David Hazall-Farrell

CSL-131

3/4/2024

Lab #6

5.

a. 5

b. 0

c. Error

Exception in thread "main" java.lang.NullPointerException: Cannot invoke "String.length()" because "empty3" is null

at StringExperiments.main(StringExperiments.java:19)

d. 1

e. 1

f. Error

Exception in thread "main" java.lang.Error: Unresolved compilation problem:

The local variable nothing may not have been initialized

at StringExperiments.main(StringExperiments.java:19)

8.

Scanner console = new Scanner(System.***in***);

String name;

System.***out***.print("Enter name: ");

name = console.nextLine();

System.***out***.println(name);

11.

Give me a sentence.

I really like to prance and dance.

The sentence, I really like to prance and dance, contains 6 spaces.

12.

System.***out***.println("Give me a sentence.");

String sentence;

sentence = console.nextLine();

int numSpaces = 0;

for(int i = 0; i <= sentence.length() - 1; i++) {

if(sentence.charAt(i) == ' ') {

numSpaces++;

}

}

System.***out***.println("The sentence, " + sentence + ", contains " + numSpaces + " spaces.");

18.

Password to test: HelloWorld35

4/5 rating.

Password to test: badpass

1/5 rating.

Password to test: AmazingPassword8!

5/5 rating.

19. This program requires you to input 2 identical passwords. It assesses how strong the password is on the right-hand side. If you click the button that says “Show passwords in clear” it shows the passwords. Then you can (hypothetically) set your password.

20.

a. An advantage of not checking the box is privacy. Potential threats can come from making your password visible to see.

b. The advantage is to double check your password to see if there is an error. Also, by making sure you haven’t accidentally typed in the wrong password twice, making it impossible to remember the correct password.

c. I don’t think it’s necessary. If you don’t remember what you’ve typed or think you typed it wrong, you can always start over. It’s better than having your information stolen.

21.

import java.util.Scanner;

/\*\*

\* PasswordUtils contains a bunch of methods that are useful in determining

\* the strength of passwords.

\* A main method is provided to exercise (and debug)the methods.

\*

\* **@author** <Your name here>

\*

\*/

public class PasswordUtils

{

public static void main(String[] args)

{

Scanner sc = new Scanner(System.***in***);

System.***out***.print("Password to test: ");

String pw = sc.nextLine();

System.***out***.println(*score*(pw) + "/5 rating.");

/\*

if (containsUpperCase(pw))

{

System.out.println(" ...contains an upper case letter");

}

else if (containsLowerCase(pw))

{

System.out.println(" ...contains a lower case letter");

}

else if (containsDigit(pw))

{

System.out.println(" ...contains a digit");

}

else if (containsSpecialChar(pw))

{

System.out.println(" ...contains a special character");

}

\*/

}

/\*\*

\* Determine if the given string contains an upper case letter

\* **@param** s the string to check

\* **@return** true if and only if s contains an upper case letter

\*/

public static boolean containsUpperCase(String s)

{

for (int letter = 0; letter < s.length(); letter++)

{

if ("ABCDEFGHIJKLMNOPQRSTUVWXYZ".indexOf(s.charAt(letter)) >= 0)

{

return true;

}

}

return false;

}

//Determines if the password has a lower case letter

public static boolean containsLowerCase(String s)

{

for (int letter = 0; letter < s.length(); letter++)

{

if ("abcdefghijklmnopqrstuvwxyz".indexOf(s.charAt(letter)) >= 0)

{

return true;

}

}

return false;

}

//determines if the password has a digit

public static boolean containsDigit(String s)

{

for (int letter = 0; letter < s.length(); letter++)

{

if ("0123456789".indexOf(s.charAt(letter)) >= 0)

{

return true;

}

}

return false;

}

//determines if the password has a special character

public static boolean containsSpecialChar(String s)

{

for (int letter = 0; letter < s.length(); letter++)

{

if ("[!@#$%^&\*()\_+{}:".indexOf(s.charAt(letter](#$%^&*()_+{}:".indexOf(s.charAt(letter))) >= 0)

{

return true;

}

}

return false;

}

public static boolean longEnough(String s)

{

for (int letter = 0; letter < s.length(); letter++)

{

if (letter >= 8)

{

return true;

}

}

return false;

}

/\*\*

\* Determine the actual strength of a password based upon various tests

\* **@param** s the password to evaluate

\* **@return** the strength (on a 1 to 5 scale, 5 is very good) of the password

\*/

public static int score(String s) {

int rating = 0;

if(*containsLowerCase*(s) == true) {

rating++;

}

if(*containsUpperCase*(s) == true) {

rating++;

}

if(*containsDigit*(s) == true) {

rating++;

}

if(*containsSpecialChar*(s) == true) {

rating++;

}

if(*longEnough*(s) == true) {

rating++;

}

return rating;

}

}