

**KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY**

[Deemed to be University](https://en.wikipedia.org/wiki/Kalinga_Institute_of_Industrial_Technology)

**School of Computer Science and Engineering**

Assignment-II

**Submitted By:-**

**Name:** Abhishikt Mohanty **Roll No:** 21052991

**Section:** CSE-40 **Branch:** Computer Science and Engineering **Subject:** DSA LAB

**Assignment 2 | 02/08/2022 | DSA Lab**

Q1. WAP to store n employees data such as employee name, gender, designation, department, basic pay. Calculate the gross pay of each employees as follows:

Gross pay-basic pay + HR + DA

HR=25% of basic, DA=75% of basic.

Code :-

#include <stdio.h>

#include <stdlib.h>

struct employee

{

    char name[100];

    char gender[20];

    char designation[100];

    char department[100];

    int basic\_pay;

    int gross\_pay;

};

void gross(struct employee emp[], int n)

{

    int i;

    for (i = 0; i < n; i++)

    {

        emp[i].gross\_pay = 0;

        int hr = (25 \* emp[i].basic\_pay) / 100;

        int da = (75 \* emp[i].basic\_pay) / 100;

        emp[i].gross\_pay = emp[i].basic\_pay + hr + da;

    }

}

int main()

{

    int n;

    printf("Enter the number of employess : ");

    scanf("%d", &n);

    struct employee emp[n];

    int i;

    for (i = 0; i < n; i++)

    {

        printf("Enter the details of employee-%d:\n", i+1);

        printf("Enter the Name : ");

        scanf("%s", &emp[i].name);

        printf("Enter the Gender : ");

        scanf("%s", &emp[i].gender);

        printf("Enter the Designation : ");

        scanf("%s", &emp[i].designation);

        printf("Enter the Department : ");

        scanf("%s", &emp[i].department);

        printf("Enter the Basic Salary : ");

        scanf("%d", &emp[i].basic\_pay);

        printf("\n");

    }

    gross(emp, n);

    for (i = 0; i < n; i++)

    {

        printf("\nDetails of employee-%d\n", i+1);

