

Q1: Your task is to write a program: Hangman using TDD.

Hangman.java

package hangman;

```
import java.util.HashSet;
import java.util.Random;
import java.util.Set;
import java.util.Scanner;

class Hangman_Program {

    public static void main(String[] args) {
        int lL = 97;
        int rL = 122;
        int tSL = 5;
        Random rr = new Random();
        String wTG = rr.ints(lL, rL + 1)
            .limit(tSL)
            .collect(StringBuilder::new, StringBuilder::appendCodePoint,
StringBuilder::append)
            .toString();
        new Hangman_Program().gw(wTG);
    }

    public void gw(String wTG) {
        int c = 0;
        Set<String> pG = new HashSet<>();
        Scanner i = new Scanner(System.in);
        int l = wTG.length();
        char[] wTGC = wTG.toCharArray();
        char[] cc = wTG.toCharArray();
        Print_Static("Secret word: ");
        For(l, cc);
        while (!String.valueOf(cc).equals(wTG)) {
            boolean ccc = false;
            boolean rrr = false;
            for (int a = 0; a < l; a++) {
                System.out.print(cc[a]);
            }
            System.out.println();
            Print_Static("Guess? ");
            String cGG = i.next().substring(0, 1);
            char cGGG = cGG.charAt(0);
            if (pG.contains(cGGG))
            {
                Print_Static("Try again. Your previous: ");
                System.out.println(pG.stream().reduce("", String::concat));
                rrr = true;
            }
            pG.add(cGGG);
            if (!rrr) {
                int times = 0;
                for (int iii = 0; iii < l; iii++) {
                    if (wTGC[iii] == cGGG) {
                        cc[iii] = cGGG;
                        ccc = true;
                        times++;
                    }
                }
                if (ccc) {
                    Print(cGGG, times);
                } else {
                    Print_Static("Sorry, the letter is not in the word. Your
secret word: ");
                }
            }
        }
    }
}
```

```

        }
        System.out.println();
    }
    c++;
}
System.out.println("You guessed the entire word " + wTG.toUpperCase() +
" correctly! It took you " + c + " attempts!");
}

private void For(int l, char[] cc) {
    for (int iii = 0; iii < l; iii++) {
        cc[iii] = '_';
    }
}

private void Print_Static(String s) {
    System.out.println(s);
}

private void Print(char cGGG, int times) {
    System.out.println("The letter " + cGGG + " is in the secret word! There
are " + times + " " + cGGG + "'s in the word. Revealing the letter(s): ");
}
}

```

Q2: Refactor your code – Code smell can give indications that there is some issue with the codes and can be solved by refactoring.

Code Smells

The names of the variables should always complete. The variables are self-explained so that these does not become mislead or confusing. I have make the separate method for generating the random numbers. Now the methods and variables are self-explanatory.

Updated Code:

```

import java.util.HashSet;
import java.util.Random;
import java.util.Scanner;
import java.util.Set;

public class HangManClass_Program {

    public static void main(String[] args) {
        new HangManClass_Program().gW(gRR());
    }

    private static String gRR(){
        int lL = 97;
        int rL = 122;
        int tSL = 4;
        Random r = new Random();
        return r.ints(lL, rL + 1)
            .limit(tSL)
            .collect(StringBuilder::new, StringBuilder::appendCodePoint,
StringBuilder::append)
            .toString();
    }
}

