ASSIGNMENT-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. Sample Input 1: 4 ABC XYZ EFG MN 3 Sample Output1: CZG$

**import** java.util.\*;

**public** **class** StringCharacter {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

}

} **class** usermaincode1{

String formString() {

String ans="";

System.***out***.println("enter string integer:");

Scanner sc = **new** Scanner(System.***in***); **int** len= sc.nextInt(); String[] array=**new** String[len]; **for**(**int** i=0;i<len;i++) {

System.***out***.println(array[i]=sc.next());

}

**int** num=sc.nextInt();

**return** ans;

}

}

## 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: 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. Sample Input: rajasthan 2 3 Sample Output: hts package Stringpack;

ANS **import** java.util.\*;

**public** **class** ReverseString {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub Scanner sc =**new** Scanner(System.***in***);

System.***out***.println("enter string");

String s =sc.next();

System.***out***.println("enter length "); **int** len =sc.nextInt();

System.***out***.println("enter position");

**int** pos=sc.nextInt(); usermaincode2 umc =**new** usermaincode2(); String ans=umc.reverseSubString(s,pos,len);

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

}

} **class** usermaincode2{

**public** String reverseSubString(String s,**int** len,**int** pos) {

StringBuilder input=**new** StringBuilder(); // append a string into StringBuilder input1 input.append(s);

input.reverse();

String ans=input.substring(pos,pos+len);

**return** ans;

}

}

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. Sample Input 1: this Sample Output 1: Hi

ANS

**import** java.util.Scanner;

**public** **class** EvenMiddleLetters {

**public** **static** **void** main(String[] args) {

|  |
| --- |
| String |
| String |

// **TODO** Auto-generated method stub Scanner sc = **new** Scanner(System.***in***); s = sc.nextLine();

s1 = UserMainCode3.*getMiddleChars*(s);

System.***out***.println(s1);

}

**class** UserMainCode3 {

**public** **static** String getMiddleChars(String str) {

**int** index, length;

**if** (str.length() % 2 == 0) { index = str.length() / 2 - 1;

length = 2;

}

**else** {

index = str.length() / 2;

length = 1;

}

**return** str.substring(index, index + length);

}

}

}

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 package Stringpack;

**import** java.util.Scanner;

**public** **class** StringAddition {

**public** **static** **void** main(String[] args) { // **TODO** Auto-generated method stub

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

String s1,s2;

System.***out***.println("Enter 1st String:");

s1=sc.next();

System.***out***.println("Enter 2nd String:"); s2=sc.next();

**if**(s1.length()>s2.length()) {

System.***out***.println(s1+s2+s1);

}

**else** {

System.***out***.println(s2+s1+s2);

}

}

}

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

Sample Output 1: LilyLily package Stringpack; import java.util.Scanner;

ANS **import** java.util.Scanner;

**public** **class** StringReplication {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

String s; **int** i;

Scanner sc =**new** Scanner(System.***in***); System.***out***.println("Enter a string");

s=sc.next();

System.***out***.println("Enter the number how many times u want the string");

i=sc.nextInt();

StringBuffer sb = **new** StringBuffer();

**for**(**int** j=0; j<i;j++) { sb.append(s);

}

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

}

}

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 Sample Output : $#45

**import** java.util.\*; **public** **class** FlushCharacter {

**public** **static** **void** main(String[] args) {

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

String s1=in.nextLine();

System.***out***.println(UserMainCode.*getSpecialChar*(s1));

in.close();

}

}

**class** UserMainCode{

**public** **static** String getSpecialChar(String s1){ **int** x=s1.length();

StringBuffer sb=**new** StringBuffer();

**for**(**int** i=0;i<x;i++){ **char** c=s1.charAt(i);

**if**(!Character.*isAlphabetic*(c))

sb.append(c);

}

**return** sb.toString();

}

}

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 . 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

Sample Output 2: Today is not misty

**import** java.util.\*; **public** **class** NegativeString {

**public** **static** **void** main(String[] args) {

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

System.***out***.println("Enter the String:");

String s=scanner.nextLine();

String ans=UserMainCodes.*negativeString*(s);

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

}

**class** UserMainCodes{

**public** **static** String negativeString(String s) { String newstring = ""; **int** l =

s.length();

**for**(**int** i = 0; i < l; i++)

{

**if**(i-1 >= 0 && Character.*isLetter*(s.charAt(i-1))||

i+2 < l && Character.*isLetter*(s.charAt(i+2)))

{

newstring += s.charAt(i);

**continue**;

}

**else** **if**(i+1 < l && s.substring(i, i+2).equals("is"))

{

newstring += "is not";

i++; } **else**

newstring += s.charAt(i);

}

**return** newstring;

}

}

}

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 Sample Output: Tendulkar R.S package Stringpack

## 

**import** java.util.\*;

**public** **class** NameShrinking {

**public** **static** **void** main(String[] args) {

//**TODO** Auto-generated method stub

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

String s1 = sc.nextLine();

System.***out***.println(UserMain.*getFrmatedString*(s1));

}

} **class** UserMain{

**public** **static** String getFrmatedString(String s1) { StringBuffer sb = **new** StringBuffer();

StringTokenizer st = **new** StringTokenizer(s1, " ");

String s2 = st.nextToken(); String s3 =

st.nextToken(); String s4 = st.nextToken(); sb.append(s4).append(" ");

sb.append(s3.substring(0, 1)); sb.append("."); sb.append(s2.substring(0, 1));

System.***out***.println(sb); **return** s1.toString();

}

}

## Q9.start case ANS

**import** java.util.\*; **public** **class** FirstLetterCapital {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

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

String s1= sc.nextLine();

System.***out***.println(UserMains.*printCapitalized*(s1));

}

} **class** UserMains{

**public** **static** String printCapitalized(String s1){

StringBuffer sb=**new** StringBuffer(); StringTokenizer t=**new** StringTokenizer(s1," "); **while**(t.hasMoreTokens())

|  |
| --- |
| String |
| String |
| String |

{ s2=t.nextToken(); s3=s2.substring(0,1); s4=s2.substring(1, s2.length());

sb.append(s3.toUpperCase()).append(s4).append(" ");

} **return** sb.toString();

}

}

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 world Sample Output 1: 2 Sample Input 2: hello world World Sample Output 2:0

**import** java.util.\*; **public** **class** OccuranceCount {

**public** **static** **void** main(String[] args) {

User sim = **new** User(); **int** y = sim.*wordCount*();

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

}

} **class** User{ **public** **static** **int** wordCount() { Scanner sc = **new** Scanner(System.***in***); **int** count = 0;

System.***out***.println("Enter a sentence");

String s = sc.nextLine();

System.***out***.println("please type the word for which you want to perform wordCount");

String s1 = sc.nextLine();

String s3 = s.toLowerCase();

String[] words =s.split(" ");

System.***out***.println("count of word you entered is:"); **for**(**int** i=0;i<words.length;i++) { **if**(words[i].equals(s1)) { count++;

}

} **return** count;

}

}

## Q11 string Ans

**import** java.util.\*;

**public** **class** StringArrangement {

**public** **static** **void** main(String[] args) {

//**TODO** Auto-generated method stub

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

String s = sc.next();

System.***out***.println(flush.*moveX*(s));

}

} **class** flush{

**public** **static** String moveX(String s) { String str=**new** String();

String s1 = s.replaceAll("[x]", "");

String s2 = s.replaceAll("[^x]", ""); System.***out***.println(s1 + s2); **return** s1;

}

}