```
Program : 1
Write a method to find the maximum occurring character in a given String
with frequency.
Method name : maxOccurance
Argument : String
Return Type : String
Example1
input String : "javaaavvaba"
output : a = 6 time
Example2
input String : "javav"
output : a = 2 time
         v = 2 time
Example3
input String : "abc"
output : a = 1 time
         b = 1 time
         c = 1 time
NOTE : If input String is null or empty then return "invalid String
input".
```

```
Program : 2
Write a method to remove all duplicates character from a given string.
Method name : removeDuplicateCharacters
Argument : String
Return Type : String
Example
input String : "javaaavvaba"
output : "javb"
NOTE : If input String is null or empty then return "invalid String
input".
```

```
Program : 3
Write a method to print the duplicate characters from the given String in
sorted order.
Method name : returnDuplicateCharacters
Argument : String
Return Type : String
Example
input String : "javavaavvaba"
output : "aaaaaavvvv"
NOTE : If input String is null or empty then return "invalid String
input".
```

```
Program : 4
Write a method to replace all duplicate characters from the given String
by given character.
Method name : replaceDuplicateCharacters
Argument : String , character
Return Type : String
Example
input String : "javAaavvaba"
input given character : 'x'
output :  "jxxAxxxxxbx"
NOTE : If input String is null or empty then return "invalid String
input".
```

Program : 5
Write a method to replace specific characters from the given String by specific String.
Method name : replaceCharactersByString
Argument : String , character
Return Type : String
Example
input String : "javaaavvaba"
input specific character : 'v'
input specific String : "NareshIt"
output : jaNareshItaaaNareshItNareshItaba
Note : If input String is null or empty then return "invalid String input".
       If input character is not present in given input String then return "invalid input character".

Program : 6
Write a method to remove characters from the first String which are present in the second String.
Method name : removeCharacters
Argument : String,String
Return Type : String
Example
first String : "India is great"
second String : "In"
output : "da s great"
NOTE : If any input String is null or empty then return "invalid String input".
       If second input string characters not present in first input String then return "second input String is invalid".

Program : 7
Write a method to check if two strings are rotations of each other.
Method name : rotationsOfString
Argument : String, String
Return Type : boolean
Example1
input String1: "XYZ"
input String2: "ZXY"
output : true
Example2
input String1: "XYZ"
input String2: "YXZ"
output : false
NOTE : If any input String is null or empty then return "invalid String input".

Program : 8
Write a method to swap first and last character of string without using loop.
Method name : swapFirstAndLastCharacter
Argument : String
Return Type : String
Example1
input String: "Naresh It"
output : "taresh iN"

```
Example2
input String: "Welcome"
output : "eelcomW"
NOTE : If input String is null or empty then return "invalid String
input".


Program : 9
Write a method to swap character of string based on index numbers.
Method name : swapCharacterByIndex
Argument : String , int , int
Return Type : String
Example1
input String : "Naresh It"
input index1 : 3
input index2 : 0
output : "earNsh It"
Example2
input String : "Naresh It"
input index1 : 2
input index2 : 6
output : "Na eshrIt"
NOTE : If input String is null or empty then return "invalid String
input".
       If input index greater than String length then return "Index Out
Of Bound"
       If any input index is less then 0 then return "Invalid index"


Program : 10
Write a method to find the smallest substring in a given string
containing all characters of another string.
Method name : smallestSubstring
Argument : String , String
Return Type : String
Example
input String1 : "this is a test string"
input String1 : "tist"
output : "t stri"
NOTE : If any input String is null or empty then return "invalid String
input".


Program : 11
Write an efficient program to print all permutations of a given String in
Java your choice.
Method name : permutation
Argument : String
Return Type : String
Example1
input : "123"
output : "123", "132", "213", "231", "312" , "321"
Example2
input : "XYZ"
output : "XYZ, XZY, YXZ, YZX, ZXY, XYX"
NOTE : If input String is null or empty then return "invalid String
input".


Program : 12
```

Write a method to print First non-repeated character from given input String.
Method name : firstNonRepeatedCharacter
Argument : String
Return Type : char
Example1 : "Naresh it"
output : 'N'
Example2 : "Software Services"
output : 'o'
NOTE : If input String is null or empty then return "invalid String input".