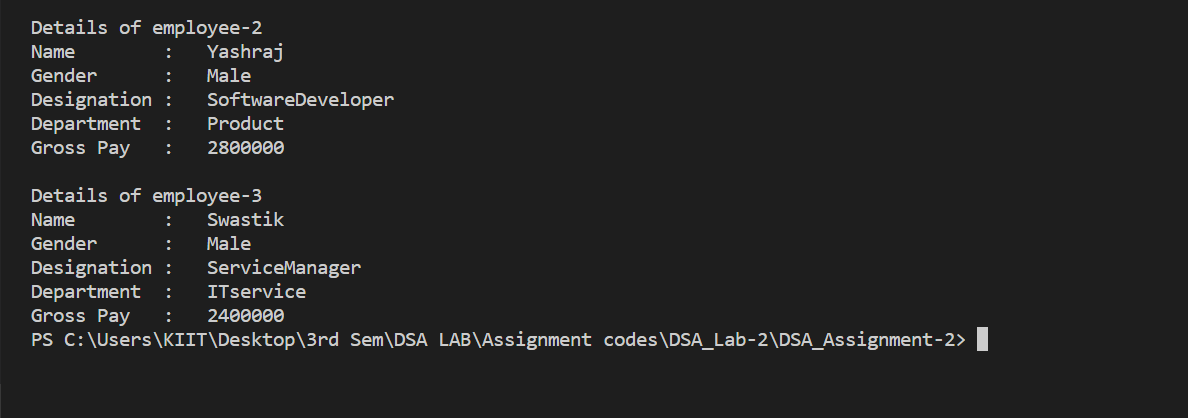
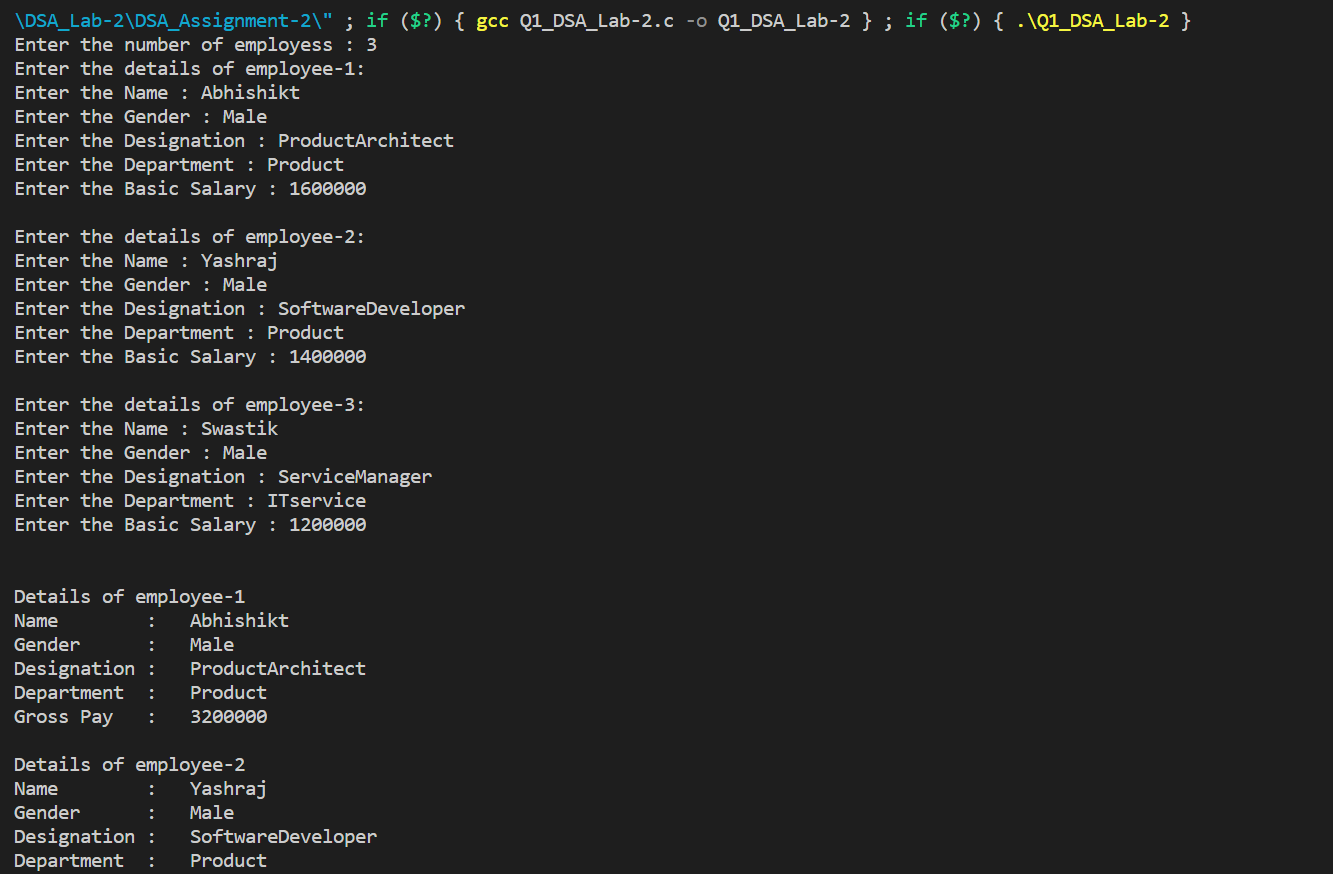
        printf("Name        :\t%s\nGender      :\t%s\nDesignation :\t%s\nDepartment  :\t%s\nGross Pay   :\t%d\n", emp[i].name, emp[i].gender, emp[i].designation, emp[i].department, emp[i].gross\_pay);

    }

    return 0;

}

Output:-



Q2. WAP to add two distances (in km-meter) by passing structure to a function.

Code :-

#include <stdio.h>

struct Distance

{

  int km;

  float mtr;

} firstDistance, secondDistance, sum;

int main()

{

  printf("Enter kilo-meter and meter(up to 2 digit places) for the first distance with a space : ");

  scanf("%d %f", &firstDistance.km, &firstDistance.mtr);

  printf("Enter kilo-meter and meter(up to 2 digit places) for the second distance with a space : ");

  scanf("%d %f", &secondDistance.km, &secondDistance.mtr);

  sum.km = firstDistance.km + secondDistance.km;

  sum.mtr = firstDistance.mtr + secondDistance.mtr;

  while (sum.mtr >= 100)

  {

    sum.mtr = sum.mtr - 100;

    sum.km++;

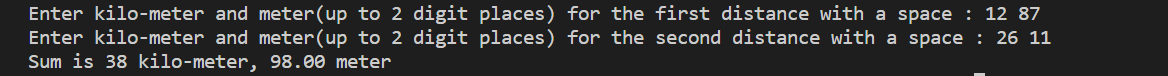
  }

  printf("Sum is %d kilo-meter, %.2f meter\n", sum.km, sum.mtr);

  return 0;

