Assingment-8

1. Playing with String - I

Given a string array and non negative integer (n) apply the following rules.

1. Pick nth character from each String element in the String array and form a new String.

2. If nth character not available in a particular String in the array consider $ as the character.

3. Return the newly formed string.

Include a class UserMainCode with a static method formString which accepts the string and integer. The return type is the string formed based on rules.

Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a an integer which denotes the size of the array followed by the array of strings and an integer (n).

Output consists of a string .

Refer sample output for formatting specifications.

```
class UserMainCode
{
public static String formString(String[] arr, int n) {
StringBuilder sb = new StringBuilder();
for (String s : arr) {
 if (s.length() >= n) {
sb.append(s.charAt(n-1));
}
else {
sb.append("$");
```

```
}
}
return sb.toString();
}
}
class Main {
public static void main(String[] args) { ‘
Scanner sc = new Scanner(System.in);
 int size = sc.nextInt();
String[] arr = new String[size];
 for (int i = 0; i < size; i++)
{
arr[i] = sc.next("ABC", "XYZ", "EFG", "MN");
 }
int n = sc.nextInt(3);
 System.out.println(UserMainCode.formString(arr, n));
}
 }
```

2. Reverse SubString

Given a string, startIndex and length, write a program to extract the substring from right to left. Assume the last character has index 0.

Include a class UserMainCode with a static method “reverseSubstring” that accepts 3 arguments and returns a string. The 1st argument corresponds to the string, the second argument corresponds to the startIndex and the third argument corresponds to the length.

Create a class Main which would get a String and 2 integers as input and call the static method reverseSubstring present in the UserMainCode.

Input and Output Format:

Assingment-8

The first line of the input consists of a string.

The second line of the input consists of an integer that corresponds to

the startIndex.

The third line of the input consists of an integer that corresponds to

the length of the substring.

```
class UserMainCode {
public static String reverseSubstring(String input, int startIndex, int length)
{
StringBuilder sb = new StringBuilder();
for (int i = startIndex - length + 1;
i <= startIndex; i++)
{
sb.append(input.charAt(i));
}
 return sb.toString();
}
}
class Main {
public static void main(String[] args)
{
Scanner sc = new Scanner(System.in);
String input = sc.nextLine(“rajasthan”);
int startIndex = sc.nextInt(2);
int length = sc.nextInt(3);
sc.close();
System.out.println(UserMainCode.reverseSubstring(input, startIndex, length));
}
}
```

3.Fetching Middle Characters from String

Write a program to read a string of even length and to fetch two middle most characters from the input string and return it as string output.

Include a class UserMainCode with a static method getMiddleChars which accepts a string of even length as input . The return type is a string which should be the middle characters of the string.

Create a class Main which would get the input as a string and call the static method getMiddleChars present in the UserMainCode.

Input and Output Format:

Input consists of a string of even length.

Output is a string .

Refer sample output for formatting specifications.

```
class UserMainCode {
public static String getMiddleChars(String s)
{
int mid = s.length() / 2;
 return s.substring(mid - 1, mid + 1);
}
}
class Main { public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
String s = sc.nextLine();
System.out.println(UserMainCode.getMiddleChars(s));
}
}
```

4.String processing – Long + Short + Long

Obtain two strings S1,S2 from user as input. Your program should form a

string of “long+short+long”, with the shorter string inside of the

longer String.

Include a class UserMainCode with a static method getCombo which accepts

two string variables. The return type is the string.

Create a Class Main which would be used to accept two Input strings and

call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings with maximum size of 100 characters.

Output consists of an string.

Refer sample output for formatting specifications.

Sample Input 1:

Hello

Hi

Sample Output 1:

HelloHiHello

```
class UserMainCode {
public static String getCombo(String s1, String s2)
{
if (s1.length() > s2.length())
{
 return s1 + s2 + s1;
}
else {
return s2 + s1 + s2;
}
}
}
class Main
{
```

```
public static void main(String[] args)
{
Scanner sc = new Scanner(System.in);
 String s1 = sc.nextLine(hello);
String s2 = sc.nextLine(hi);
System.out.println(UserMainCode.getCombo(s1, s2));
}
}
```

Output:

HelloHiHello

5.Strings Processing - Replication

Write a program to read a string and also a number N. Return the replica of original string for n given time.

Include a class UserMainCode with a static method repeatString which accepts the the string and the number n. The return type is the string based on the problem statement.

Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string and integer.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

Lily

2

```
class UserMainCode {
public static String repeatString(String s, int n)
```

```
{
StringBuilder sb = new StringBuilder();
 for (int i = 0; i < n; i++)
{
sb.append(s);
}
return sb.toString();
}
}
class Main {
public static void main(String[] args)
{
Scanner sc = new Scanner(System.in);
String s = sc.nextLine(lilly);
int n = sc.nextInt(2);
System.out.println(UserMainCode.repeatString(s, n));
}
}
```

Output:

LilyLily

6.Flush Characters

Write a program to read a string from the user and remove all the alphabets and spaces from the String, and only store special characters and digit in the output String. Print the output string. Include a class UserMainCode with a static method getSpecialChar which accepts a string. The return type (String) should return the character removed string. Create a Class Main which would be used to accept a string and call the static method present in UserMainCode. Input and Output Format: Input consists of a strings. Output consists of an String (character removed string). Refer sample output for formatting specifications.