Program : 13
wtrite a method to print Last non-repeated character from given input String.
Method name : lastNonRepeatedCharacter
Argument : String
Return Type : char
Example1 : "Naresh it"
output : 't'
Example2 : "Software Services"
output : 'c'
NOTE : If input String is null or empty then return "invalid String input".

Program : 14
Write an efficient method in Java to remove all occurrences of a given character in String.
Method name : removeCharacter
Argument : String
Return Type : String
input : "Programming"
given character : 'm'
output : "Prograing"
NOTE : If input String is null or empty then return "invalid String input".

Program : 15
Write a program to print Highest occured character from given String. if more than one characters having same count then also print first highest occured character only.
Method name : highestOccuredCharacter
Argument : String
Return Type : char
Example1 :
input : "aaaaaabbcccdd"
output : 'a'
Example2 :
input : "aaaaaabbcccdddddd"
output : 'a'
NOTE : If input String is null or empty then return "invalid String input".

Program : 16
Write a program to count the characters, digit, special characters from the given string.

Method name : countCharacter
Argument : String
Return Type : String
Example :
input : "#NareshIt@123#"
output :
character : 8
digit : 3
special character : 3
NOTE : If input String is null or empty then return "invalid String input".

Program : 17
Write a program to find the the percentage of Characters,digits,special characters from the given String.
Method name : percentageOfCharacters
Argument : String
Return Type : String
Example :
input : "Nacre@123%"
output :
character : 50.00%
digit : 30.00%
special character : 20.00%
NOTE : If input String is null or empty then return "invalid String input".

Program : 18
Write a program to check if a given string contains valid parentheses or not.
Given a string containing just the characters '(', ')', '{', '}', '[' and ']',write a function in Java to check if the input string is valid. The brackets must close in the correct order, "()" and "()[]{}" are all valid but "(]" and "([)]" are not.
Method name : validParentheses
Argument : String
Return Type : boolean
NOTE : If input String is null or empty then return "invalid String input".

Program : 19
Given an "out" string length 4, such as "<<>>", and a word, return a new string where the word is in the middle of the out string, e.g. "<<word>>". Note: use str.substring(i, j) to extract the String starting at index i and going up to but not including index j.
Method name : makeOutWord
Argument : String , String
Return Type : String
Example1 :
input : "<<>>", "NareshIt"
output : "<<NareshIt>>"
Example2 :
input : "<<>>", "WooHoo"
output : "<<WooHoo>>"
Example3 :
input : "[[]]", "word"
output : "[[word]]"

NOTE : If any input String is null or empty then return "invalid String input".

Program : 20
Given 2 strings, return their concatenation, except omit the first char of each. The strings will be at least length 1.
Method name : nonStart
Argument : String , String
Return Type : String
Example1 :
input : "Naresh", "Technologies"
output : "areshechnologies"
Example2 :
input : "java", "code"
output : "avaode"
Example2 :
input : "shotl", "java"
output : "hotlava"
NOTE : If any input String is null or empty then return "invalid String input".

Program : 21
Given a string of even length, return the first half. So the string "WooHoo" yields "Woo".

```
firstHalf("WooHoo") ? "Woo"
firstHalf("HelloThere") ? "Hello"
firstHalf("abcdef") ? "abc
```

Program : 22
Given a string, return a string length 1 from its front, unless front is false, in which case return a string length 1 from its back. The string will be non-empty.

```
theEnd("Hello", true) ? "H"
theEnd("Hello", false) ? "o"
theEnd("oh", true) ? "o"
```

Program : 23
Given a string, return true if it ends in "ly".

```
endsLy("oddly") ? true
endsLy("y") ? false
endsLy("oddy") ? false
```

Program : 24
Given a string of odd length, return the string length 3 from its middle, so "Candy" yields "and". The string length will be at least 3.

```
middleThree("Candy") ? "and"
middleThree("and") ? "and"
middleThree("solving") ? "lvi"
```

Program : 25

Given 2 strings, a and b, return a new string made of the first char of a and the last char of b, so "yo" and "java" yields "ya". If either string is length 0, use '@' for its missing char.

```
lastChars("last", "chars") ? "ls"
lastChars("yo", "java") ? "ya"
lastChars("hi", "") ? "h@"
```

