**Level 1:**

**Easy**

1.Write a program to encrypt the text “INDIA” to “KPFKC” and decrypt “KPFKC” to get the original string “INDIA”.

**package** com.stringpractice;

**public** **class** Easy1 {

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

String s="INDIA";

String encrypted="";

**for**(**char** c: s.toCharArray()) {

encrypted += (**char**)(c+2);

}

System.***out***.println("Encrypted text: "+encrypted);

String decrypted="";

**for**(**char** c: encrypted.toCharArray()){

decrypted += (**char**)(c-2);

}

System.***out***.println("Decrypted text: "+decrypted);

}

}

Output:

Encrypted text: KPFKC

Decrypted text: INDIA

1. A string S is passed as the input. S can contain alphabets, numbers and special characters. The program must print only the alphabets in S. Input: abcd\_5ef8!xyz Output: abcdefxyz

**package** com.stringpractice;

**public** **class** Easy2 {

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

String s = "abcd\_5ef8!xyz";

String alphabet = "";

**for** (**char** c : s.toCharArray()) {

**if** ((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z')) {

alphabet += c;

}

}

System.***out***.println("String with only alphabets: " + alphabet);

}

}

Output:

String with only alphabets: abcdefxyz

1. Write a program to sort set of names stored in an array in alphabetical order.

**package** com.stringpractice;

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

**public** **class** Easy3 {

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

String arr[] = {"jeevi", "dharani", "gayu"};

Arrays.*sort*(arr);

System.***out***.println("After sorting the names:");

**for** (String name : arr) {

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

}

}

}

Output:

After sorting the names:

dharani

gayu

jeevi

1. Write a program to input a string in uppercase and print the occurrence of each character.

**package** com.stringpractice;

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

**public** **class** Easy4 {

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

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

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

String s = sc.nextLine();

**int**[] frequency = **new** **int**[26];

**for** (**char** c : s.toCharArray()) {

**if** (c >= 'A' && c <= 'Z') {

frequency[c - 'A']++;

}

}

System.***out***.println("Character occurrences:");

**for** (**int** i = 0; i < 26; i++) {

**if** (frequency[i] > 0) {

System.***out***.println((**char**) (i + 'A') + " : " + frequency[i]);

}

}

}

}

Output:

Enter the string in uppercase:WELCOME HOME

Character occurrences:

C : 1

E : 3

H : 1

L : 1

M : 2

O : 2

W : 1

**Level 2:**

**Medium**

1. In a real-time conferencing, there is a delay in getting the 3rd word of the text message received. To overcome this issue, the word “... Pls wait “is inserted as the third word in the given message. Write a code to fix it. Sample 1 Input: All is well. Output: All is ... Pls wait well. Sample 2: Input: Nature is pure. Respect it. Enjoy it. Output: Nature is ... Pls wait pure. Respect it. Enjoy it.

**package** com.stringpractice;

**public** **class** Med1 {

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

String s="All is Well";

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

**for**(**int** i=0;i<2;i++) {

System.***out***.print(arr[i]+" ");

}

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

System.***out***.println("let..wait");

**for**(**int** i=2;i<arr.length;i++) {

System.***out***.print(arr[i]);

}

}

}

Output:

All is

let..wait

Well

2. Write a C# program to generate password from first name and last name. password includes 1st three character in first name, last three character in last name and length of first and last name. Sample input: Sachin Tendulkar Sample output: sackar15

**package** com.stringpractice;

**public** **class** Med2 {

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

String s1 = "Sachin";

String s2 = "Tendulkar";

String firstName = "";

**for** (**int** i = 0; i < 3; i++) {

firstName += s1.charAt(i);

}

String lastName = "";

**for** (**int** i = s2.length() - 3; i < s2.length(); i++) {

lastName += s2.charAt(i);

}

**int** totalLength = s1.length() + s2.length();

System.***out***.println("Password: " + firstName + lastName + totalLength);

}

}

Output:

Password: Sackar15

3. Write the program to arrange the following names in alphabetic order. The sorting is to be done on the first three characters of the first name.(Ashok, Alok, Akash, Amit, Amol, Anil, Ashish and Anand)

**package** com.stringpractice;

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

**public** **class** Med3 {

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

String[] names = {"Ashok", "Alok", "Akash", "Amit", "Amol", "Anil", "Ashish", "Anand"};

Arrays.*sort*(names, (a, b) -> a.substring(0, 3).compareTo(b.substring(0, 3)));

System.***out***.println("Sorted Names:");

**for** (String x : names) {

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

}

}

}

Output:

Sorted Names:

Akash

Alok

Amit

Amol

Anand

Anil

Ashok

Ashish

4.Write a program to count the number of occurrences of any two vowels in succession in a line of text. For example, in the sentence “Please read this application and give me gratuity” such occurrences are ea,ea,ui

