

# Software Engineering

*-supervised by:*

Dr. Suparna Biswas (Saha)

ROHIT DAS

B. Tech(Computer Sc. and Engg)

Roll: 30000114022

Regn. No.:143000110023

6th Semester,2016



*Maulana Abul Kalam Azad University of Technology,  
West Bengal.*

L<sup>A</sup>T<sub>E</sub>X 2017

---

## Contents

<b>1. Assignment-1: C programming (Date: 27/10/2017)</b>	<b>3</b>
1..1 Write a program to implement Calendar program to display Day of the month. Program will accept year, month and date from user and will display the day of the month. . . . .	3
1..2 Write a program to find inverse of 3x3 matrix. . . . .	4
1..3 Write a program to check whether matrix is Magic Square or not. . . . .	5
1..4 Write a program to read last n characters from a file(input should be a.txt). . . . .	7
1..5 Write a program to print binary numbers in pyramid pattern. . . . .	8
1..6 Write a program to input password for validation of username. Enter password: ***** Password entered: sourav . . . . .	9
1..7 Write a program to create your own header file in C. . . . .	10
1..8 Write a program to compare two strings without using library function(strcmp). . . . .	11
1..9 Write a program to print a rectangle using line and special symbols. . . . .	12
1..10 Write a program to find addition of lower triangular matrix. . . . .	14
<b>2. Assignment-2: Java programming (Date: 30/10/2017)</b>	<b>16</b>
2..1 An array contains 20 integers arranged randomly. Write a Java program using quick sort to sort these numbers in ascending order. . . . .	16
2..2 Write a Java program to take an integer not greater than 20 as input and calculate the factorial of the integer recursively. . . . .	17
2..3 You are starting out on a long (really long) trip. On the way, there are N gas stations, the locations of which are given as $a_1, a_2, \dots, a_N$ . Initially you are located at the gas station at $a_1$ , and your destination is at location $a_N$ . . . . .	18
2..4 Given N separate integer points on the Cartesian plane satisfying: there is no any three of them sharing a same X-coordinate. Your task is to count the number of rectangles (whose edges parrallel to the axes) created from any four of given points. . . . .	19
2..5 Given two strings, write a program that outputs the shortest sequence of character insertions and deletions that turn one string into another. . . . .	23
2..6 Write a java program to generate a 3x3 magic square matrix, where a magic square is a matrix whose all numbers along every row, column and diagonal add up to the same number. . . . .	24

---

## 1. Assignment-1: C programming (Date: 27/10/2017)

1.1 Write a program to implement Calendar program to display Day of the month. Program will accept year, month and date from user and will display the day of the month.

Program:

```
/*
 * Copyright Rohit Das (C) 2017
 * To display day from date
 */
#include<stdio.h>
#include<math.h>

int fm(int date, int month, int year) {
    int _month, leap;
    //leap function 1 for leap & 0 for non-leap
    if ((year % 100 == 0) && (year % 400 != 0))    leap = 0;
    else if (year % 4 == 0)    leap = 1;
    else    leap = 0;
    _month = 3 + (2 - leap) * ((month + 2) / (2 * month))
        + (5 * month + month / 9) / 2;
    _month %= 7;
    return _month;
}

int day_of_week(int date, int month, int year) {
    int day;
    int yy = year % 100;
    int century = year / 100;
    printf("\nDate: %d/%d/%d \n", date, month, year);
    day = 1.25 * yy + fm(date, month, year) + date - 2 * (century % 4);
    //remainder on division by 7
    day %= 7;
    switch (day) {
        case 0:
            printf("weekday = Saturday");    break;
        case 1:
            printf("weekday = Sunday");    break;
        case 2:
            printf("weekday = Monday");    break;
        case 3:
            printf("weekday = Tuesday");    break;
        case 4:
            printf("weekday = Wednesday");    break;
        case 5:
            printf("weekday = Thursday");    break;
        case 6:
            printf("weekday = Friday");    break;
        default:
            printf("Incorrect data");
    }
}
```

```

    printf("\n");    return 0;
}

int main() {
    int date, month, year;
    printf("\nEnter the year: ");    scanf("%d", &year);
    printf("\nEnter the month: ");    scanf("%d", &month);
    printf("\nEnter the date: ");    scanf("%d", &date);
    day_of_week(date, month, year);
    return 0;
}

```

Output:

```

mouri@intell0-hub > ~/SoftwareEngg/assign1 > ./1
Enter the year 2017
Enter the month 11
Enter the date 6
Date: 6/11/2017
weekday = Monday
mouri@intell0-hub > ~/SoftwareEngg/assign1 > |

