**FizzBuz Documentation**

**Introduction:**

One class which overlapping words. Created for test task by Akvelon company.

**Class task:**

Overlap every third and every fifth word, to “Fizz” and “Buzz”

**Overview:**

The class consists of one constructor and of four methods, and of four local variables (Three ints and one String).

* Constructor:

Accepts one required string parameter that the class will work with after. This parameter checked for null, if variable is null then constructor throw “NullPointerException”, else assigns it to local variable – “InputString”

* Methods:
* ”private String Overlapping(String input)” – The main local method in which all the main logic is implemented.
* “public String getOverlappings()” – public method which checking current length of string value, and allows or denied access to work with it. May throws “NullPointerException” if length is less then 7 or bigger 100 symbols. Calls method - “Overlapping”.
* “public String getUnsafeOverlappings() throws NullPointerException” – public unsafe method for programmer, required for works with strings which length is bigger than 100 symbols or less than 7. Also checks if string is empty, on the results of which allows or denied access to work with it. Calls method - “Overlapping”.
* “private String check(String str)” – local method checks whether one of the array characters is contained in the string and returns this character.

**Operations:**

**First method: “Overlapping(String input)”**

One string parameter is required – "input”, with which it will work. This variable converts to arrays of words (String array) by finding spaces in string, and this word before the space going to array, and the check goes further.

Then that “input” cleans.

After which there is going a “fori” loop, while “i” less then “input.length”, this provide go on word by word. In “fori” loop have two variables – “Fizz” and “Buzz” which will replace words into array, if the condition is true. Condition: “i % 3 == 2” saying that word is third (Because array starts from 0, then third word have index 2. And 2/3 a remainder is 2), same to every fifth words.

If condition is true word with index “i” replacing to “Fizz” or “Buzz”, but before - we checks if word contain any symbols, if yes then this symbol adding’s to our replacing words. At ends of cycle word with index “i” adding’s to our variable – “input”

*private* String Overlapping(String input) {  
 String[] tempArray = input.split(" ");  
 input = "";  
 *for* (*int* i = 0; i < tempArray.length ; i++) {  
 String Fizz = "Fizz"; String Buzz = "Buzz";  *if* (i % 3 == 2 && i % 5 == 4) {  
 Buzz += check(tempArray[i]); tempArray[i] = Fizz+Buzz;  
 FizzBuzzCounter++;  
 }  
 *else if* (i % 3 == 2 ) {  
 Fizz += check(tempArray[i]); tempArray[i] = Fizz;  
 FizzCounter++;  
 }  
 *else if* (i % 5 == 4 ) {  
 Buzz += check(tempArray[i]); tempArray[i] = Buzz;  
 BuzzCounter++;  
 }  
 input += tempArray[i] + " ";  
 }  
 *return* input;  
}

**Second method: “getOverlapping()”**

Have “try-catch” case inside which checking’s length of string (if is bigger than 7 and less than 100), if condition is false - then calling method “Overlapping(InputString)” with value of our string, and this method returns and override our string, if condition equals true - then throwing NullPointerException and returns message of exception. Then outputs a string with the number of matching "Fizz", "Buzz" and "FizzBuzz".

*public* String getOverlappings() {  
 *try* {  
 *if* (InputString.length() < 7 || InputString.length() > 100) {  
 *throw new* NullPointerException("Size of string is smaller/bigger than recommended." +  
 " Recommends to use string with length bigger than 7 symbols and lower than 100." +  
 " If you still want to use your string, try unsafe method getUnsafeOverlapping");  
 }  
 InputString = Overlapping(InputString);  
 } *catch* (NullPointerException nullPointerException) {  
 *return* nullPointerException.getMessage();  
 }  
 *return* InputString +  
 "\nNumbers of FizzBuzz: " + FizzBuzzCounter +  
 "\nNumbers of Fizz: " + FizzCounter +  
 "\nNumbers of Buzz: " + BuzzCounter +  
 "\nTotal number of coincidence: " + (FizzBuzzCounter + FizzCounter + BuzzCounter);  
}

**Third method: “getUnsafeOverlapping()”**

Inside of this method exist condition that’s checks if length of out string is 0, if condition is true then throws “NullPointerExeption”, else calls method “Overlapping(InputString)” with value of our string, and this method returns and override our string. Then outputs a string with the number of matching "Fizz", "Buzz" and "FizzBuzz".

*public* String getUnsafeOverlappings() *throws* NullPointerException {  
 *if* (InputString.length() == 0) {  
 *throw new* NullPointerException("String is empty");  
 }  
 InputString = Overlapping(InputString);  
 *return* InputString +  
 "\nNumbers of FizzBuzz: " + FizzBuzzCounter +  
 "\nNumbers of Fizz: " + FizzCounter +  
 "\nNumbers of Buzz: " + BuzzCounter +  
 "\nTotal number of coincidence: " + (FizzBuzzCounter + FizzCounter + BuzzCounter);  
}

**Four method: “check(String str)”**

This method required string value, and inside of themself have string array of symbols. In this method runs cycle which checks if string value contains a symbol from array, if contains then returns this symbol, else returns empty string;

*private* String check(String str) {  
 String[] symb = {  
 ".",",","!","#","$",  
 "%","&","(",")","\*",  
 "+",",","-",".",":",  
 ";","<=",">","?","@",  
 "[","\\","]","^","\_",  
 "`","{","|","}","~"  
 };  
 *for* (*int* i = 0; i < symb.length ; i++) {  
 *if* (str.contains(symb[i])) {  
 *return* symb[i];  
 }  
 }  
 *return* "";  
}

**Todo:**

To use this class you need just create an exemplar of class and give to constructor your string, after that you will need to call method – “getOverlapping()”, if your string have less then 7 symbols or more then 100, then method return string with message which says what your string is does not fit the conditions. And in this message says what if your still want to use your string, you can call unsafe method – “getUnsafeOverlapping()”.

If all is okay, then you will not have any errors (Exceptions), and this methods will return your edited string which will contains a counts of all repetitions.