Program : 26
Given a string, if the string begins with "red" or "blue" return that color string, otherwise return the empty string.

```
seeColor("redxx") ? "red"
seeColor("xxred") ? ""
seeColor("blueTimes") ? "blue"
```

Program : 27
Given a string, return a new string made of 3 copies of the first 2 chars of the original string. The string may be any length. If there are fewer than 2 chars, use whatever is there.

```
extraFront("Hello") ? "HeHeHe"
extraFront("ab") ? "ababab"
extraFront("H") ? "HHH"
```

Program : 28
Given a string and a second "word" string, we'll say that the word matches the string if it appears at the front of the string, except its first char does not need to match exactly. On a match, return the front of the string, or otherwise return the empty string. So, so with the string "hippo" the word "hi" returns "hi" and "xip" returns "hip". The word will be at least length 1.

```
startWord("hippo", "hi") ? "hi"
startWord("hippo", "xip") ? "hip"
startWord("hippo", "i") ? "h"
```

Program : 29
Given two strings, a and b, return the result of putting them together in the order abba, e.g. "Hi" and "Bye" returns "HiByeByeHi".

```
makeAbba("Hi", "Bye") ? "HiByeByeHi"
makeAbba("Yo", "Alice") ? "YoAliceAliceYo"
makeAbba("What", "Up") ? "WhatUpUpWhat"
```

Program : 30
Given a string, return a new string made of 3 copies of the last 2 chars of the original string. The string length will be at least 2.

```
extraEnd("Hello") ? "lololo"
extraEnd("ab") ? "ababab"
extraEnd("Hi") ? "HiHiHi"
```

Program : 31

Given a string, return a version without the first and last char, so "Hello" yields "ell". The string length will be at least 2.

```
withoutEnd("Hello") ? "ell"
withoutEnd("java") ? "av"
withoutEnd("coding") ? "odin"
```

Program : 32
Given a string, return a "rotated left 2" version where the first 2 chars are moved to the end. The string length will be at least 2.

```
left2("Hello") ? "lloHe"
left2("java") ? "vaja"
left2("Hi") ? "Hi"
```

Program : 33
Given a string, return a version without both the first and last char of the string. The string may be any length, including 0.

```
withouEnd2("Hello") ? "ell"
withouEnd2("abc") ? "b"
withouEnd2("ab") ? ""
```

Program : 34
Given a string and an int n, return a string made of the first and last n chars from the string. The string length will be at least n.

```
nTwice("Hello", 2) ? "Helo"
nTwice("Chocolate", 3) ? "Choate"
nTwice("Chocolate", 1) ? "Ce"
```

Program : 35
Given a string, return true if "bad" appears starting at index 0 or 1 in the string, such as with "badxxx" or "xbadxx" but not "xxbadxx". The string may be any length, including 0. Note: use .equals() to compare 2 strings.

```
hasBad("badxx") ? true
hasBad("xbadxx") ? true
hasBad("xxbadxx") ? false
```

Program : 36
Given two strings, append them together (known as "concatenation") and return the result. However, if the concatenation creates a double-char, then omit one of the chars, so "abc" and "cat" yields "abcat".

```
conCat("abc", "cat") ? "abcat"
conCat("dog", "cat") ? "dogcat"
conCat("abc", "") ? "abc"
```

Program : 37
Given a string, return true if the first 2 chars in the string also appear at the end of the string, such as with "edited".

```
frontAgain("edited") ? true
frontAgain("edit") ? false
frontAgain("ed") ? true
```

Program : 38
Given a string, if the first or last chars are 'x', return the string without those 'x' chars, and otherwise return the string unchanged.

```
withoutX("xHix") ? "Hi"
withoutX("xHi") ? "Hi"
withoutX("Hxix") ? "Hxi"
```

Program : 39
The web is built with HTML strings like "<i>Yay</i>" which draws Yay as italic text. In this example, the "i" tag makes <i> and </i> which surround the word "Yay". Given tag and word strings, create the HTML string with tags around the word, e.g. "<i>Yay</i>".

```
makeTags("i", "Yay") ? "<i>Yay</i>"
makeTags("i", "Hello") ? "<i>Hello</i>"
makeTags("cite", "Yay") ? "<cite>Yay</cite>"
```