```

## 1.2 Write a program to find inverse of 3x3 matrix.

Program:

```

/*
 * Copyright Rohit Das (C) 2017
 * To display inverse of a 3x3 matrix
 */
#include<stdio.h>

int main(){
    int a[3][3], i, j;
    float determ=0;
    printf("Enter the 9 elements of matrix: \n");
    for(i=0;i<3;i++)        for(j=0;j<3;j++)        scanf("%d",&a[i][j]);
    printf("\nThe matrix is:\n");
    for(i=0;i<3;i++){
        printf("\n");
        for(j=0;j<3;j++)        printf("%d\t",a[i][j]);
    }
    for(i=0;i<3;i++)        determ += (a[0][i]*(a[1][((i+1)%3])*a[2][((i+2)%3] -
a[1][((i+2)%3]*a[2][((i+1)%3]));
    printf("\nInverse of matrix is: \n\n");
    for(i=0;i<3;i++){
        for(j=0;j<3;j++)
            printf("%.2f\t",((a[((i+1)%3][(j+1)%3] * a[((i+2)%3][(j+2)%3]) -
(a[((i+1)%3][(j+2)%3]*a[((i+2)%3][(j+1)%3]))/ determ);

```

```

        printf("\n");
    }
    return 0;
}

```

Output:

```

mouri@intell0-hub ~/SoftwareEngg/assign1 ./2
Enter the 9 elements of matrix: 1 5 3 4 5 9 7 8 2

The matrix is
1      5      3
4      5      9
7      8      2
Inverse of matrix is:
-0.30   0.27  -0.01
0.07   -0.09   0.13
0.15   0.01  -0.07
mouri@intell0-hub ~/SoftwareEngg/assign1 |

```

### 1.3 Write a program to check whether matrix is Magic Square or not.

Program:

```

/*
 * Copyright Rohit Das (C) 2017
 * To check if a matrix is Magic Square
 */
#include<stdio.h>
#include<stdlib.h>

void input();    void init();    void getdata();    void validate();
void validmat();    int leftdia();    int rightdia();    int rowsum();
int colsum();

//global vars
int arr[100][100],size;

//functions
void input(){
    int num=0;
    printf("Enter magic square size: \n");    scanf("%d",&size);
    init(size,num);
    printf("enter the elements: \n");        getdata(size);
}

void init(int size,int num)
{
    int i,j;
    for(i=0;i<size;i++)        for(j=0;j<size;j++)        arr[i][j]=num;
}

```

```

void getdata(int size)
{
    int i,j,num;
    for(i=0;i<size;i++){
        for(j=0;j<size;j++){
            scanf("%d",&num);          arr[i][j]=num;
        }
    }
}

void validate()
{
    int i,j,k;    int ar[size*2+2];
    int arsize=(size*2+2);
    for(i=0;i<(size*2+2);i++)    ar[i]=0;
    for(i=0;i<size;i++)    ar[i]=rowsum(arr,i);
    for(j=0;j<size;j++)    ar[j+i]=colsum(arr,j);
    ar[j+i]=leftdia();    ar[j+i+1]=rightdia();
    for (i = 0; i < size; i++) {
        for (j = 0; j < size; j++) printf("%d ",arr[i][j]);
        printf("\n");
    }
    validmat(ar,arsize);
}

void validmat(int ar[],int arsize)//validation
{
    int i,valid=0;
    for(i=0;i<arsize-1;i++)
    {
        if(ar[i]==ar[i+1])    valid=1;
        else    valid=0;
    }
    if(valid==1)    printf("This matrix is a magic square.\n");
    else    printf("This matrix is not a magic square.\n");
}

int leftdia()    //left diagonal sum
{
    int i,sum=0;
    for(i=0;i<size;i++)    sum=sum+arr[i][i];
    return sum;
}

int rightdia()    //right diagonal sum
{
    int i,sum=0;
    for(i=0;i<size;i++)    sum=sum+arr[i][size-1-i];
    return sum;
}

```

```

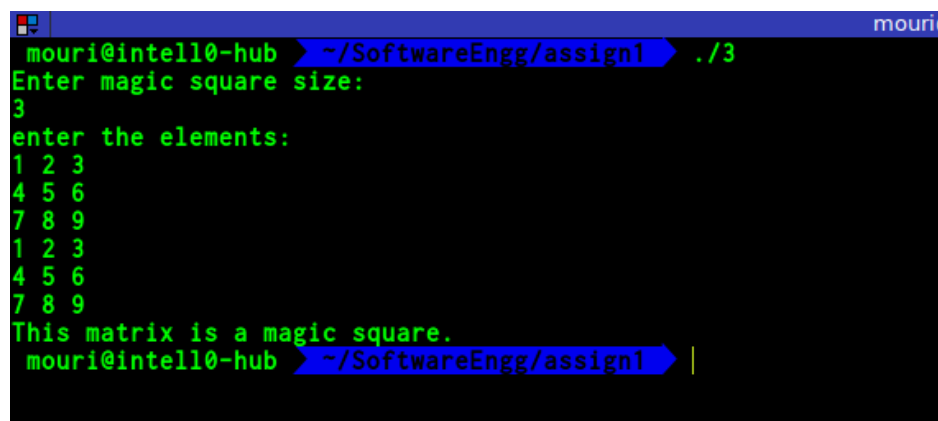
int rowsum(int arr[100][100],int index) //calculates row sum
{
    int i,x=0;
    for(i=0;i<size;i++)    x=x+arr[index][i];
    return x;
}

int colsum(int arr[100][100],int index)//calculates sum of the columns
{
    int i,x=0;
    for(i=0;i<size;i++)    x=x+arr[i][index];
    return x;
}

int main()
{
    input();    validate();
    return 0;
}

```

Output:



```

mouri@intell0-hub > ~/SoftwareEngg/assign1 > ./3
Enter magic square size:
3
enter the elements:
1 2 3
4 5 6
7 8 9
1 2 3
4 5 6
7 8 9
This matrix is a magic square.
mouri@intell0-hub > ~/SoftwareEngg/assign1 > |

```

1..4 Write a program to read last n characters from a file(input should be a.txt).

