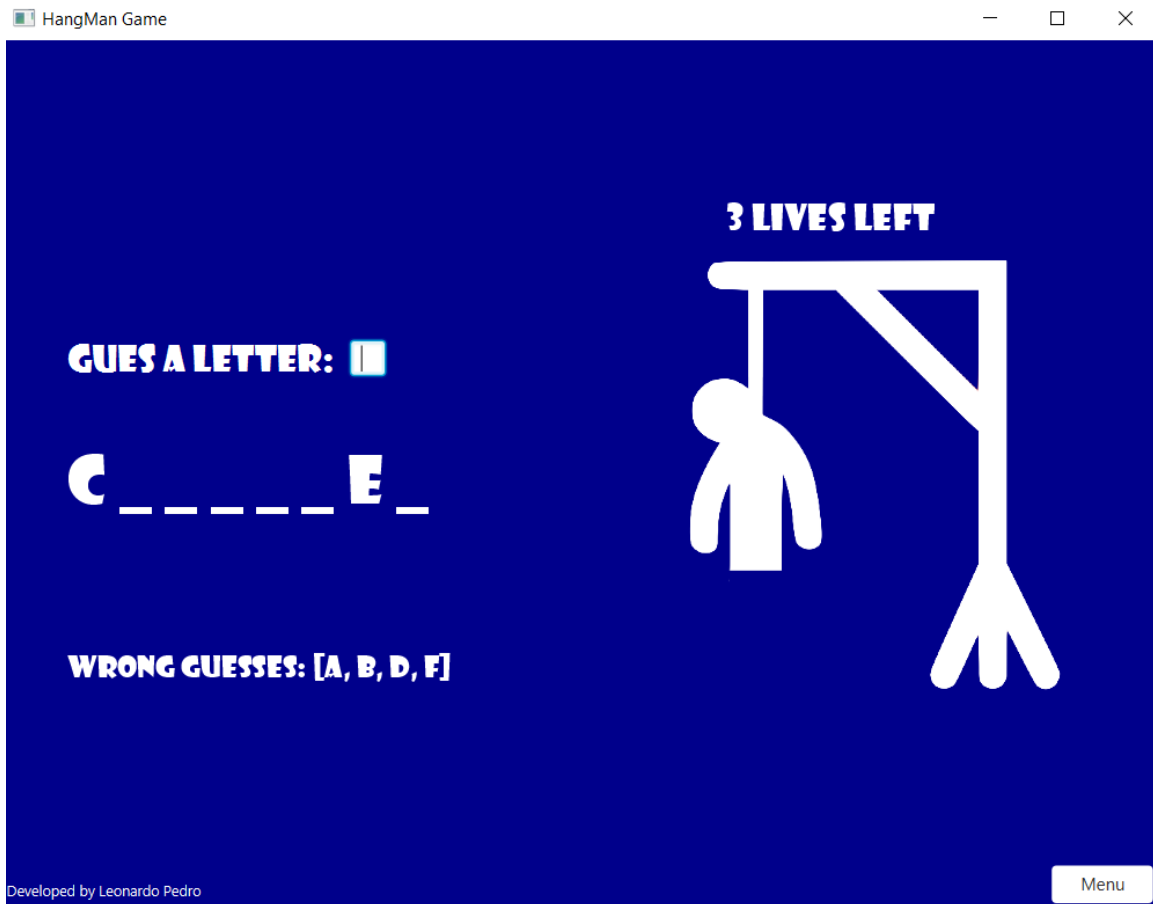
Preconditions: UC1.

Test steps

- Press the "PLAY" Button.