}

Output:-



Q3. Add two complex numbers by passing structures to a function

Code:-

#include <stdio.h>

typedef struct complex

{

    float r;

    float i;

} complex;

complex addition(complex num1, complex num2);

int main()

{

    complex num1, num2, value;

    printf("Enter real and imaginary parts of first complex number: ");

    scanf("%f %f", &num1.r, &num1.i);

    printf("Enter real and imaginary parts of second complex number: ");

    scanf("%f %f", &num2.r, &num2.i);

    value = addition(num1, num2);

    printf("Addition of given complex number is %.1f + %.1fi", value.r, value.i);

    return 0;

}

complex addition(complex num1, complex num2)

{

    complex temp;

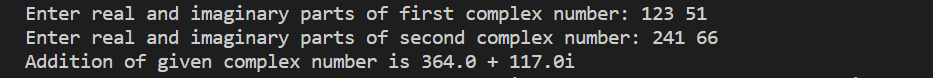
    temp.r = num1.r + num2.r;

    temp.i = num1.i + num2.i;

    return (temp);

}

Output:-



Q4. Calculate the difference between two time periods using structure

Code:-

#include<stdio.h>

struct time

{

    int hours;

    int minutes;

    int seconds;

};

int main()

{

    struct time first, second, diff;

    printf("Enter the first time in 24 hour format:\n");

    printf("Enter hours, minutes and seconds respectively(hh mm ss): ");

    scanf("%d %d %d", &first.hours,&first.minutes, &first.seconds);

    printf("Enter the second time in 24 hour format:\n");

    printf("Enter hours, minutes and seconds respectively(hh mm ss): ");

    scanf("%d %d %d", &second.hours,&second.minutes, &second.seconds);

    if(first.seconds > second.seconds)

    {

        second.seconds += 60;

        --second.minutes;

    }

    if(first.minutes > second.minutes)

    {

        second.minutes += 60;

        --second.hours;

    }

    diff.seconds = second.seconds - first.seconds;

    diff.minutes = second.minutes - first.minutes;

    diff.hours = second.hours - first.hours;

    printf("\nTIME PERIOD : %d:%d:%d - ", first.hours, first.minutes, first.seconds);

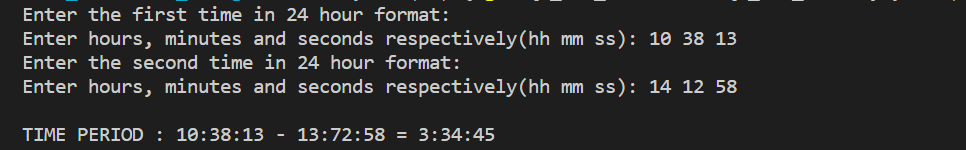
    printf("%d:%d:%d ", second.hours, second.minutes, second.seconds);

    printf("= %d:%d:%d\n", diff.hours, diff.minutes, diff.seconds);

    return 0;

}

Output:-



Q5. Store information of n students using structures and Dynamic Memory Allocation.

Code:-

#include <stdio.h>

#include <stdlib.h>

struct student

{

  char name[100];

  int roll\_number;

  int class;

  char section;

};

int main()

{

  int n;

  printf("Enter the number of students : ");

  scanf("%d", &n);

  struct student \*ptr;

  ptr = (struct student \*)malloc(n \* sizeof(struct student));

  for (int i = 0; i < n; i++)

  {

    printf("\nEnter the deatil of student %d\n", i + 1);

    printf("Enter the name : ");

    scanf("%s", &(ptr + i)->name);

    printf("Enter the Roll number : ");

    scanf("%d", &(ptr + i)->roll\_number);

    printf("Enter the class (in number) : ");

    scanf("%d", &(ptr + i)->class);

    printf("Enter the section : ");

    scanf("%s", &(ptr + i)->section);

  }

  printf("\nDetails of the Students: \n");

  printf("\n");

  for (int i = 0; i < n; i++)

  {

    printf("Student %d", i + 1);

    printf("\nName        : %s", (ptr + i)->name);

    printf("\nRoll Number : %d", (ptr + i)->roll\_number);

    printf("\nClass       : %d", (ptr + i)->class);

    printf("\nSection     : %c", (ptr + i)->section);