Program:

```

/*
 * Copyright Rohit Das (C) 2017
 * a.txt for 4.c
 */
12345678910

```

```

/*
 * Copyright Rohit Das (C) 2017
 * To read last n characters from a file (used a.txt)
 */
#include <stdio.h>
#include <stdlib.h>

int main() {

```

```

FILE *fp;    char ch;
int num;    long length;
printf("Enter the value of num : ");
scanf("%d", &num);
fp = fopen("a.txt", "r");           //open the file
if (fp == NULL) {
    puts("cannot open this file");    exit(1);
}
fseek(fp, 0, SEEK_END);             //set fp to end of file
length = ftell(fp);
fseek(fp, (length - num - 1), SEEK_SET); //2 because of EOF
do {
    ch = fgetc(fp);    putchar(ch);
} while (ch != EOF);
printf("\n");
fclose(fp);    return(0);
}

```

Output:

```

mouri@intell0-hub > ~/SoftwareEngg/assign1 > ./4
Enter the value of num : 5
6789100
mouri@intell0-hub > ~/SoftwareEngg/assign1 > |

```

### 1.5 Write a program to print binary numbers in pyramid pattern.

Program:

```

/*
 * Copyright Rohit Das (C) 2017
 * To display binary pattern as shown:
1
0 1
1 0
1 0 1
0 1 0 1
1 0 1 0 1
 */
#include<stdio.h>
#include<stdlib.h>

int main()
{
    int n,i,j,k;    k = 1;
    printf("Enter the number of lines\n");    scanf("%d",&n);
    for(i=0;i<n;i++) {
        for(j=0;j<=i;j++) {
            printf("%d",k);
            if(k==1)    k=0;

```



```

        else    k=1;
    }
    printf("\n");
}
return 0;
}

```

Output:

```

mouri@intell0-hub ~/SoftwareEngg/assign1 ./5
Enter the number of lines
7
1
01
010
1010
10101
010101
0101010
mouri@intell0-hub ~/SoftwareEngg/assign1 |

```

### 1..6 Write a program to input password for validation of username.

Enter password: \*\*\*\*\*

Password entered: sourav

Program:

```

/*
 * Copyright Rohit Das (C) 2017
 * To validate entered password
 */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <termios.h>
#include <ctype.h>

/* Use this variable to remember original terminal attributes. */
struct termios saved_attributes;

void reset_input_mode (void)
{
    tcsetattr (STDIN_FILENO, TCSANOW, &saved_attributes);
}

void set_input_mode (void)
{
    struct termios tattr;    char *name;
    /* Make sure stdin is a terminal. */
    if (!isatty (STDIN_FILENO)){
        fprintf (stderr, "Not a terminal.\n");
    }
}

```

```

    exit (EXIT_FAILURE);
}
/* Save the terminal attributes so we can restore them later. */
tcgetattr (STDIN_FILENO, &saved_attributes);
atexit (reset_input_mode);
/* Set the funny terminal modes. */
tcgetattr (STDIN_FILENO, &tattr);
tattr.c_lflag &= ~(ICANON | ECHO); /* Clear ICANON and ECHO. */
tattr.c_cc[VMIN] = 1;
tattr.c_cc[VTIME] = 0;
tcsetattr (STDIN_FILENO, TCSAFLUSH, &tattr);
}

int main ()
{
    int i = 0;
    char c, password[100], asterisk = '*';
    set_input_mode ();
    while (read (STDIN_FILENO, &c, 1) && (isalnum (c) || ispunct (c))
        && i < sizeof (password) - 2)
    {
        password[i++] = c;
        write (STDOUT_FILENO, &asterisk, 1);
    }
    password[i] = 0;
    printf ("\nPassword was: [%s]\n", password);
    return EXIT_SUCCESS;
}

```

Output:

```

mouri@intell10-hub ~/SoftwareEngg/assign1 ./6
*****
Password was: [1234abcs]
mouri@intell10-hub ~/SoftwareEngg/assign1 |

```

### 1.7 Write a program to create your own header file in C.

Program:

```

/*
 * Copyright Rohit Das (C) 2017
 * test.h for 7.c
 */
#ifndef TEST_H
#define TEST_H

int tester;

void test_it_out ();