Program : 40
Given a string and an index, return a string length 2 starting at the given index. If the index is too big or too small to define a string length 2, use the first 2 chars. The string length will be at least 2.

```
twoChar("java", 0) ? "ja"
twoChar("java", 2) ? "va"
twoChar("java", 3) ? "ja"
```

Program : 41
Given a string of any length, return a new string where the last 2 chars, if present, are swapped, so "coding" yields "codign".

```
lastTwo("coding") ? "codign"
lastTwo("cat") ? "cta"
lastTwo("ab") ? "ba"
```

Program : 42
Given two strings, append them together (known as "concatenation") and return the result. However, if the strings are different lengths, omit chars from the longer string so it is the same length as the shorter string. So "Hello" and "Hi" yield "loHi". The strings may be any length.

```
minCat("Hello", "Hi") ? "loHi"
minCat("Hello", "java") ? "ellojava"
minCat("java", "Hello") ? "javaello"
```

Program : 43
Given a string, return a string where for every char in the original, there are two chars.

```
doubleChar("The") ? "TThhee"
doubleChar("AAbb") ? "AAAAbbbb"
doubleChar("Hi-There") ? "HHii--TThheerree"
```

Program : 44
Return the number of times that the string "code" appears anywhere in the given string, except we'll accept any letter for the 'd', so "cope" and "cooe" count.

```
countCode("aaacodebbb") ? 1
countCode("codexxcode") ? 2
countCode("cozexxcope") ? 2
```

Program : 45
Return true if the given string contains a "bob" string, but where the middle 'o' char can be any char.

```
bobThere("abcbob") ? true
bobThere("b9b") ? true
bobThere("bac") ? false
```

Program : 46
Given a string and an int n, return a string made of n repetitions of the last n characters of the string. You may assume that n is between 0 and the length of the string, inclusive.

```
repeatEnd("Hello", 3) ? "llollollo"
repeatEnd("Hello", 2) ? "lolo"
repeatEnd("Hello", 1) ? "o"
```

Program : 47
Given a string, consider the prefix string made of the first N chars of the string. Does that prefix string appear somewhere else in the string? Assume that the string is not empty and that N is in the range 1..str.length().

```
prefixAgain("abXYabc", 1) ? true
prefixAgain("abXYabc", 2) ? true
prefixAgain("abXYabc", 3) ? false
```

Program : 48
Return a version of the given string, where for every star (*) in the string the star and the chars immediately to its left and right are gone. So "ab*cd" yields "ad" and "ab**cd" also yields "ad".

```
starOut("ab*cd") ? "ad"
starOut("ab**cd") ? "ad"
starOut("sm*eilly") ? "silly
```

Program : 49
Return the number of times that the string "hi" appears anywhere in the given string.

```
countHi("abc hi ho") ? 1
countHi("ABChi hi") ? 2
countHi("hihi") ? 2
```

Program : 50
Given two strings, return true if either of the strings appears at the very end of the other string, ignoring upper/lower case differences (in other words, the computation should not be "case sensitive"). Note: str.toLowerCase() returns the lowercase version of a string.

```
endOther("Hiabc", "abc") ? true
endOther("AbC", "HiaBc") ? true
endOther("abc", "abXabc") ? true
```

Program : 51
We'll say that a String is xy-balanced if for all the 'x' chars in the string, there exists a 'y' char somewhere later in the string. So "xxy" is balanced, but "xyx" is not. One 'y' can balance multiple 'x's. Return true if the given string is xy-balanced.

```
xyBalance("aaxbby") ? true
xyBalance("aaxbb") ? false
xyBalance("yaaxbb") ? false
```

Program : 52
Given a string and an int n, return a string made of the first n characters of the string, followed by the first n-1 characters of the string, and so on. You may assume that n is between 0 and the length of the string, inclusive (i.e. n >= 0 and n <= str.length()).

```
repeatFront("Chocolate", 4) ? "ChocChoChC"
repeatFront("Chocolate", 3) ? "ChoChC"
repeatFront("Ice Cream", 2) ? "IcI"
```

Program : 53
Given a string, does "xyz" appear in the middle of the string? To define middle, we'll say that the number of chars to the left and right of the "xyz" must differ by at most one.

```
xyzMiddle("AAxyzBB") ? true
xyzMiddle("AxyzBB") ? true
xyzMiddle("AxyzBBB") ? false
```