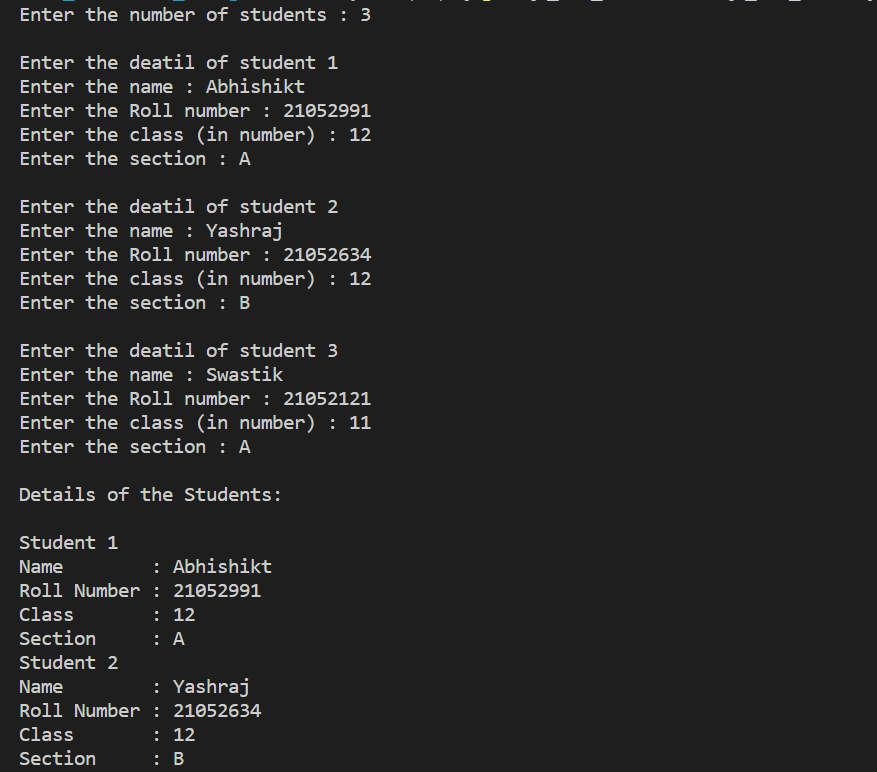
    printf("\n");

  }

  return 0;

}

Output:-



Q6. C program to read a one dimensional array, print sum of all elements along with inputted array elements using Dynamic Memory Allocation.

Code:-

#include <stdio.h>

#include <stdlib.h>

struct array

{

    int a;

    int sum;

};

int main()

{

    int n;

    printf("Enter the length of the array : ");

    scanf("%d", &n);

    struct array \*ptr;

    ptr = (struct array \*)malloc(n \* sizeof(struct array));

    printf("Enter the elements of the array : ");

    for (int i = 0; i < n; i++)

    {

        scanf("%d", &(ptr + i)->a);

    }

    ptr->sum = 0;

    for (int i = 0; i < n; i++)

    {

        ptr->sum += (ptr + i)->a;

    }

    printf("Elements of the array : ");

    for (int i = 0; i < n; i++)

    {

        printf("%d  ", (ptr + i)->a);

    }

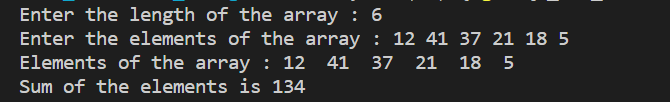
    printf("\n");

    printf("Sum of the elements is %d", ptr->sum);

    return 0;

}

Output:-



Q7. WAP using C to Evaluate the Given Polynomial Equation f(x). Note: Order of polynomial, co-efficient and value of x will be user input.

Code:-

#include <stdio.h>

#include <stdlib.h>

#define MAXSIZE 10

void main()

{

  int a[MAXSIZE];

  int i, N, power;

  float x, polySum;

  printf("Enter the order of the polynomial\n");

  scanf("%d", & N);

  printf("Enter the value of x\n");

  scanf("%f", & x);

  printf("Enter %d coefficients\n", N + 1);

  for (i = 0; i <= N; i++) {

    scanf("%d", & a[i]);

  }

  polySum = a[0];

  for (i = 1; i <= N; i++) {

    polySum = polySum \* x + a[i];

  }

power = N;

    printf("Given polynomial is: \n");

    for (i = 0; i <= N; i++)

    {

        if (power < 0)

        {

            break;

        }

        if (a[i] > 0 & i!=0)

            printf(" + ");

        else if (a[i] < 0)

            printf(" - ");

        else

            printf(" ");