```

---

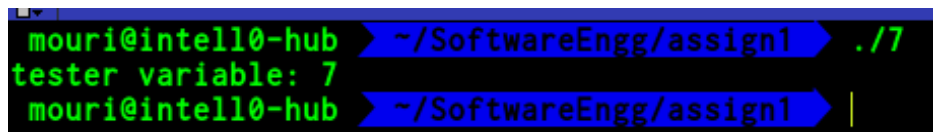
```
#endif
```

```
/*
 * Copyright Rohit Das (C) 2017
 * To create a header file and use it (used test.h)
 */
#include "test.h"
#include <stdio.h>

void test_it_out()
{
    tester = 7;    printf("tester variable: %d\n", tester);
}

int main()
{
    test_it_out();
    return 0;
}
```

Output:



```
mouri@intell0-hub ~/SoftwareEngg/assign1 ./7
tester variable: 7
mouri@intell0-hub ~/SoftwareEngg/assign1 |
```

1.8 Write a program to compare two strings without using library function(strcmp).

Program:

```
/*
 * Copyright Rohit Das (C) 2017
 * To compare two strings without strcmp
 */
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

int check(char* s1, char* s2, int len1, int len2)
{
    if (len1 != len2) {
        printf("Not equal\n");
        return 1;
    }

    for (int i = 0; i < len1; i++) {
        if (s1[i] != s2[i]) {
            printf("Not equal\n");
            return 1;
        }
    }
    printf("Equal\n");
}
```

```

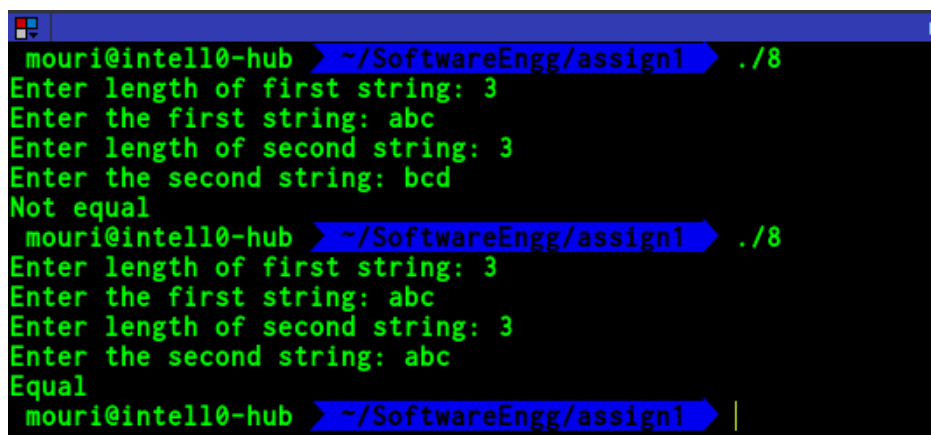
        return 0;
    }

    int main()
    {
        int len1, len2;
        printf("Enter length of first string: ");
        scanf("%d", &len1);
        printf("Enter the first string: ");
        char* s1 = (char *)malloc((len1 + 1) * sizeof(char));
        scanf("%s", s1);
        printf("Enter length of second string: ");
        scanf("%d", &len2);
        printf("Enter the second string: ");
        char* s2 = (char *)malloc((len2 + 1) * sizeof(char));
        scanf("%s", s2);

        check(s1, s2, len1, len2);
        return 0;
    }

```

Output:



```

mouri@intell0-hub > ~/SoftwareEngg/assign1 > ./8
Enter length of first string: 3
Enter the first string: abc
Enter length of second string: 3
Enter the second string: bcd
Not equal
mouri@intell0-hub > ~/SoftwareEngg/assign1 > ./8
Enter length of first string: 3
Enter the first string: abc
Enter length of second string: 3
Enter the second string: abc
Equal
mouri@intell0-hub > ~/SoftwareEngg/assign1 > |

```

1..9 Write a program to print a rectangle using line and special symbols.

```

▲▲▲▲▲▲▲▲▲▲▲▲
▲- - - - -▲
▲- - - - -▲
▲- - - - -▲
▲- - - - -▲
▲- - - - -▲
▲- - - - -▲
▲- - - - -▲
▲▲▲▲▲▲▲▲▲▲▲▲

```

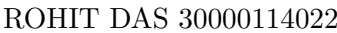
Program:

```

/*
 * Copyright Rohit Das (C) 2017
 * To display pattern as shown
 */

```

Output:



---

### 1..10 Write a program to find addition of lower triangular matrix.

Program:

```
/*
 * Copyright Rohit Das (C) 2017
 * To display addition of lower triangular elements
 */
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int i, j;

    int size;
    printf("Enter size: ");
    scanf("%d", &size);

    int **a = (int **) malloc(size * sizeof(int *));
    for (i = 0; i < size; i++) {
        a[i] = (int *) malloc(size * sizeof(int));
    }

    int **b = (int **) malloc(size * sizeof(int *));
    for (i = 0; i < size; i++) {
        b[i] = (int *) malloc(size * sizeof(int));
    }

    int **c = (int **) malloc(size * sizeof(int *));
    for (i = 0; i < size; i++) {
        c[i] = (int *) malloc(size * sizeof(int));
    }

    printf("Enter the elements for first matrix:\n");
    for (i = 0; i < size; i++) {
        for (j = 0; j < size; j++) {
            scanf("%d", &a[i][j]);
        }
    }

    printf("Enter the elements for second matrix:\n");
    for (i = 0; i < size; i++) {
        for (j = 0; j < size; j++) {
            scanf("%d", &b[i][j]);
        }
    }

    for (i = 0; i < size; i++) {
        for (j = 0; j < size; j++) {
            if (i + j < size - 1) {
                continue;
            } else {
```