- Write "a" and Press the Key "Enter".
- Write "b" and Press the Key "Enter".
- Write "c" and Press the Key "Enter".
- Write "d" and Press the Key "Enter".
- Write "e" and Press the Key "Enter".
- Write "f" and Press the Key "Enter".
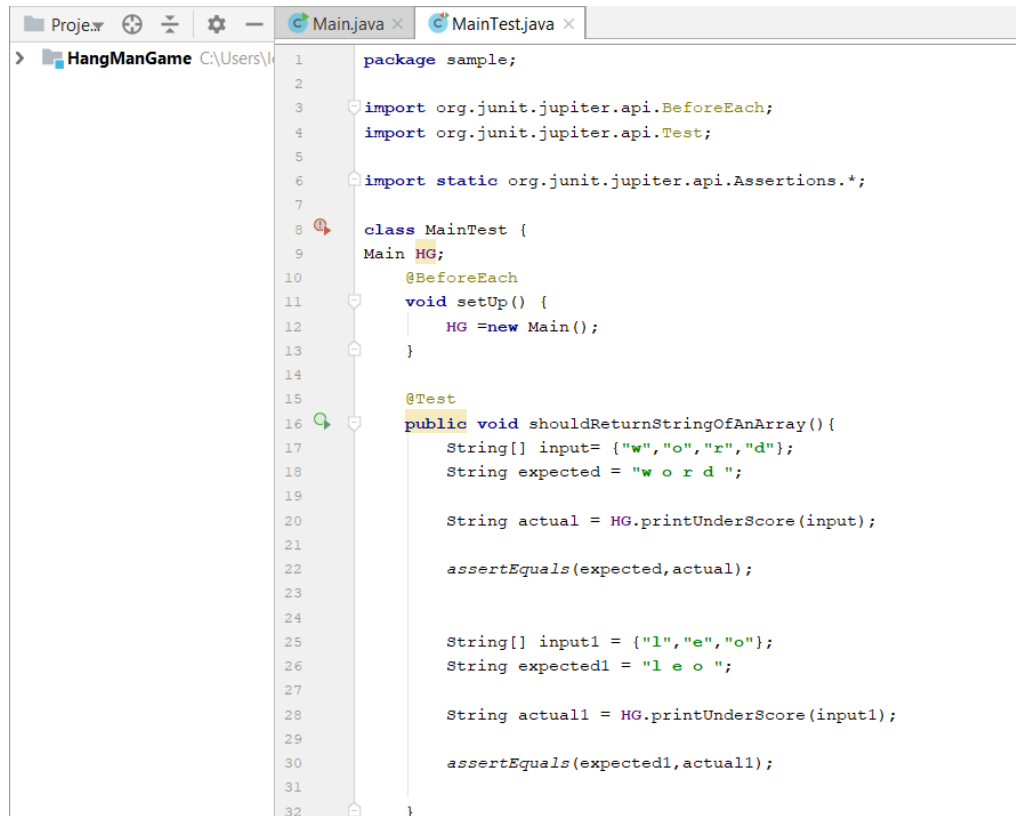
Expected
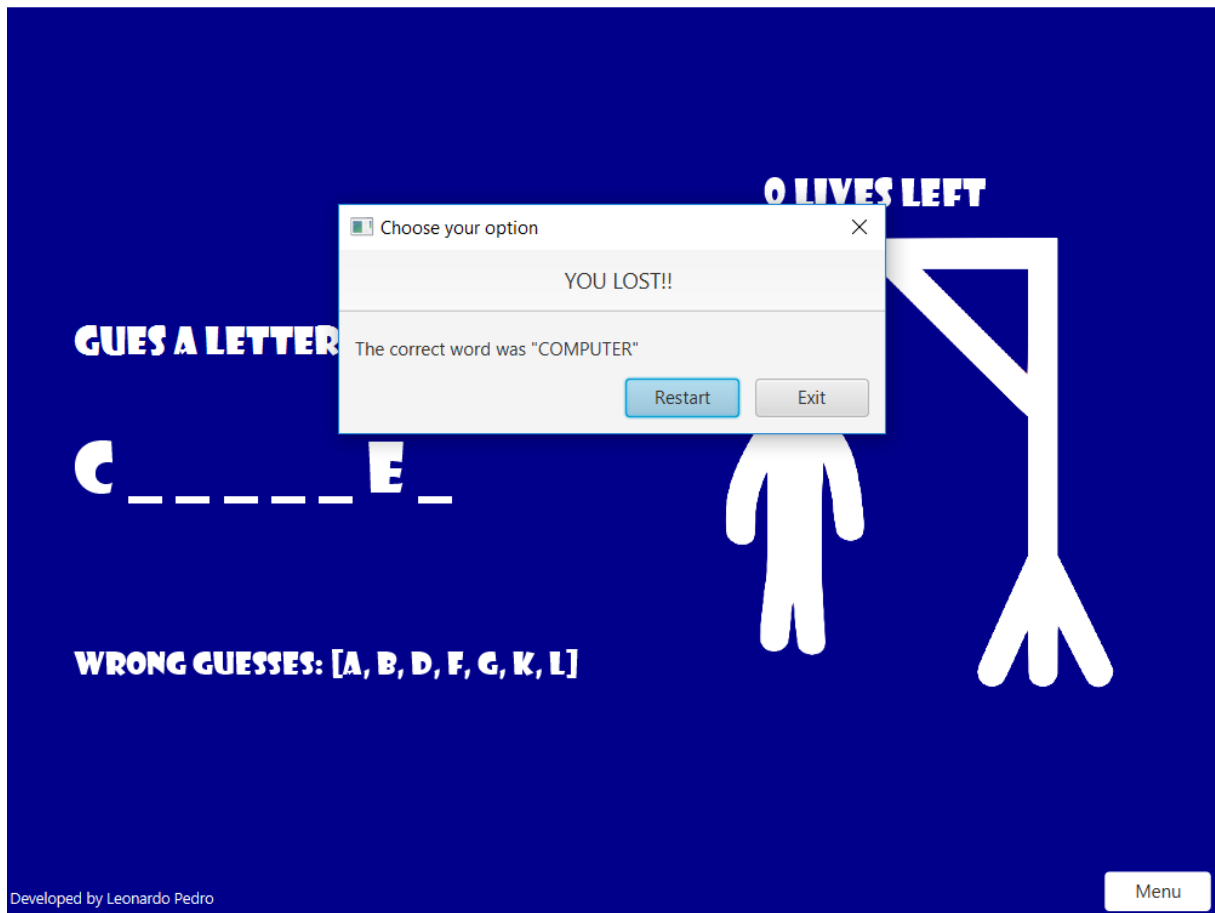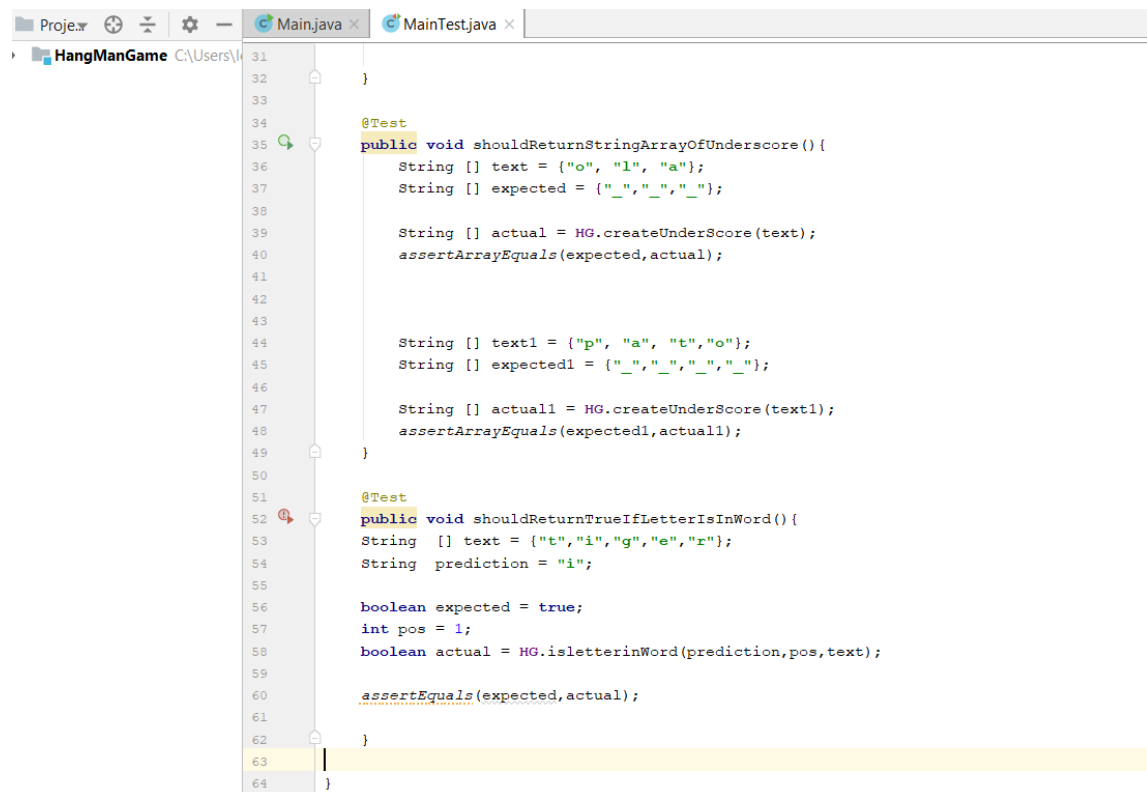
- Write "g" and Press the Key "Enter".
- Write "k" and Press the Key "Enter".
- Write "l" and Press the Key "Enter".
- The system should show all the wrong guess and the lives left as the game goes on.
- The system should pop up a window saying "YOU LOST!!" and revealing the correct word.
- The system should show a Button "Restart" and an "Exit" one.
- The image should show the hanged man.