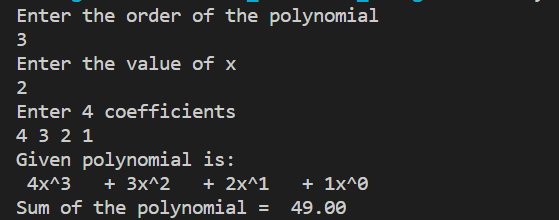
        printf("%dx^%d  ", abs(a[i]), power--);

  }

  printf("\nSum of the polynomial = %6.2f\n", polySum);

}

Output:-



Q8. WAP using function that adds given two polynomials f(x) = h(x) + g(x)

Code:-

#include <stdio.h>

struct poly

{

    int coeff;

    int expo;

};

struct poly p1[10], p2[10], p3[10];

int readPoly(struct poly[]);

int addPoly(struct poly[], struct poly[], int, int, struct poly[]);

void displayPoly(struct poly[], int terms);

int main()

{

    int t1, t2, t3;

    t1 = readPoly(p1);

    printf("\nFirst polynomial : \n");

    displayPoly(p1, t1);

    t2 = readPoly(p2);

    printf("\nSecond polynomial : \n");

    displayPoly(p2, t2);

    t3 = addPoly(p1, p2, t1, t2, p3);

    printf("\n\nResultant polynomial after addition : \n");

    displayPoly(p3, t3);

printf("\n");

    return 0;

}

int readPoly(struct poly p[10])

{

    int t1, i;

    printf("\n\nEnter the total number of terms in the polynomial: ");

    scanf("%d", &t1);

    printf("\nEnter the COEFFICIENT and EXPONENT in DESCENDING ORDER\n");

    for (i = 0; i < t1; i++)

    {

        printf("Enter the Coefficient(%d): ", i + 1);

        scanf("%d", &p[i].coeff);

        printf("Enter the exponent(%d): ", i + 1);

        scanf("%d", &p[i].expo);

    }

    return (t1);

}

int addPoly(struct poly p1[10], struct poly p2[10], int t1, int t2, struct poly p3[10])

{

    int i, j, k;

    i = 0;

    j = 0;

    k = 0;

    while (i < t1 && j < t2)

    {

        if (p1[i].expo == p2[j].expo)

        {

            p3[k].coeff = p1[i].coeff + p2[j].coeff;

            p3[k].expo = p1[i].expo;

            i++;

            j++;

            k++;

        }

        else if (p1[i].expo > p2[j].expo)

        {

            p3[k].coeff = p1[i].coeff;

            p3[k].expo = p1[i].expo;

            i++;

            k++;

        }

        else

        {

            p3[k].coeff = p2[j].coeff;

            p3[k].expo = p2[j].expo;

            j++;

            k++;

        }

    }

    while (i < t1)

    {

        p3[k].coeff = p1[i].coeff;

        p3[k].expo = p1[i].expo;

        i++;

        k++;

    }

    while (j < t2)

    {

        p3[k].coeff = p2[j].coeff;

        p3[k].expo = p2[j].expo;

        j++;

        k++;

    }

    return (k);

}

void displayPoly(struct poly p[10], int term)

{

int k;

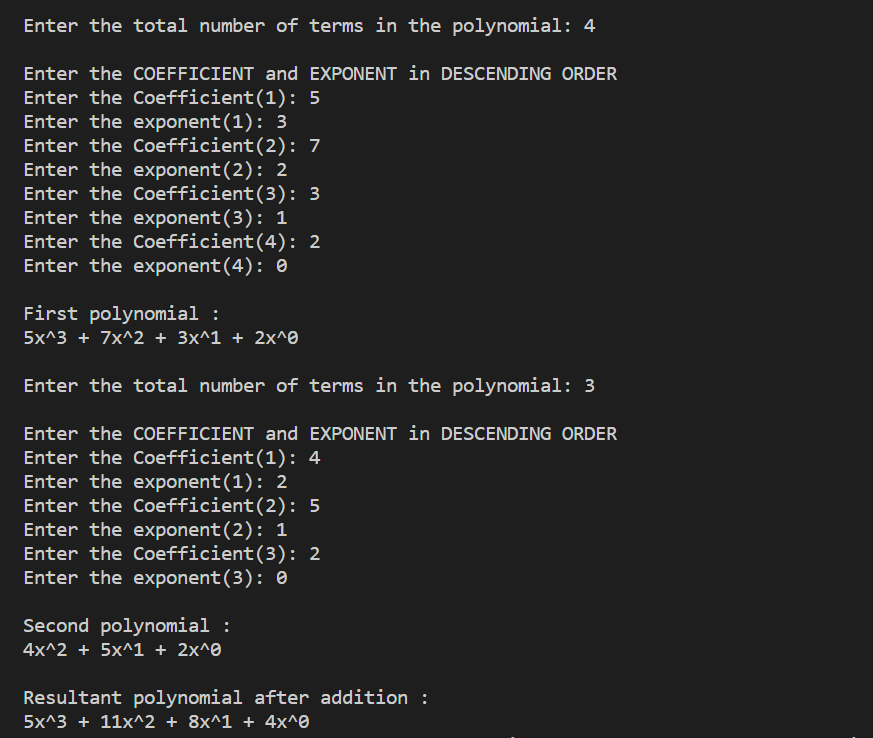
    for (k = 0; k < term - 1; k++)