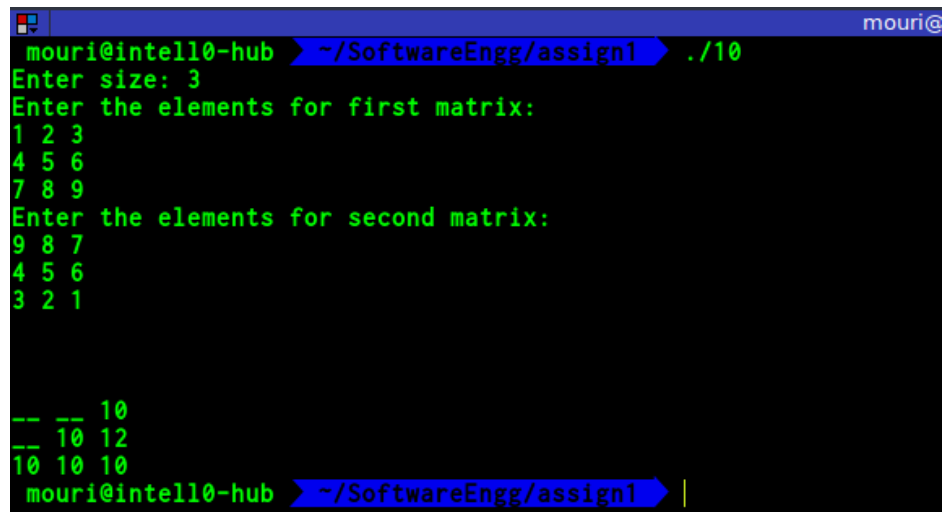
```

        c[i][j] = a[i][j] + b[i][j];
    }
    }
    printf("\n");
}

for (i = 0; i < size; i++) {
    for (j = 0; j < size; j++) {
        if (i + j < size - 1) {
            printf("___ ");
        } else {
            printf("%d ", c[i][j]);
        }
    }
    printf("\n");
}
return 0;
}

```

Output:



```

mouri@intell0-hub > ~/SoftwareEngg/assign1 > ./10
Enter size: 3
Enter the elements for first matrix:
1 2 3
4 5 6
7 8 9
Enter the elements for second matrix:
9 8 7
4 5 6
3 2 1

___ __ 10
___ 10 12
10 10 10
mouri@intell0-hub > ~/SoftwareEngg/assign1 > |

```

---

## 2. Assignment-2: Java programming (Date: 30/10/2017)

2..1 An array contains 20 integers arranged randomly. Write a Java program using quick sort to sort these numbers in ascending order.

Program:

```
/*
 * Copyright Rohit Das (C) 2017
 * To sort an array using Quick Sort
 */
import java.util.*;

class QuickSort
{
    int partition(int arr[], int low, int high) {
        int pivot = arr[high]; int i = (low-1);
        for (int j=low; j<high; j++) {
            if (arr[j] <= pivot) {
                i++;
                // swap arr[i] and arr[j]
                int temp = arr[i];
                arr[i] = arr[j]; arr[j] = temp;
            }
        }
        // swap arr[i+1] and arr[high] (or pivot)
        int temp = arr[i+1];
        arr[i+1] = arr[high]; arr[high] = temp;
        return i+1;
    }

    void sort(int arr[], int low, int high) {
        if (low < high) {
            int part = partition(arr, low, high);
            sort(arr, low, part-1);
            sort(arr, part+1, high);
        }
    }

    /* A utility function to print array of size n */
    static void printArray(int arr[]) {
        int n = arr.length;
        for (int i=0; i<n; ++i)
            System.out.print(arr[i]+" ");
        System.out.println();
    }

    // Driver program
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        int n = 20;
        System.out.println("Enter " + n + " elements:");
    }
}
```



```

        int arr[] = new int[n];
        for (int i = 0; i < n; i++) arr[i] = sc.nextInt();
        QuickSort obj = new QuickSort();
        obj.sort(arr, 0, n-1);
        System.out.println("sorted array");
        printArray(arr);
    }
}

```

Output:

```

mouri@intell0-hub: ~/SoftwareEngg/assign2 $ java QuickSort
Enter 20 elements:
2 1 4 6 20 1 99 45 23 64 47 79 25 24 11 34 69 88 73 20
sorted array
1 1 2 4 6 11 20 20 23 24 25 34 45 47 64 69 73 79 88 99

```

## 2..2 Write a Java program to take an integer not greater than 20 as input and calculate the factorial of the integer recursively.