Program : 54
Given a string, compute a new string by moving the first char to come after the next two chars, so "abc" yields "bca". Repeat this process for each subsequent group of 3 chars, so "abcdef" yields "bcaefd". Ignore any group of fewer than 3 chars at the end.

```
oneTwo("abc") ? "bca"
oneTwo("tca") ? "cat"
oneTwo("tcagdo") ? "catdog"
```

Program : 55
Given a string and a non-empty word string, return a version of the original String where all chars have been replaced by pluses ("+"), except for appearances of the word string which are preserved unchanged.

```
plusOut("12xy34", "xy") ? "++xy++"
plusOut("12xy34", "1") ? "1+++++"
plusOut("12xy34xyabcxy", "xy") ? "++xy++xy+++xy"
```

Program : 56
Return true if the string "cat" and "dog" appear the same number of times in the given string.

```
catDog("catdog") ? true
catDog("catcat") ? false
catDog("1cat1cadodog") ? true
```

Program : 57
Return true if the given string contains an appearance of "xyz" where the xyz is not directly preceeded by a period (.). So "xxyz" counts but "x.xyz" does not.

```
xyzThere("abcxyz") ? true
xyzThere("abc.xyz") ? false
xyzThere("xyz.abc") ? true
```

Program : 58
Given two strings, a and b, create a bigger string made of the first char of a, the first char of b, the second char of a, the second char of b, and so on. Any leftover chars go at the end of the result.

```
mixString("abc", "xyz") ? "axbycz"
mixString("Hi", "There") ? "HTihere"
mixString("xxxx", "There") ? "xTxhxexre"
```

Program : 59
Given two strings, word and a separator sep, return a big string made of count occurrences of the word, separated by the separator string.

```
repeatSeparator("Word", "X", 3) ? "WordXWordXWord"
repeatSeparator("This", "And", 2) ? "ThisAndThis"
repeatSeparator("This", "And", 1) ? "This"
```

Program : 60
A sandwich is two pieces of bread with something in between. Return the string that is between the first and last appearance of "bread" in the given string, or return the empty string "" if there are not two pieces of bread.

```
getSandwich("breadjambread") ? "jam"
getSandwich("xxbreadjambreadyy") ? "jam"
getSandwich("xxbreadyy")
```

Program : 61
Look for patterns like "zip" and "zap" in the string -- length-3, starting with 'z' and ending with 'p'. Return a string where for all such words, the middle letter is gone, so "zipXzap" yields "zpXzp".

```
zipZap("zipXzap") ? "zpXzp"
zipZap("zopzop") ? "zpzp"
zipZap("zzzopzop") ? "zzzpzp"
```

Program : 62
Given a string and a non-empty word string, return a string made of each char just before and just after every appearance of the word in the string. Ignore cases where there is no char before or after the word, and a char may be included twice if it is between two words.

```
wordEnds("abcXY123XYijk", "XY") ? "c13i"
wordEnds("XY123XY", "XY") ? "13"
wordEnds("XY1XY", "XY") ? "11"
```

Program : 63
Given a string, count the number of words ending in 'y' or 'z' -- so the 'y' in "heavy" and the 'z' in "fez" count, but not the 'y' in "yellow" (not case sensitive). We'll say that a y or z is at the end of a word if there is not an alphabetic letter immediately following it. (Note: Character.isLetter(char) tests if a char is an alphabetic letter.)

```
countYZ("fez day") ? 2
countYZ("day fez") ? 2
countYZ("day fyyyz") ? 2
```

Program : 64
Given a string, return the sum of the numbers appearing in the string, ignoring all other characters. A number is a series of 1 or more digit chars in a row. (Note: Character.isDigit(char) tests if a char is one of the chars '0', '1', .. '9'. Integer.parseInt(string) converts a string to an int.)

```
sumNumbers("abc123xyz") ? 123
sumNumbers("aa11b33") ? 44
sumNumbers("7 11") ? 18
```

Program : 65
Given a string, look for a mirror image (backwards) string at both the beginning and end of the given string. In other words, zero or more characters at the very begining of the given string, and at the very end of the string in reverse order (possibly overlapping). For example, the string "abXYZba" has the mirror end "ab".

```
mirrorEnds("abXYZba") ? "ab"
mirrorEnds("abca") ? "a"
mirrorEnds("aba") ? "aba"
```

Program : 66
Implement atoi() like function to convert a string to an integer.
Consider all possible cases
e.g. positive and negative String, the presence of + or - character, etc.
For example, if the given input String is "123" then your program should return 123 and if a given input is "+231" then your program should return 231.

