ALGORITHMS & DATA STRUCTURES

MODIFY STRİNGS

Aydın Güler

Absurd class has lots of methods and properties. I designed the class structure as requested. After a lot of searching the Internet efficiently and effectively, I solved my problems.

I used Netbeans IDE 8.2 as working environment.

Firstly, I created a main class. In the same packet, I also created a class and named it Absurd. In the main class I just created an object of Absurd. It is Absurd test = new Absurd();. I did nothing else in this class. Next, I wrote my whole code into Absurd class. Then I imported the necessary classes of io and util packages. In Absurd class, I created a private Instance variable named MyString in order to store its state. After that, in order to get my stored string and use it, I wrote a method getMyString() to retrieve the MyString. Then, return the value of MyString to caller.

```
public class Absurd {
    private String MyString;
    private String getMyString(){
        return MyString;
    }
```

Then I created a constructor called Absurd. In this constructor, I defined a scanner object called scanUser so that I can get the user's input. I also defined a Boolean called breakTheWhileLoop to break the while loop.

```
public Absurd() {
    Scanner scanUser = new Scanner(System.in);
    boolean breakTheWhileLoop = false;
```

With this case I am changing the status of breakTheWhileLoop to true.

```
case ('x'):
    breakTheWhileLoop = true;
break;
```

With this statement I am checking If the breakTheWhileLoop is true. If it is true then it breaks the while loop.

```
    }
    if (breakTheWhileLoop) break;
}
```

Here, the while loop covers the whole switch block. I created a string called newString to read the user input. Switch block depends on newString's first char. Thus, I can select one of the case which contains many codes to be executed.

```
while(true){
    String newString = scanUser.next().toLowerCase();
    switch (newString.charAt(0)){
```

As I mentioned before, with x/X case the while loop breaks and it terminates the program. With s/S case, user can enter the string and the GetString method takes the input. Then the MyString takes the result.

The commands are:

s or S The user can enter the string

d/D n the n'th character is deleted from the string and the result is shown

p/P Program check whetheer the sring is palindrome or not

r/R The string will be reversed and result is shown

i/I n <String> The <String> will be inserted after n'th position of the string and the result is shown

m/M The string will be capitilized and result is shown

o/O output string to screen

t/T<Pathname\FileName> The string is read from the file

w/W<Pathname\FileName> The string is written to the file

f/F Program reverse the words and result is shown

x/X Terminates the program

Most of these commands are executed by methods. Now, I will explain some of them.

Below, We see the GetString method takes the user input and returns it with userInput.

```
public String GetString(Scanner userInput){
    String userString = userInput.nextLine().trim();
    return userString;
}
```

Here, The Palindrome method takes the string and checks it whether the string is palindrome or not with the help of for loop. Then it returns the Boolean result.

```
public boolean Palindrome(String isThatPalindrome) {
    int n = isThatPalindrome.length();
    for( int i = 0; i < n/2; i++ )
        if (isThatPalindrome.charAt(i) != isThatPalindrome.charAt(n-i-1)) {
            return false;
        }
    return true;
}
```

Delete method takes the string and a int value which will be the deleted string. It splits the string into two parts. Then it sum them up

```
public String Delete(String removeLetter, int k){
    String fitstPart = removeLetter.substring(0,k);
    String secondPart = removeLetter.substring(k+1,removeLetter.length());
    return fitstPart+secondPart;
}
```

OutReverse method takes the string and reverses it in characters. What does it do is simply to take the last character and makes it the new first character and so on.

```
public String OutReverse(String reverseLetters){
    int i, len = reverseLetters.length();
    StringBuilder dest = new StringBuilder(len);
    for (i = (len - 1); i >= 0; i--){
        dest.append(reverseLetters.charAt(i));
    }
    return dest.toString();
}
```

Here I used a method and called it WordReverse. I found it on a website. Here is the link:

https://www.geeksforgeeks.org/print-words-string-reverse-order/

```
public String WordReverse(String reverseWords){
    int i = reverseWords.length() - 1;
    int start, end = i + 1;
    String reversedString = "";
    while(i >= 0)
    {
        if(reverseWords.charAt(i) == ' ')
        {
            start = i + 1;
            while(start != end)
                reversedString += reverseWords.charAt(start++);
                reversedString += ' ';
                end = i;
        }
        i--;
    }
```

Insert method takes inputString to insert a character to in it. Int k points the place where character will be inserted. InsertX is the character that will be inserted. Like in Delete method it splits the string into two and inserts the new character.

```
public String Insert(String inputString, int k, String insertX){
    String insertedX = insertX;
    String partOne = inputString.substring(0,k);
    String partTwo = inputString.substring(k,inputString.length());
    return partOne+insertedX+partTwo;
}
```

MakeCapitalize uses .toUpperCase to capitalize the string.

```
public String Output(String getScreen){
    String showOutput = getScreen;
    return showOutput;
}
```

Output returns the string.

```
public String MakeCapitalize(String makeCapitalizedMyString){
    String upperMyString = makeCapitalizedMyString.toUpperCase();
    return upperMyString;
}
```

InFile takes the string and writes it into a file in a specific location.

```
public String InFile(String fileDirection2) throws IOException{
    String content = new String(Files.readAllBytes(Paths.get(fileDirection2)), StandardCharsets.UTF_8);
return content;
}
```

PrintString gets the string with getMyString and prints it. Then it returns the MyString.

```
private String PrintString()
{
    System.out.println(""+getMyString());
    return MyString;
}
```

Testing the Program:

```
run:
s Test this string.
Test this string.
d 2
Tst this string.
i 2 e
Test this string.
r
.gnirts siht tseT
r
Test this string.
f
string. this Test
f
Test this string.
p
The String is not a palindrome
m
TEST THIS STRING.
t d:\Myfile.txt
The file has been opened and the string has been read
o
This is new string.
m
THIS IS NEW STRING.
x
The Program has terminated
BUILD SUCCESSFUL (total time: 4 minutes 18 seconds)
```