HangMan Game

0 LIVES LEFT

GUES A LETTER

C _ _ _ _ _ E _

WRONG GUESSES: [A, B, D, F, G, K, L]

Choose your option

YOU LOST!!

The correct word was "COMPUTER"

Restart   Exit

Developed by Leonardo Pedro

Menu

Task 3, JUnit Test

```java
package sample;

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

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

class MainTest {
Main HG;
    @BeforeEach
    void setUp() {
        HG =new Main();
    }

    @Test
    public void shouldReturnStringOfAnArray(){
        String[] input= {"w","o","r","d"};
        String expected = "w o r d ";

        String actual = HG.printUnderScore(input);

        assertEquals(expected,actual);


        String[] input1 = {"l","e","o"};
        String expected1 = "l e o ";

        String actual1 = HG.printUnderScore(input1);

        assertEquals(expected1,actual1);

    }
```
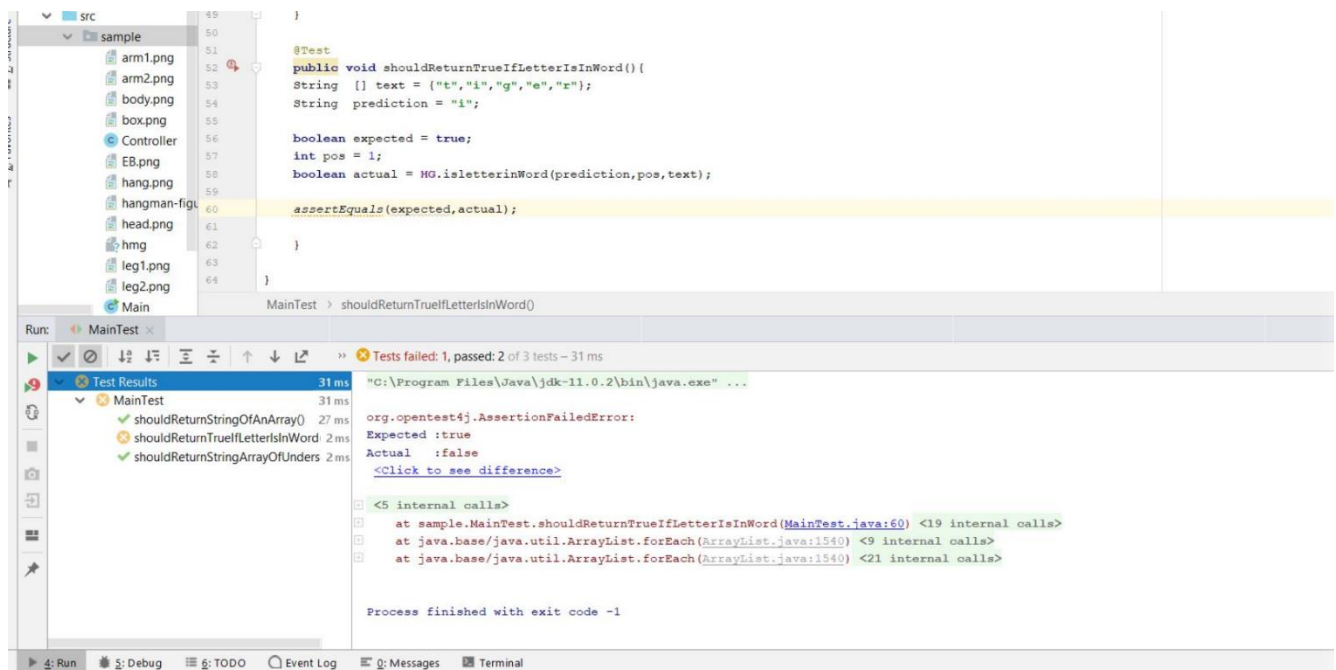
```java
    }

    @Test
    public void shouldReturnStringArrayOfUnderscore(){
        String [] text = {"o", "l", "a"};
        String [] expected = {"_","_","_"};

        String [] actual = HG.createUnderScore(text);
        assertArrayEquals(expected,actual);



        String [] text1 = {"p", "a", "t","o"};
        String [] expected1 = {"_","_","_","_"};

        String [] actual1 = HG.createUnderScore(text1);
        assertArrayEquals(expected1,actual1);
    }

    @Test
    public void shouldReturnTrueIfLetterIsInWord(){
    String [] text = {"t","i","g","e","r"};
    String prediction = "i";

        boolean expected = true;
        int pos = 1;
        boolean actual = HG.isletterinWord(prediction,pos,text);

        assertEquals(expected,actual);

    }
}
```

```
  src
  ├─ sample
  │   ├─ arm1.png
  │   ├─ arm2.png
  │   ├─ body.png
  │   ├─ box.png
  │   ├─ Controller
  │   ├─ EB.png
  │   ├─ hang.png
  │   ├─ hangman-figu
  │   ├─ head.png
  │   ├─ hmg
  │   ├─ leg1.png
  │   ├─ leg2.png