**package** com.stringpractice;

**import** java.util.Scanner;

**public** **class** Med4 {

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

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

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

String s=sc.nextLine();

**int** c=0;

**for**(**int** i=0;i<s.length()-1;i++) {

**if**((s.charAt(i)=='a'||s.charAt(i)=='e'||s.charAt(i)=='i'||s.charAt(i)=='o'||s.charAt(i)=='u') && (s.charAt(i+1)=='a'|| s.charAt(i+1)=='e'|| s.charAt(i+1)=='i'||s.charAt(i+1)=='o'||s.charAt(i+1)=='u')){

c++;

}

}

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

}

}

Output:

enter the string

please read this application and give me gratuity

the count of occurence of successive vowels :4

**Level 3:**

**Hard:**

1. A Caesar cipher or shift cipher encrypts the given message by substituting the shifted letters of alphabets as key per. With a shift key of 1, would encode an A as a B, an M as an N, and a Z as an A, and so on. With a shift key of 3, would encode an A as a D, an M as an P, and a Z as an C, and so on. The sending end does this and the receiving end does the reverse process. Write C# code for either sending or receiving end. Sample 1 1.Sender 2.Receiver. Enter your choice 1 Enter the key: 2 Enter the original text : HELLO Op: The encrypted text is... JGNNQ Sample 2 1.Sender 2.Receiver. Enter your choice 2 Enter the key: 3 Enter the encrypted text: grj Op: The original text is... dog

**package** com.stringpractice;

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

**public** **class** Hard1 {

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

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

String s = "HELLO";

**int** key = 2;

System.***out***.println("enter the choice: 1-sender 2-receiver");

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

String res = "";

**if** (choice == 1) {

**for** (**char** c : s.toCharArray()) {

**if** (c >= 'A' && c <= 'Z') {

**char** shiftedChar = (**char**) ((c - 'A' + key) % 26 + 'A');

res += shiftedChar;

} **else** {

res += c;

}

}

System.***out***.println("Encrypted text: " + res);

}

**else** **if** (choice == 2) {

**for** (**char** c : s.toCharArray()) {

**if** (c >= 'A' && c <= 'Z') {

**char** shiftedChar = (**char**) ((c - 'A' - key + 26) % 26 + 'A');

res += shiftedChar;

} **else** {

res += c;

}

}

System.***out***.println("Decrypted text: " + res);

}

}

}

Output:

enter the choice: 1-sender 2-receiver

1

Encrypted text: JGNNQ

1. Anne joined a social networking site to stay in touch with her friends. The signup page required her to input a name and a password. However, the password must be strong. The website considers a password to be strong if it satisfies the following criteria: a. Its length is at least 6. b. It contains at least one digit. c. It contains at least one lowercase English character. d. It contains at least one uppercase English character. e. It contains at least one special character. The special characters are: !@#$%^&\*()-+ She typed a random string of length n in the password field but wasn’t sure if it was strong. Given the string she typed, can you find the minimum number of characters she must add to make her password strong? Note: Here’s the set of types of characters in a form you can paste in your solution: numbers = “0123456789” lower\_case = “abcdefghijklmnopqrstuvwxyz” upper\_case = “ABCDEFGHIJKLMNOPQRSTUVWXYZ” special\_characters = “!@#$%^&\*()-+” Given a string, find its strength. Let a strong password is one that satisfies all above conditions. A moderate password is one that satisfies first three conditions and has length at least 6. Otherwise password is week.

Sample Input : "Password!@12" Output : Strong Input : "pa!@12" Output : Moderate

**package** com.stringpractice;

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

**public** **class** Hard2 {

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

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

System.***out***.print("Enter Password: ");

String pass = sc.nextLine();

sc.close();

**int** d = 0, l = 0, u = 0, lo = 0, s = 0;

String specialChars = "!@#$%^&\*()\_+-";

**if** (pass.length() >= 6) {

l = 1;

}

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

**char** a = pass.charAt(i);

**if** (Character.*isDigit*(a)) {

d = 1;

}

**if** (Character.*isLowerCase*(a)) {

lo = 1;

}

**if** (Character.*isUpperCase*(a)) {

u = 1;

}

**if** (specialChars.contains(String.*valueOf*(a))) {

s = 1;

}

}

**if** (d == 1 && l == 1 && lo == 1 && u == 1 && s == 1) {

System.***out***.println("Strong");

} **else** **if** (l == 1 && d == 1 && lo == 1) {

System.***out***.println("Moderate");

} **else** {

System.***out***.println("Weak");

}

}

}

Output:

Enter Password: Jeevika&222222

Strong