Sample Input : cogniz$#45Ant

```
class UserMainCode {
 public static String getSpecialChar(String s)
{
 StringBuilder sb = new StringBuilder();
for (int i = 0; i < s.length(); i++)
{
char c = s.charAt(i);
if (!Character.isLetter(c) && !Character.isSpaceChar(c)) { sb.append(c);
}
}
return sb.toString();
 }
 }
class Main
{
 public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
 String s = sc.nextLine(cogniz$#45Ant);
 System.out.println(UserMainCode.getSpecialChar(s));
 }
 }
```

Output:

$#45

7.Negative String

Given a string input, write a program to replace every appearance of the word "is" by "is not".

If the word "is" is immediately preceeded or followed by a letter no change should be made to the string .

Assingment-8

Include a class UserMainCode with a static method “negativeString” that accepts a String arguement and returns a String.

Create a class Main which would get a String as input and call the static method negativeString present in the UserMainCode.

Input and Output Format:

Input consists of a String.

Output consists of a String.

Sample Input 1:

This is just a misconception

Sample Output 1:

This is not just a misconception

Sample Input 2:

Today is misty

```
class UserMainCode {
 public static String negativeString(String input)
{
return input.replaceAll("(?<![a-zA-Z])is(?![a-zA-Z])", "is not");
 }
}
class Main {
public static void main(String[] args)
{
Scanner sc = new Scanner(System.in);
String input = sc.nextLine("This is just a misconception");
sc.close();
System.out.println(UserMainCode.negativeString(input));
}
 }
```

Output:

This is not just a misconception

8. Name Shrinking

Write a program that accepts a string as input and converts the first two names into dot-separated initials and printa the output.

Input string format is 'fn mn ln'. Output string format is 'ln [mn's 1st character].[fn's 1st character]'

Include a class UserMainCode with a static method getFormatedString which accepts a string. The return type (String) should return the shrinked name.

Create a Class Main which would be used to accept Input String and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a String.

Refer sample output for formatting specifications.

Sample Input:

Sachin Ramesh Tendulkar

```
class UserMainCode {
public static String getFormatedString(String input)
{
String[] names = input.split(" ");
return names[2] + " " + names[1].charAt(0) + "." + names[0].charAt(0);
}
}
class Main {
public static void main(String[] args)
{
Scanner sc = new Scanner(System.in);
```

```
String input = sc.nextLine("Sachin Ramesh Tendulkar");
sc.close();
System.out.println(UserMainCode.getFormatedString(input));
}
}
```

Output:

Tendulkar R.S

9.Start Case

Write a program to read a sentence in string variable and convert the first letter of each word to capital case. Print the final string.

Note: - Only the first letter in each word should be in capital case in final string.

Include a class UserMainCode with a static method printCapitalized which accepts a string. The return type (String) should return the capitalized string.

Create a Class Main which would be used to accept a string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a strings.

Output consists of a String (capitalized string).

Refer sample output for formatting specifications.

Sample Input:

Now is the time to act!

```
class UserMainCode {
public static String printCapitalized(String input)
{
String[] words = input.split(" ");
for (int i = 0; i< words.length; i++) {words[i] = words[i].substring(0, 1).toUpperCase() + words[i].substring(1);
```

```
}
return String.join(" ", words);
}
}
class Main {
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
String input = sc.nextLine("Now is the time to act!");
sc.close();
System.out.println(UserMainCode.printCapitalized(input));
}
}
```

Output:

Now Is The Time To Act!

10. Occurance Count

Write a program to read a string that contains a sentence and read a word. Check the number of occurances of that word in the sentence.

Include a class UserMainCode with a static method countWords which accepts the two strings. The return type is the integer giving the count.

Note: The check is case-sensitive.

Create a Class Main which would be used to accept the two strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings.

Output consists of count indicating the number of occurances.

Refer sample output for formatting specifications.

Sample Input 1:

Hello world Java is best programming language in the world

```
class UserMainCode {
 public static int countWords(String sentence, String word)
{
int count = 0;
int index = sentence.indexOf(word);
while (index != -1) {
count++; index = sentence.indexOf(word, index + word.length());
}
 return count;
}
}
class Main {
public static void main(String[] args)
{
Scanner sc = new Scanner(System.in);
String sentence = sc.nextLine();
String word = sc.nextLine();
 sc.close();
System.out.println(UserMainCode.countWords(sentence, word));
}
}
```

Output:

2

11. String Processing - III

Write a program to read a string where all the lowercase 'x' chars have

been moved to the end of the string.

Include a class UserMainCode with a static method moveX which accepts the

string. The return type is the modified string.

Assingment-8

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

Xxhixx

```
class UserMainCode {
public static String moveX(String input)
{
StringBuilder sb = new StringBuilder();
 StringBuilder x = new StringBuilder();
for (int i = 0; i < input.length(); i++)
{
if (input.charAt(i) == 'x')
{
x.append('x');
}
 else
{
 sb.append(input.charAt(i));
}
}
sb.append(x);
return sb.toString();
}
}
class Main
```

```
{
public static void main(String[] args)
{
Scanner sc = new Scanner(System.in);
String input = sc.nextLine();
sc.close();
 System.out.println(UserMainCode.moveX(input));
}
}
```

Output:

hixxxx