  │   └─ Main
```

```
49          }
50
51          @Test
52          public void shouldReturnTrueIfLetterIsInWord(){
53              String  [] text = {"t","i","g","e","r"};
54              String  prediction = "i";
55
56              boolean expected = true;
57              int pos = 1;
58              boolean actual = HG.isletterinWord(prediction,pos,text);
59
60              assertEquals(expected,actual);
61
62          }
63
64      }
```

MainTest > shouldReturnTrueIfLetterIsInWord()

Run: MainTest

Tests failed: 1, passed: 2 of 3 tests – 31 ms

```
Test Results                              31 ms
  MainTest                                31 ms
    shouldReturnStringOfAnArray()   27 ms
    shouldReturnTrueIfLetterIsInWord 2 ms
    shouldReturnStringArrayOfUnders  2 ms
```

```
"C:\Program Files\Java\jdk-11.0.2\bin\java.exe" ...

org.opentest4j.AssertionFailedError:
Expected :true
Actual   :false
<Click to see difference>

<5 internal calls>
    at sample.MainTest.shouldReturnTrueIfLetterIsInWord(MainTest.java:60) <19 internal calls>
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1540) <9 internal calls>
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1540) <21 internal calls>


Process finished with exit code -1
```

4: Run   5: Debug   6: TODO   Event Log   0: Messages   Terminal

```java
public String printUnderScore(String[] arr){
    String help="";
    for (int i = 0; i < arr.length; i++) {
        help=  help+ arr[i] + " ";
    }
    return help;
}
```

```java
public String[] createUnderScore(String[] arr){

    String[] help = arr.clone();

    for (int i = 0; i < help.length; i++) {
        help[i] = help[i].replaceAll( regex: "[A-Za-z]", replacement: "_");
    }
    return help;
}
```

```java
public boolean isletterinWord(String str,int position,String[] arr){
    String help = arr[position];
    if (str.equals(help)) {
        return true;
    }else
        return false;
}
```