Program : 67
Write a program to check given String is palindrom or not
input String : "madam" or "Madam"
output : true

Program : 68
Write a program to findout longest palindrom word from input String
Example :
input String : "we are calling to radha madam as 121 mam only"
output : 3 palindrom words are there in given input string {madam , 121, mam} and the longest palindrom word is {madam}

Program : 69
Write a program to sum of all digits of String.
Example :
input String:"Naresh123It456Hyderabad27"
output: 30

Program : 70
Write a program to sum of all digits of String in folowing formate.
Example :
input String:"Naresh123It456Hyderabad27"
output: 6:16:9

Program : 71
Write a program to Remove uppercase, lowercase, special, numeric, and non-numeric characters from a String.
Example :
Input : "J@va12"
Output:

```
After removing uppercase characters: @va12
After removing lowercase characters: J@12
After removing special characters: Jva12
After removing numeric characters: J@va
After removing non-numeric characters: 12
```

Program : 72
Write a program to skip all number/digits from string and print only words as below formate.
Input String : "Welcome to 007 naresh it";
Output : "Welcome","to","naresh","it"

Program : 73
Write a program to reverse perticular word by position from String.
Example1 :
Input String : "Welcome to naresh it"
Input position : 3
Output : Welcome to hseran it

Program : 74
Write a program that swaps the two Strings without using the third string.
Example :

String1 ="Naresh"
String2 ="It"
Output :
String1 ="It"
String2 ="Naresh"

Program : 75
Write a program to find all missing characters from String.
Example :
Input String : "naresh"
Output : "bcdfgijklmopq"

NOTE : all characters must be in small letters.

Program : 76
Write a program to print all leader characters from a String.(Ignore Space)
Example :
Input String : "naxresh"
Output : x,s,h

Program : 77
Write a program to remove consecative characters from String.
--> Given String str. for each index i(1<=i<=N-1), erase it if str[i] is equal to s[i-1] in the string.
Example1 :
Input : String str = "aabb"
Output : ab
Explanation : 'a' at 2nd position is
appearing 2nd time consecutively.
Similiar explanation for b at
4th position.

Example2 :

Input :
 String str = aabaa

Output : aba

Explanation : 'a' at 2nd position is
appearing 2nd time consecutively.
'a' at fifth position is appearing
2nd time consecutively.

Program : 78
Write a program to convert a sentence into its equivalent mobile numeric keypad sequence.
Example1 :
Input : "NARESH IT"
Output : 6273740480
Example2 :
Input : "NARESH IT HYD"
Output : 6273740480493

NOTE :
1) check your mobile dialpad to genarate mobile number by string.
2) add 0 if the character is space
3) if string characters less than 10 or string length<=10 then add extra 0's at end and generate 10 digit mobile number like above given Example1.

Program : 79
Write a program to Swap Corner Words and Reverse Middle Characters of a String.
Example :
```
Input   : "Welcome to naresh it"
Output  : it hseran ot Welcome
```

Program : 80
Write a program to Add Two Binary Strings.
Example1 :
```
Input   :  x = "10", y = "01"
Output  : "11"
```
Example2 :
```
Input   : x = "110", y = "011"
Output  : "1001"
Explanation: 110 + 011=1001
```

Program : 81
Write a program to remove duplicate characters from String during creation of pairs of 4 character and print the output like below format.
Example1:
```
Input String :"Welcome to our institute naresh it"
Output : welc,omet,ouri,nsti,tuen,ares,hit
```

Program : 82
Write a program to check max occured caracter from String array and replace that perticular string by its max occured character and print the string array like below formate.
Example :
```
input String array : String arr[]={"welcome","to","nareshn",it};
Output : {"eeeeeee","to","nnnnnnn","it"}
```

NOTE: if count of all characters of pericular String is equal then do not replace.(keep same string).