```

```

}

public void gW(String wwTG) {
    int aaa = 0;
    Set<String> pGG = new HashSet<>();
    Scanner i = new Scanner(System.in);
    int wL = wwTG.length();
    char[] wTGC = wwTG.toCharArray();
    char[] c = wwTG.toCharArray();
    System.out.print("Secret word: ");
    System.out.println(wwTG);
    for (int o = 0; o < wL; o++) {
        c[o] = '_';
    }
    while (!String.valueOf(c).equals(wwTG)) {
        boolean iC = false;
        boolean iR = false;
        for (int a = 0; a < wL; a++) {
            System.out.print(c[a]);
        }
        System.out.println();
        System.out.print("Guess? ");
        String cG = i.next().substring(0, 1);
        char cGC = cG.charAt(0);
        if (pGG.contains(cG))
        {
            System.out.println("Try again. Your previous guesses: ");
            System.out.println(pGG.stream().reduce("", String::concat));
            iR = true;
        }
        pGG.add(cG);
        if (!iR) {
            int t = 0;
            for (int index = 0; index < wL; index++) {
                if (wTGC[index] == cGC) {
                    c[index] = cGC;
                    iC = true;
                    t++;
                }
            }
            if (iC) {
                System.out.print("The letter ");
                System.out.print(cGC);
                System.out.print(" is in the secret word! There are ");
                System.out.print(t);
                System.out.print(" ");
                System.out.print(cGC);
                System.out.println("'s in the word.");
            } else {
                System.out.println("Sorry, the letter is not in the word.
Your secret word: ");
            }
            System.out.println();
        }
        aaa++;
    }
    System.out.print("You guessed the entire word ");
    System.out.print(wwTG.toUpperCase());
    System.out.print(" correctly! It took you ");
    System.out.print(aaa);
    System.out.print(" attempts!");
}

```

@Override

```
public String toString() {  
    return "HangManClass{}";  
}  
}
```

Q3: Create a Git directory for your assignment (including word or pdf documents and programming code)

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

[Import a repository.](#)

Owner *

 coding366 ▾

Repository name *

/ Hangman ✓

Great repository names are short and memorable. Need inspiration? How about **crispy-enigma**?

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☐ **Initialize this repository with a README**

This will let you immediately clone the repository to your computer.

Add .gitignore: None ▾

Add a license: None ▾



Creating repository...

Link to the Github Repository : <https://github.com/coding366/Hangman>

Hangman /

Drag files here to add them to your repository

Or [choose your files](#)

Commit changes

Add files via upload

Add an optional extended description...

master1 branch0 tags

Go to fileAdd fileCode

coding366 added hangman.java70b8d49 now1 commits

HangMan.javaadded hangman.janow

Help people interested in this repository understand your project by adding a README.

Add a README

About

No description, website, or topics provided.

Releases

No releases published
[Create a new release](#)

Packages

No packages published
[Publish your first package](#)

master


Hangman / HangMan.java / <> Jump to

Go to file

...