        printf("%dx^%d + ", p[k].coeff, p[k].expo);

    printf("%dx^%d ", p[term - 1].coeff, p[term - 1].expo);

}

Output:-



Q9. WAP to check whether the given matrix is sparse matrix or not.

Code:-

#include <stdio.h>

#include <stdlib.h>

int main()

{

    int row, col, i, j, a[10][10], count = 0;

    printf("Enter number of row: ");

    scanf("%d", &row);

    printf("Enter number of Column: ");

    scanf("%d", &col);

    printf("Enter element of the Matrix: \n");

    for (i = 0; i < row; i++)

    {

        for (j = 0; j < col; j++)

        {

            scanf("%d", &a[i][j]);

        }

    }

    printf("Elements are:\n");

    for (i = 0; i < row; i++)

    {

        for (j = 0; j < col; j++)

        {

            printf("%d\t", a[i][j]);

        }

        printf("\n");

    }

    for (i = 0; i < row; i++)

    {

        for (j = 0; j < col; j++)

        {

            if (a[i][j] == 0)

                count++;

        }

    }

    if (count > ((row \* col) / 2))

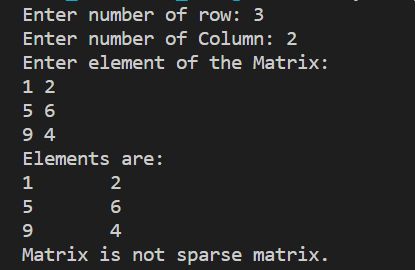
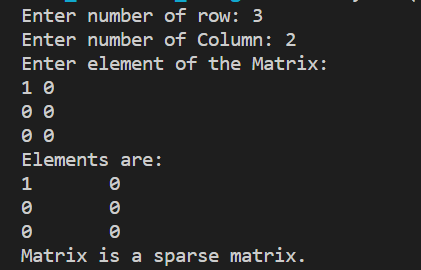
        printf("Matrix is a sparse matrix.\n");

    else

        printf("Matrix is not sparse matrix.\n");

}

Output:- Output:-

**Home-Assignment 2 | 02/08/2022 | DSA Lab**

Q1. WAP to print all permutations of a given string using pointers.

Code:-

#include <stdio.h>

#include <string.h>

void swap(char \*a,char\*b)

{

    char temp;

    temp = \*a;

    \*a = \*b;

    \*b = temp;

};

void per(char \*c,int s1,int s2)

{

    int i;

    if(s1==s2)

    {

        printf("%s\n",c);

    }

    else

    {

        for(i=s1;i<=s2;i++)

        {

            swap((c+s1),(c+i));

            per(c,s1 + 1,s2);

            swap((c+i),(c+s1));

        }

    }

};

int main()

{

    char ch[100];

    printf("Enter the word (limit of 100) : ");

    scanf("%s",&ch);

    int n = strlen(ch);

