Test Documentation

Task II

Criteria applied for all functions:

**To get 100 % methods coverage, 100 % methods lines coverage , 100 % outputs handling , coverage .**

1-For Length() method:

to verify the criteria we needed **8 test cases**

Tc1: passing a string with valid lengths as assertion expectation

Tc2: passing a string with valid lengths not as assertion expectation

Tc3: passing a string with valid lengths not as assertion expectation

Tc4: passing a string with non –valid lengths.

Tc5: passing a non-zero strings , “spaces , and newlines”

Tc6: passing an empty string

Tc7: passing a blank/ null String

Tc8: passing a zero length string

Base case(s): *Tc1, Tc2 , Tc3 with valid lengths*

Initial Test Requirements: *valid test cases, valid method’s class to test*

Infeasible Requirements:  *no valid test cases , or method to test*

Infeasible cases: *Tc6, Tc7, Tc8*

Result: **We got 75 % passed Tc’s 25 % failed Tc’s**

2-For CharAt() method:

to verify the criteria we needed **3 test cases**

Tc1: passing a valid index > 0

Tc2: passing not valid index < 0

Tc3: passing an index out of string boundaries index > str.length ()

Base case(s): *Tc1*

Initial Test Requirements: *valid test cases, valid method’s class to test*

Infeasible Requirements:  *no valid test cases, or method to test*

Infeasible cases: *Tc2, Tc3*

Result: **We got 100% Tc’s Passed**

3-For subsequence() method:

to verify the criteria we needed **5 test cases**

Tc1: passing valid start index 0<index<str.length (), valid end index 0<end index<str.length (), and valid string

Tc2: passing invalid start index< 0, valid end index 0<end index<str.length (), and valid string

Tc3: passing valid start index 0<start index<str.length() , invalid end index > str.length() , and valid string

Tc4: passing valid start index 0<start index<str.length() , valid end index 0<start index<str.length() , and invalid string “null ,empty”

Tc5: passing invalid start index start index > end index, invalid end index <start index, and valid string

Base case(s): *Tc1*

Initial Test Requirements: *valid test cases, valid method’s class to test*

Infeasible Requirements:  *no valid test cases, or method to test*

Infeasible cases: *Tc2, Tc3,Tc4,Tc5*

Result: **We got 100% Tc’s Passed**

4-For toString() method:

To verify the criteria we needed **two test cases**

Tc1: passing valid string

Tc2: passing invalid string “null, blank”

Base case(s): *Tc1*

Initial Test Requirements: *valid test cases, valid method’s class to test*

Infeasible Requirements:  *no valid test cases, or method to test*

Infeasible cases: *Tc2*

Result: **We got 100% Tc’s Passed**

5-ForCompare () method:

To verify the criteria we needed **two test cases**

Tc1: passing valid strings equal each other’s in lengths

Tc2: passing valid strings the first> the second

Tc3: passing valid strings the first< the second

Base case(s): *Tc1, Tc2, Tc3*

Initial Test Requirements: *valid test cases, valid method’s class to test*

Infeasible Requirements:  *no valid test cases, or method to test*

Infeasible cases:  all of them are possible to get

Result: **We got 100% Tc’s Passed**

6-For Intstream codepoints() method:

To verify the criteria we needed **two test cases**

Tc1: passing valid string

Tc2: passing invalid string “null, blank”

Base case(s): *Tc1,Tc2*

Initial Test Requirements: *valid test cases, valid method’s class to test*

Infeasible Requirements:  *no valid test cases, or method to test*

Infeasible cases: *all of them are possible to get*

Result: **We got 100% Tc’s Passed**

7-For Intstream Chars() method:

To verify the criteria we needed **two test cases**

Tc1: passing valid string

Tc2: passing invalid string “null, blank”

Base case(s): *Tc1,Tc2*

Initial Test Requirements: *valid test cases, valid method’s class to test*

Infeasible Requirements:  *no valid test cases, or method to test*

Infeasible cases: *all of them are possible to get*

Result: **We got 100% Tc’s Passed**