Program:

```

/*
 * Copyright Rohit Das (C) 2017
 * To calculate factorial of a no. recursively
 */
import java.util.Scanner;

class FactorialDemo{
    public static void main(String args[]){
        //Scanner object for capturing the user input
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the number:");
        //Stored the entered value in variable
        int num = scanner.nextInt();
        //Called the user defined function fact
        int factorial = fact(num);
        System.out.println("Factorial of entered number is: "
            +factorial);
    }
    static int fact(int n)
    {
        int output;
        if(n==1) return 1;
        //Recursion: Function calling itself!!
        output = fact(n-1)* n; return output;
    }
}

```

Output:

```
mouri@intell0-hub: ~/SoftwareEngg/assign2
mouri@intell0-hub > ~/SoftwareEngg/assign2 master ● java FactorialDemo
Enter the number:
15
Factorial of entered number is: 2004310016
mouri@intell0-hub > ~/SoftwareEngg/assign2 master ● java FactorialDemo
Enter the number:
2
Factorial of entered number is: 2
mouri@intell0-hub > ~/SoftwareEngg/assign2 master ● java FactorialDemo
Enter the number:
3
Factorial of entered number is: 6
```

2..3 You are starting out on a long (really long) trip. On the way, there are  $N$  gas stations, the locations of which are given as  $a_1, a_2, \dots, a_N$ . Initially you are located at the gas station at  $a_1$ , and your destination is at location  $a_N$ .

Your car can only store enough fuel to travel atmost  $M$  units without refilling. You can stop at any station and refill the car by any amount. Now you wish to plan your trip such that the number of intermediate stops needed to reach the destination is minimum, and also how many ways are there to plan your trip accordingly.

Input :

The first line two space seperated integers  $N$  and  $M$ .  $N$  lines follow, and the  $i$ th line has the value  $a_i$  ( $0 \leq a_i \leq 1000000000$ ). The input will be such that a solution will always exist.

Output :

Output two space seperated integers : The least number of stops, and the number of ways to plan the trip which uses the least number of stops. Output this value modulo 1000000007.

Program:

```
/*
 * Copyright Rohit Das (C) 2017
 * To calculate min. no. of stops for a car to refuel
 */
import java.io.*;

class Solution_MinStop{
    public static final int MOD = 1000000007;

    public static void main(String[] args)
        throws java.io.IOException {
        BufferedReader rd = new BufferedReader(
            new InputStreamReader(System.in));
        PrintWriter wr = new PrintWriter(System.out);
        System.out.println("Enter no. of stops and max. units of
            dist. to travel:");
        String[] tokens = rd.readLine().split(" ");
        int n = Integer.parseInt(tokens[0]);
        int m = Integer.parseInt(tokens[1]);
        System.out.println("Enter the destinations:");
        int[] array = new int[n];
        for (int i = 0; i < n; i++)
            array[i] = Integer.parseInt(rd.readLine());
        int dp[] = new int[n];    int arr[] = new int[n];
        long counter = 0;
```

```

        int k = 0;          int j = 0;
        dp[0] = 1;
        for (int i = 1; i < n; i++) {
            while (array[i] - array[j] > m) {
                counter -= dp[j];          j++;
            }
            arr[i] = arr[j] + 1;
            while (arr[k] == arr[j]) {
                counter += dp[k];          k++;
            }
            dp[i] += counter % MOD;
        }
        wr.println((arr[n-1]-1) + " " + dp[n-1]);
        wr.close();
    }
}

```

Output:

```

mouri@intell0-hub: ~/SoftwareEngg/assign2 $ java Solution_MinStop
Enter no. of stops and max. units of dist. to travel:
6 3
Enter the destinations:
0
1
3
4
7
10
3 2
mouri@intell0-hub: ~/SoftwareEngg/assign2 $

```

**2..4** Given  $N$  separate integer points on the Cartesian plane satisfying: there is no any three of them sharing a same X-coordinate. Your task is to count the number of rectangles (whose edges parrallel to the axes) created from any four of given points.

**Input:**

There are several test cases (ten at most), each formed as follows:

The first line contains a positive integer  $N$  ( $N \leq 105$ ).  $N$  lines follow, each containing a pair of integers (each having an absolute value of 109 at most) describing coordinates of a given point. The input is ended with  $N = 0$ .

**Output:**

For each test case, output on a line an integer which is the respective number of rectangles found.