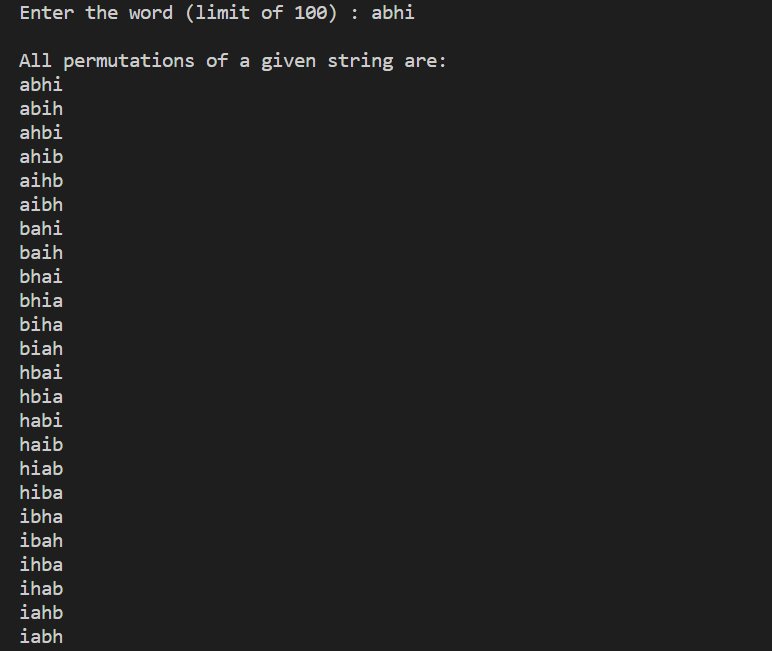
    printf("\nAll permutations of a given string are: \n");

    per(ch,0,n-1);

    return 0;

}

Output:-



Q2. WAP to arrange the elements of an array such that all even numbers are

followed by all odd numbers.

Code:-

#include <stdio.h>

int main(){

    int a[100],b[100],i,n,j,k,temp,c=0;

    printf("Enter number of elements in the  array: ");

    scanf("%d", &n);

    printf("Enter the elements in array: ");

    for(i=0; i<n; i++){

        scanf("%d",&a[i]);

        if(a[i]%2==1)

         c++;

    }

    for(i=0; i<n-1; i++){

        for(j=0; j<n-i-1; j++){

           if(a[j]>a[j+1]){

            temp=a[j];

            a[j]=a[j+1];

            a[j+1]=temp;

           }

        }

    }

    k=0;

    j=n-c;

    for(i=0; i<n; i++){

        if(a[i]%2==0){

            if(k<n-c)

              b[k++]=a[i];

        }

        else{

            if(j<n)

              b[j++]=a[i];

        }

    }

    printf("\nArranged the elements of the array such that all even numbers are followed by all odd numbers:\n");

    for(i=0; i<n; i++){

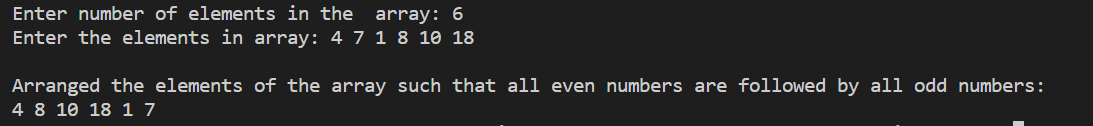
        a[i]=b[i];

        printf("%d ",a[i]);

    }

 }

Output:-



Q3. WAP to find the transpose of a matrix.

Code:-

#include <stdio.h>

int main()

{

    int a[10][10], transpose[10][10], r, c;

    printf("Rows of the martix : ");

    scanf("%d", &r);

    printf("coloumn of the martix : ");

    scanf("%d", &c);

    printf("\nEnter matrix elements:\n");

    for (int i = 0; i < r; ++i)

        for (int j = 0; j < c; ++j)

        {

            scanf("%d", &a[i][j]);

        }

    for (int i = 0; i < r; ++i)

        for (int j = 0; j < c; ++j)

        {

            transpose[j][i] = a[i][j];

        }

    printf("\nTranspose of the matrix:\n");

    for (int i = 0; i < c; ++i)

        for (int j = 0; j < r; ++j)

        {

            printf("%d  ", transpose[i][j]);

            if (j == r - 1)

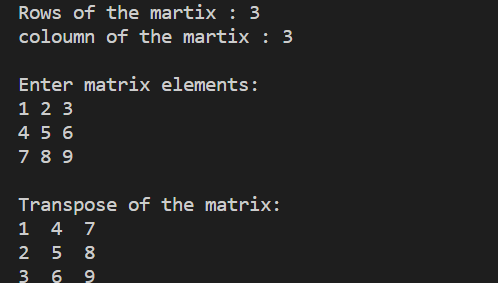
                printf("\n");

        }

    return 0;

}

Output:-



Q4. WAP to find determinant of 3×3 Matrix.

