**1)Expansion Of String :**

import java.util.Scanner;

public class ExpansionOfString {

public String printfreq(String s) {

StringBuilder str = new StringBuilder();

int count\_occ = 1;

char let;

for (int i = 0; i < s.length(); i += 2) {

let = s.charAt(i);

count\_occ = Character.getNumericValue(s.charAt(i + 1));

for (int j = 0; j < count\_occ; j++) {

str.append(let);

}

}

String result = str.toString();

return result;

}

public static void main(String[] args) {

ExpansionOfString Exp = new ExpansionOfString();

Scanner scan = new Scanner(System.in);

System.out.print("Enter the string: ");

String str = scan.nextLine();

String output = Exp.printfreq(str);

System.out.println("Output: " + output);

scan.close();

}

}

**2)Compression Of String:**

import java.util.Scanner;

public class StringCompression {

public static String compress(String input) {

StringBuilder compressedString = new StringBuilder();

int count = 1;

for (int i = 0; i < input.length(); i++) {

char currentChar = input.charAt(i);

if (i + 1 < input.length() && input.charAt(i + 1) == currentChar) {

count++;

} else {

compressedString.append(currentChar);

compressedString.append(count);

count = 1;

}

}

return compressedString.toString();

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the string: ");

String input = scanner.nextLine();

scanner.close();

String compressed = compress(input);

System.out.println("Compressed string: " + compressed);

}

}

**3)Numbers to their respective names:**

import java.util.Scanner;

public class NumberToName {

private static final String[] ones = {"", "One", "Two", "Three", "Four", "Five", "Six", "Seven", "Eight", "Nine"};

private static final String[] tens = {"", "", "Twenty", "Thirty", "Forty", "Fifty", "Sixty", "Seventy", "Eighty", "Ninety"};

private static final String[] teens = {"Ten", "Eleven", "Twelve", "Thirteen", "Fourteen", "Fifteen", "Sixteen", "Seventeen", "Eighteen", "Nineteen"};

public static String numberToName(int number) {

if (number == 0) {

return "Zero";

}

return convertToWords(number);

}

private static String convertToWords(int number) {

if (number < 10) {

return ones[number];

} else if (number < 20) {

return teens[number - 10];

} else if (number < 100) {

return tens[number / 10] + (number % 10 != 0 ? " " + ones[number % 10] : "");

} else if (number < 1000) {

return ones[number / 100] + " Hundred" + (number % 100 != 0 ? " " + convertToWords(number % 100) : "");

} else if (number < 10000) {

return ones[number / 1000] + " Thousand" + (number % 1000 != 0 ? " " + convertToWords(number % 1000) : "");

} else {

return "Number out of range";

}

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a number: ");

int number = scanner.nextInt();

scanner.close();

String name = numberToName(number);

System.out.println("Output: " + name);

}

}

**4)Mismatching Pair of Strings:**

import java.util.Scanner;

public class MismatchString {

public static void compareStrings(String str1, String str2) {

int minLength = Math.min(str1.length(), str2.length());

System.out.println("Mismatching characters:");

for (int i = 0; i < minLength; i++) {

if (str1.charAt(i) != str2.charAt(i)) {

System.out.println(str1.charAt(i) + "," + str2.charAt(i));

}

}

if (str1.length() != str2.length()) {

int maxLength = Math.max(str1.length(), str2.length());

String longerString = str1.length() > str2.length() ? str1 : str2;

for (int i = minLength; i < maxLength; i++) {

System.out.println("N/A," + longerString.charAt(i));

}

}

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the first string: ");

String str1 = scanner.nextLine();

System.out.print("Enter the second string: ");

String str2 = scanner.nextLine();

scanner.close();

compareStrings(str1, str2);

}

}

**5)Justify the string**

public class TextJustification {

public static void main(String[] args) {

String text = "Zoho\_Corp\_Madurai";

int desiredLength = 25;

String justifiedText = justifyText(text, desiredLength);

System.out.println("Input: " + text);

System.out.println("Output: " + justifiedText);

}

private static String justifyText(String text, int desiredLength) {

String[] words = text.split("\_");

int numberOfSpaces = words.length - 1;

int totalSpacesToAdd = desiredLength - text.length();

if (numberOfSpaces == 0) {

return text;

}

int spacesToAddPerWord = totalSpacesToAdd / numberOfSpaces;

int extraSpaces = totalSpacesToAdd % numberOfSpaces;

StringBuilder justifiedText = new StringBuilder(words[0]);

for (int i = 1; i < words.length; i++) {

for (int j = 0; j < spacesToAddPerWord; j++) {

justifiedText.append(' ');