Program:

```

/*
 * Copyright Rohit Das (C) 2017
 * To count no/ of rectangles formed by a set of points
 */
import java.io.*;
import java.util.*;
import java.math.*;
import java.lang.*;

```

```

class Point implements Comparable<Point>
{
    int x,y;
    public Point(int x1,int y1) {
        x=x1;    y=y1;
    }
    public int compareTo(Point p) {
        if(p.x<x)    return 1;
        if(p.x>x)    return -1;
        if(p.y<y)    return 1;
        if(p.y>y)    return -1;
        return -1;
    }
}

class Pair
{
    int a,b;
    public Pair(int x,int y) {

        a=x;    b=y;
    }

    @Override
    public boolean equals(Object obj) {
        if (obj == null)    return false;
        if (getClass() != obj.getClass())    return false;
        final Pair other = (Pair) obj;
        if (this.a != other.a)    return false;
        if (this.b != other.b)    return false;
        return true;
    }

    @Override
    public int hashCode() {
        int hash = 3;
        hash = 23 * hash + this.a;
        hash = 23 * hash + this.b;
        return hash;
    }
}

class Main {

    public static void main(String[] args) {
        // TODO code application logic here
        try {
            Parserdoubt pd=new Parserdoubt(System.in);
            while(true) {
                System.out.println("Enter no. of points:");
                int n=pd.nextInt();
                if(n==0) break;
                Point pts[]=new Point[n];
            }
        }
    }
}

```

```

        System.out.println("Enter the points:");
        for(int i=0;i<n;i++)
            pts[i]=new Point(pd.nextInt(),pd.nextInt());
        Arrays.sort(pts);
        HashMap<Pair,Integer> map=new HashMap<Pair,Integer>();
        long counts[]=new long[100000];
        int c=0;
        for(int i=1;i<pts.length;i++) {
            if(pts[i].x==pts[i-1].x) {
                int tmp1=pts[i-1].y;    int tmp2=pts[i].y;
                Pair tmp=new Pair(tmp1,tmp2);
                if(map.containsKey(tmp)) counts[map.get(tmp)]++;
                else    map.put(tmp, c++);
            }
        }
        long sum=0;
        for(int i=0;i<c;i++) {
            long temp=counts[i]*(counts[i]+1);
            temp/=2;    sum+=temp;
        }
        System.out.println(sum);
    }
}
catch(Exception e){
    e.printStackTrace();
}
}
}

class Parserdoubt
{
    final private int BUFFER_SIZE = 1 << 17;
    private DataInputStream din;
    private byte[] buffer;
    private int bufferPointer, bytesRead;

    public Parserdoubt(InputStream in) {
        din = new DataInputStream(in);
        buffer = new byte[BUFFER_SIZE];
        bufferPointer = bytesRead = 0;
    }

    public String nextString() throws Exception {
        StringBuffer sb=new StringBuffer("");
        byte c = read();
        while (c <= ' ') c = read();
        do {
            sb.append(((char)c));    c=read();
        } while(c>' ');
        return sb.toString();
    }
}

```

```

public char nextChar() throws Exception {
    byte c=read();
    while(c<=' ') c= read();
    return (char)c;
}

public int nextInt() throws Exception {
    int ret = 0;
    byte c = read();
    while (c <= ' ') c = read();
    boolean neg = c == '-';
    if (neg) c = read();
    do {
        ret = ret * 10 + c - '0';    c = read();
    } while (c > ' ');
    if (neg) return -ret;
    return ret;
}

public long nextLong() throws Exception {
    long ret = 0;
    byte c = read();
    while (c <= ' ') c = read();
    boolean neg = c == '-';
    if (neg) c = read();
    do{
        ret = ret * 10 + c - '0';    c = read();
    } while (c > ' ');
    if (neg) return -ret;    return ret;
}

private void fillBuffer() throws Exception {
    bytesRead = din.read(buffer, bufferPointer = 0, BUFFER_SIZE);
    if (bytesRead == -1) buffer[0] = -1;
}

private byte read() throws Exception {
    if (bufferPointer == bytesRead) fillBuffer();
    return buffer[bufferPointer++];
}
}

```

Output:



```

mouri@intell0-hub ~/SoftwareEngg/assign2 master ● java Main
Enter no. of points:
6
Enter the points:
7 1
3 5
3 1
1 5
1 1
7 5
3
Enter no. of points:
0
mouri@intell0-hub ~/SoftwareEngg/assign2 master ●

```

---

**2..5** Given two strings, write a program that outputs the shortest sequence of character insertions and deletions that turn one string into another.