 coding366 added hangman.java

Latest commit 70b8d49 28 seconds ago  History

 1 contributor

72 lines (67 sloc) | 3.34 KB

Raw

Blame

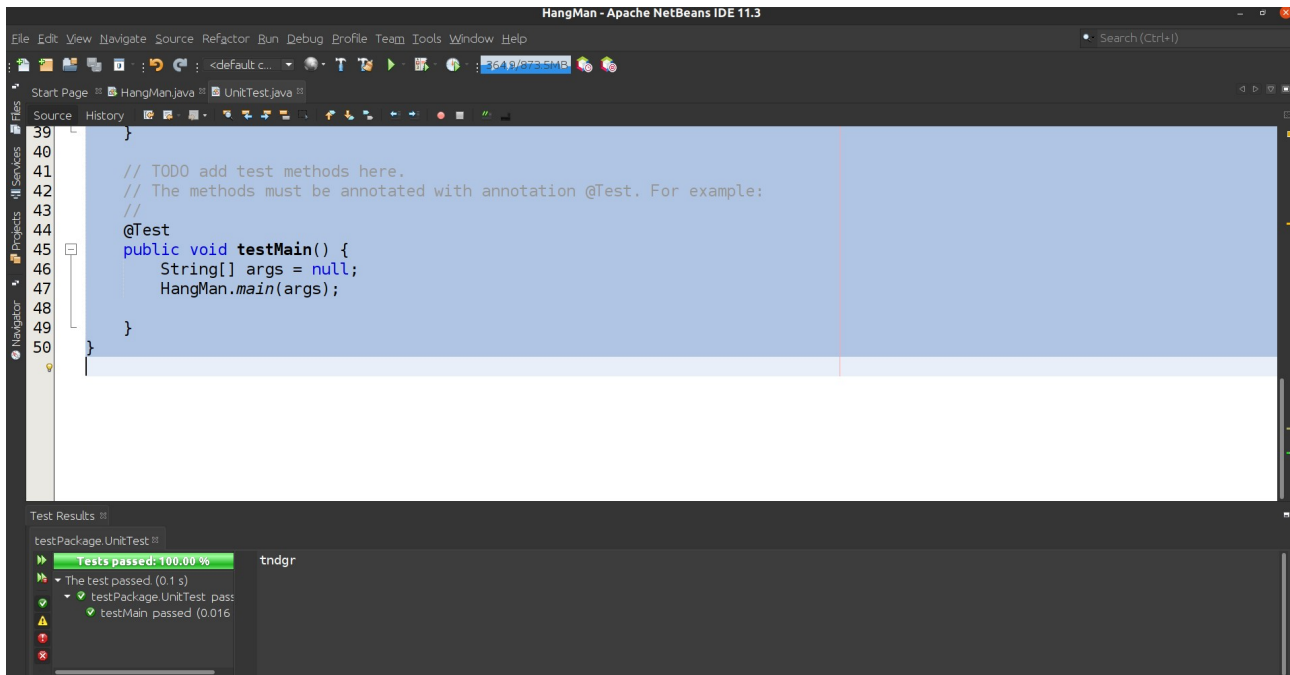


```
1 package hangman;
2
3 import java.util.HashSet;
4 import java.util.Random;
5 import java.util.Set;
6 import java.util.Scanner;
7
8 class Hangman {
9
10     public static void main(String[] args) {
11         int chances = 0;
12         Set<String> prevGuesses = new HashSet<>();
13         Scanner input = new Scanner(System.in);
14         int leftLimit = 97; // letter 'a'
15         int rightLimit = 122; // letter 'z'
16         int targetStringLength = 5; // length of the word
17         Random random = new Random();
18         // generating random word using pre-built java8 method.
19         String wordToGuess = random.ints(leftLimit, rightLimit + 1)
20             .limit(targetStringLength)
21             .collect(StringBuilder::new, StringBuilder::appendCodePoint, StringBuilder::append)
22         .....
```

Q No.4

a) how TDD has been implemented to create your program.

We have used the TDD approach for the development of this application. We could easily test using the method of Random Generator to check that a random no of specified is being generated or not. The screenshot of the tests are being attached.



Code :

```
import org.junit.After;
import org.junit.AfterClass;
import org.junit.Before;
import org.junit.BeforeClass;
import org.junit.Test;
import static org.junit.Assert.*;

public class HangManClass_ProgramTest {

    public HangManClass_ProgramTest() {

    }

    @Test
    public void tM() {
        System.out.println("main");
        String[] args = null;
        HangManClass.main(args);
        fail("The test case is a prototype.");
    }

    @Test
    public void tTS() {
        System.out.println("toString");
        HangManClass ii = new HangManClass();
        String eR = "";
        String rr = ii.toString();
        assertEquals(eR, rr);
    }
}
```



```
        fail("The test case is a prototype.");
    }

    @Test
    public void tRG() {
        System.out.println("Random Generator");
        HangManClass ii = new HangManClass();
        ii.guessWord(ii.generateRandomWord());
        fail("The test case is a prototype.");
    }
}
```

b) How refactoring has been done, the code smell, issues and solutions.

The refactoring of the code is done and also the latest code committed to github repository. You could check the commit history. The issue were that the variables were not self-explanatory and all the code is the main method. If any problem occurs then I have to debug complete code to find out the issue. But now different methods are formed according to their functions. Now the methods and variables are well-defined.