1. Given a string as input, write a program to convert the characters of given string into hexadecimal equivalent of ASCII values.

Examples :

Input : Geek

Output : 4765656b

Input : IronMan part 3

Output : 49726f6e4d616e20706172742033

Answer:

import java.util.\*;

public class Q1 {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

String str=input.nextLine();

String hex="";

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

char ch=str.charAt(i);

int in=(int)ch;

hex+=Integer.toHexString(in);

}

System.out.println(hex);

}

}

2. Swap corner words and reverse middle characters

Write a Java program to take an input string and exchange the first and last word and revers the middle word.

Examples:

Input : Hello World GFG Welcomes You

Output :You semocleW GFG dlroW Hello

Answer:

(USING SPLIT METHOD AND ArrayList)

import java.util.Scanner;

public class Q2 {

public static void main(String args[]) {

Scanner input = new Scanner(System.in);

String str = input.nextLine();

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

String temp = arr[0];

arr[0] = arr[arr.length - 1];

arr[arr.length - 1] = temp;

String mid = "";

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

mid += arr[i] + " ";

}

String rev = "";

for (int i = mid.length() - 1; i >= 0; i--) {

rev += mid.charAt(i);

}

System.out.println(arr[0] +rev + " " + arr[arr.length - 1]);

}

}

OR

(WITHOUT USING SPLIT)

import java.util.Scanner;

public class Q2a {

public static void main(String args[]){

Scanner input = new Scanner(System.in);

String str = input.nextLine();

String first = "";

int i=0;

for(; i<str.length();){

while(str.charAt(i)!=' '){

first+=str.charAt(i);

i++;

}

break;

}

String last="";

int j=str.length()-1;

for(; j>=0;){

while(str.charAt(j)!=' '){

last=str.charAt(j)+last;

j--;

}

break;

}

System.out.print(last);

for(int k=j;k>=i;k--){

System.out.print(str.charAt(k));

}

System.out.print(first);

}

}

3. Longest Common Substring in an Array of Strings

We are given a list of words sharing a common stem i.e the words originate from same word for ex: the words sadness, sadly and sad all originate from the stem ‘sad’.

Our task is to find and return the Longest Common Substring also known as stem of those words. In case there are ties, we choose the smallest one in alphabetical order.

Examples:

Input : grace graceful disgraceful gracefully

Output : grace

Input : sadness sad sadly

Output : sad

Answer:

import java.util.\*;

public class Q3 {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

int n = input.nextInt();

String[] words = new String[n];

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

words[i] = input.next();

}

System.out.println(longestCommonSubstring(words));

}

public static String longestCommonSubstring(String[] words) {

int n = words.length;

String s = words[0];

int len = s.length();

String res = "";

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

for (int j = i + 1; j <= len; j++) {

String stem = s.substring(i, j);

int k = 1;

for (k = 1; k < n; k++){

if (!words[k].contains(stem))

break;

}

if (k == n && res.length() <= stem.length())

if(res.length()==stem.length()){

if(res.compareTo(stem)>0)

res = stem;

}

else

res = stem;

}

}

if(res.length() == 0)

res+="No common substring";

return res;

}

}

//complexity: O(n^2)

4. An anagram is a word or a phrase made by transposing the letters of another word or phrase; for example, "parliament" is an anagram of "partial men," and "software" is an anagram of "swear oft." Write a program that figures out whether one string is an anagram of another string. The program should ignore white space and punctuation.

Answer:

import java.util.Scanner;

import java.util.Arrays;

public class Q4 {

public static void main(String args[]){

Scanner input=new Scanner(System.in);

String str1=input.nextLine();

String str2=input.nextLine();

str1=str1.replaceAll("\\p{Punct}|\\p{Space}", "");

str2=str2.replaceAll("\\p{Punct}|\\p{Space}", "");

char[] arr1=str1.toCharArray();

Arrays.sort(arr1);

char[] arr2=str2.toCharArray();

Arrays.sort(arr2);

int i=0;

for(;i<arr1.length;i++){

if(arr1[i]==arr2[i]){

continue;

}

else{

System.out.println("Given Strings are not Anagram of each other");

break;

}

}

if(i==arr1.length){

System.out.println("Given Strings are Anagram of each other");

}

}

}

5. Write a Java program to print the following grid. Go to the editor

Expected Output :

- -

- - - -

- - - - - -

- - - - - - - -

- - - - - - - - - -

- - - - - - - -

- - - - - -

- - - -

- -

Answer:

import java.util.Scanner;

public class Q5 {

public static void main(String args[]){

Scanner input = new Scanner(System.in);

int n=input.nextInt();

if(n%2!=0){

for(int i=1;i<=n;i++){

if(i<=n/2+1){

for(int j=n/2;j>=i;j--){

System.out.print(" ");

}

for(int k=1;k<=2\*i;k++){

System.out.print("-");

}

System.out.println();

}

else{

for(int j=1;j<=i-n/2-1;j++){

System.out.print(" ");

}

for(int k=1;k<=2\*(n-i+1);k++){

System.out.print("-");

}

System.out.println();

}

}

}

else{

for(int i=1;i<=n;i++){

if(i<=n/2){

for(int j=n/2-1;j>=i;j--){

System.out.print(" ");

}

for(int k=1;k<=2\*i;k++){

System.out.print("-");

}

System.out.println();

}

else{

for(int j=1;j<=i-n/2-1;j++){

System.out.print(" ");

}

for(int k=1;k<=2\*(n-i+1);k++){

System.out.print("-");

}

System.out.println();

}

}

}

}

}

6. Ludic numbers are obtained by considering list of natural numbers (starting from 2) and removing i-th number in i-th iteration (where i begins with 2). In every iteration, the first removed number is Ludic. 1 is considered as Ludic.

Process of generating Ludic numbers :

Ludic = {1, …}

Consider natural numbers from 2,

2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21 …

Delete every 2nd number

3, 5, 7, 9, 11, 13, 15, 17, 19, 21 ..

The first deleted number is 2.

Ludic = {1, 2, …}

Delete every 3rd number.

5, 7, 11, 13, 17, 19, 22 ..

The first deleted number is 3

Ludic = {1, 2, 3, …}

Delete every 4th number.

5, 7, 11, 13, 17, 19, 22 ..

The first deleted number is 5

Ludic = {1, 2, 3, 5, ..}

This process continues..

Given a number n, print all Ludic numbers smaller than or equal to n.

Examples :

Input : n = 10

Output : 1, 2, 3, 5, 7

Input : n = 25

Output : 1, 2, 3, 5, 7, 11, 13, 17, 23, 25

Answer:

import java.util.ArrayList;

import java.util.Scanner;

public class Q6 {

public static void main(String args[]){

Scanner input=new Scanner(System.in);

int n=input.nextInt();

ArrayList<Integer> ludic=new ArrayList<Integer>();

for(int i=1;i<=n;i++){

ludic.add(i);

}

for(int index=1;index<ludic.size();index++){

int first=ludic.get(index);

int rem=first+index;

while(rem<ludic.size()){

ludic.remove(rem);

rem+=first-1;

}

}

System.out.println(ludic);

}

}