Program:

```
/*
 * Copyright Rohit Das (C) 2017
 * To calculate edit distance
 */
import java.util.*;

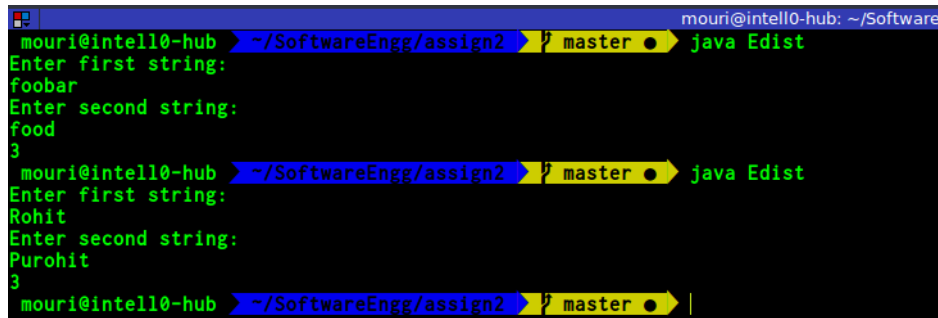
class Edist
{
    static int min(int x,int y,int z)
    {
        if (x<=y && x<=z) return x;
        if (y<=x && y<=z) return y;
        else return z;
    }

    static int editDist(String str1 , String str2 , int m ,int n)
    {
        // If first string is empty, the only option is to
        // insert all characters of second string into first
        if (m == 0) return n;
        // If second string is empty, the only option is to
        // remove all characters of first string
        if (n == 0) return m;
        // If last characters of two strings are same, nothing
        // much to do. Ignore last characters and get count for
        // remaining strings.
        if (str1.charAt(m-1) == str2.charAt(n-1))
            return editDist(str1 , str2 , m-1, n-1);
        // If last characters are not same, consider all three
        // operations on last character of first string, recursively
        // compute minimum cost for all three operations and take
        // minimum of three values.
        return 1 + min ( editDist(str1 , str2 , m, n-1),    // Insert
                        editDist(str1 , str2 , m-1, n),    // Remove
                        editDist(str1 , str2 , m-1, n-1)    // Replace
                    );
    }

    public static void main(String args [])
    {
        Scanner sc = new Scanner(System.in);
        String str1 , str2;
        System.out.println("Enter first string:");
        str1 = sc.next();
        System.out.println("Enter second string:");
        str2 = sc.next();
        System.out.println( editDist( str1 , str2 , str1.length() ,
                                     str2.length()) );
    }
}
```

```
}  
}
```

Output:



```
mouri@intell0-hub: ~/SoftwareEngg/assign2 $ java Edist  
Enter first string:  
foobar  
Enter second string:  
food  
3  
mouri@intell0-hub: ~/SoftwareEngg/assign2 $ java Edist  
Enter first string:  
Rohit  
Enter second string:  
Purohit  
3  
mouri@intell0-hub: ~/SoftwareEngg/assign2 $
```

2..6 Write a java program to generate a 3x3 magic square matrix, where a magic square is a matrix whose all numbers along every row, column and diagonal add up to the same number.

Program:

```
/*  
 * Copyright Rohit Das (C) 2017  
 * To generate a magic square  
 */  
import java.util.*;  
  
class Magic  
{  
    // Function to generate odd sized magic squares  
    static void generateSquare(int n) {  
        int [][] magicSquare = new int[n][n];  
        // Initialize position for 1  
        int i = n/2;    int j = n-1;  
        // One by one put all values in magic square  
        for (int num=1; num <= n*n; ) {  
            if (i== -1 && j==n) {  
                j = n-2;    i = 0;  
            }  
            else {  
                if (j == n)    j = 0;  
                if (i < 0)    i = n-1;  
            }  
            if (magicSquare[i][j] != 0)  
            {  
                j -= 2;    i++;  
                continue;  
            }  
            else    magicSquare[i][j] = num++;  
            j++;    i--;  
        }  
    }  
}
```



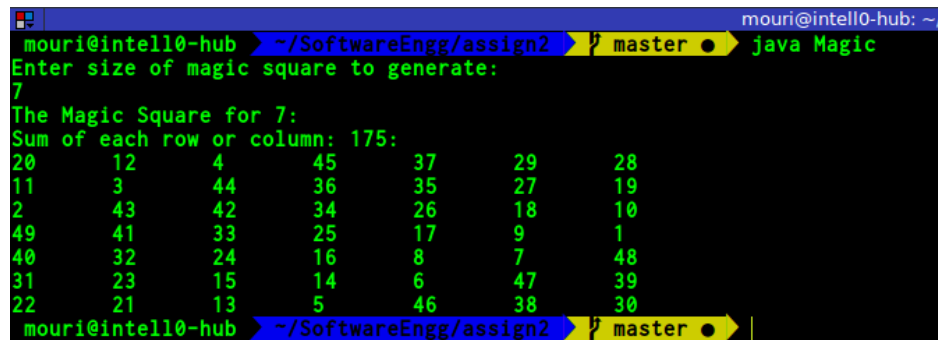
```

        // print magic square
        System.out.println("Magic Square for "+n+":");
        System.out.println("Sum of each row/column: "+n*(n*n+1)/2+":");
        for(i=0; i<n; i++) {
            for(j=0; j<n; j++) System.out.print(magicSquare[i][j]+"\\t");
            System.out.println();
        }
    }

    // driver program
    public static void main (String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter size of magic square to generate:");
        int n;    n = sc.nextInt();
        generateSquare(n);
    }
}

```

Output:



```

mouri@intell0-hub > ~/SoftwareEngg/assign2 > master ● java Magic
Enter size of magic square to generate:
7
The Magic Square for 7:
Sum of each row or column: 175:
20    12    4    45    37    29    28
11    3    44    36    35    27    19
2    43    42    34    26    18    10
49    41    33    25    17    9    1
40    32    24    16    8    7    48
31    23    15    14    6    47    39
22    21    13    5    46    38    30
mouri@intell0-hub > ~/SoftwareEngg/assign2 > master ● |

```