}

if (extraSpaces > 0) {

justifiedText.append(' ');

extraSpaces--;

}

justifiedText.append(words[i]);

}

return justifiedText.toString();

}

}

**6)Palindrome**

import java.util.Scanner;

public class Palindrome {

public static boolean isPalindrome(String str) {

int left = 0;

int right = str.length() - 1;

while (left < right) {

char leftChar = str.charAt(left);

char rightChar = str.charAt(right);

if (!Character.isLetterOrDigit(leftChar)) {

left++;

continue;

}

if (!Character.isLetterOrDigit(rightChar)) {

right--;

continue;

}

if (Character.toLowerCase(leftChar) != Character.toLowerCase(rightChar)) {

return false;

}

left++;

right--;

}

return true;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a string: ");

String input = scanner.nextLine();

scanner.close();

if (isPalindrome(input)) {

System.out.println("True.");

} else {

System.out.println("False.");

}

}

}

**7)Permutation**

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class Permutation {

public static List<String> permutationsWithoutRepetition(String s) {

List<String> permutations = new ArrayList<>();

backtrack(s.toCharArray(), new boolean[s.length()], new StringBuilder(), permutations);

return permutations;

}

private static void backtrack(char[] chars, boolean[] used, StringBuilder current, List<String> permutations) {

if (current.length() == chars.length) {

permutations.add(current.toString());

return;

}

for (int i = 0; i < chars.length; i++) {

if (!used[i]) {

used[i] = true;

current.append(chars[i]);

backtrack(chars, used, current, permutations);

current.deleteCharAt(current.length() - 1);

used[i] = false;

}

}

}

public static void main(String[] args) {

Scanner scan = new Scanner(System.in);

System.out.print("Enter the string: ");

String inputString = scan.nextLine();

List<String> result = permutationsWithoutRepetition(inputString);

for (String permutation : result) {

System.out.println(permutation);

}

}

}

**8)Mismatched SubStringss**

public class MismatchedSubstring {

public static void main(String[] args) {

String str1 = "AABBCCDD";

String str2 = "ABCDCCAD";

System.out.println("Input: " + str1 + ", " + str2);

findMismatchedSubstrings(str1, str2);

}

private static void findMismatchedSubstrings(String str1, String str2) {

int minLength = Math.min(str1.length(), str2.length());

for (int i = 0; i < minLength; i++) {

if (str1.charAt(i) != str2.charAt(i)) {

int j = i + 1;

while (j < minLength && str1.charAt(j) != str2.charAt(j)) {

j++;

}

System.out.println(str1.substring(i, j) + "," + str2.substring(i, j));

i = j - 1;

}

}

}

}

**9)Vowels count in the string**

import java.util.HashMap;

import java.util.Map;

import java.util.Scanner;

public class VowelCount {

public static Map<Character, Integer> countVowels(String str) {

Map<Character, Integer> vowelCountMap = new HashMap<>();

str = str.toLowerCase();

for (char c : str.toCharArray()) {

if (isVowel(c)) {

vowelCountMap.put(c, vowelCountMap.getOrDefault(c, 0) + 1);

}

}

return vowelCountMap;

}

private static boolean isVowel(char c) {

c = Character.toLowerCase(c);

return c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u';

}

public static void main(String[] args) {

Scanner scan = new Scanner(System.in);

System.out.print("Enter the string: ");

String inputString = scan.nextLine();

Map<Character, Integer> vowelsCountMap = countVowels(inputString);

char[] vowels = {'a', 'e', 'i', 'o', 'u'};

for (char vowel : vowels) {

System.out.println(vowel + "-" + vowelsCountMap.getOrDefault(vowel, 0));

}

}

}

**10)Next palindrome number for given number**

import java.util.Scanner;

public class NextPalindrome {

public static int nextPalindrome(int num) {

while (true) {

num++;

if (isPalindrome(num)) {

return num;

}

}

}

private static boolean isPalindrome(int num) {

String strNum = String.valueOf(num);

int left = 0;

int right = strNum.length() - 1;

while (left < right) {

if (strNum.charAt(left) != strNum.charAt(right)) {

return false;

}

left++;

right--;

}

return true;

}

public static void main(String[] args) {

Scanner scan = new Scanner(System.in);

System.out.print("Enter the number: ");

int givenNumber = scan.nextInt();

int nextPalin = nextPalindrome(givenNumber);

System.out.println("Output:" + nextPalin);

}

}