Code:-

#include <stdio.h>

int main()

{

    int row = 3, col = 3;

    int a[row][col];

    int tr[row][col];

    printf("Enter the 3 X 3 Matrix : \n");

    for (int i = 0; i < row; i++)

    {

        for (int j = 0; j < col; j++)

        {

            scanf("%d", &a[i][j]);

        }

    }

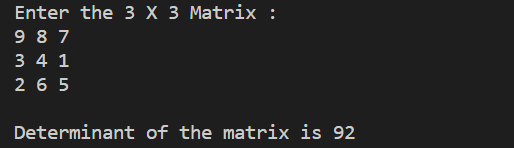
    int det = a[0][0] \* (a[1][1] \* a[2][2] - a[1][2] \* a[2][1]) - a[0][1] \* (a[1][0] \* a[2][2] - a[1][2] \* a[2][0]) + a[0][2] \* (a[1][0] \* a[2][1] - a[1][1] \* a[2][0]);

    printf("\nDeterminant of the matrix is %d", det);

    return 0;

}

Output:-



Q5. WAP to Find Largest Element in an Array using Recursion.

Code:-

#include <stdio.h>

int max(int a, int b)

{

    return a > b ? a : b;

}

int findmax(int A[], int n)

{

    if (n == 1)

    {

        return A[0];

    }

    return max(A[n - 1], findmax(A, n - 1));

}

int main()

{

    int n;

    printf("Enter number of elements in the array : ");

    scanf("%d", &n);

    int a[n];

    printf("Enter the elements of array : ");

    for (int i = 0; i < n; i++)

    {

        scanf("%d", &a[i]);

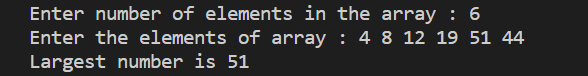
    }

    int size = sizeof(a) / sizeof(a[0]);

    printf("Largest number is %d", findmax(a, size));

}

Output:-



Q6. WAP using function to find frequency of occurrence of numbers in an array.

Code:-

#include<stdio.h>

#include<stdlib.h>

void occurance(int n, int freq[], int a[]){

    int Count;

    for (int i = 0; i < n; i++)

    {

        Count = 1;

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

        {

            if (a[i] == a[j])

            {

                Count++;

                freq[j] = 0;

            }

        }

        if (freq[i] != 0)

        {

            freq[i] = Count;

        }

    }

    printf("\nFrequency is \n");

    for (int i = 0; i < n; i++)

    {

        if (freq[i] != 0)

        {

            printf("%d Occurs %d Times \n", a[i], freq[i]);

        }

    }

}

int main()

{

    int n;

    printf("Enter the length of the array : ");

    scanf("%d",&n);

    int a[n],freq[n];

    printf("Enter elements of the array : ");

    for(int i=0;i<n;i++)

    {

        scanf("%d",&a[i]);

        freq[i] = -1;

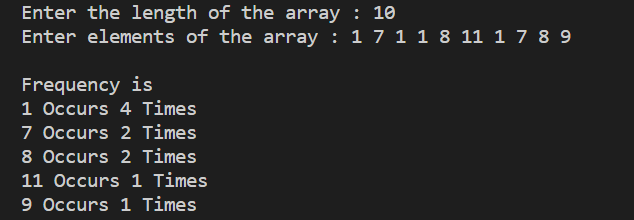
    }

    occurance(n, freq, a);

    return 0;

}

Output:-



Q7. WAP to determine whether the given matrix is a lower triangular or upper triangular or tri-diagonal matrix.

Code:-

#include <stdio.h>

void lower(int row, int col, int a[row][col])

{

    int flag = 0;

    int i, j;

    for (i = 0; i < row; i++)

    {

        for (j = 0; j < col; j++)

        {

            if (i < j)

            {

                if (a[i][j] != 0)

                {

                    flag = -1;

                }

            }

        }

    }

    if (flag == 0)

    {

        printf("The matrix is Lower Matrix.");

    }

    else

    {

        printf("The matrix is not Lower Matrix.");

    }

    printf("\n");

};

void upper(int row, int col, int a[row][col])

{

    int flag = 0;

    int c = col - 1;

    for (int i = 0; i < row; i++)

    {

        for (int j = c--; j < col; j++)

        {

            if (i > j)

            {

                if (a[i][j] != 0)

                {

                    flag = -1;

                }

            }

        }

    }

    if (flag == 0)

    {

        printf("The matrix is Upper Matrix.");

    }

    else

    {

        printf("The matrix is not Upper Matrix.");

    }

    printf("\n");

};

void tri(int row, int col, int a[row][col])

{

    int flag = 1;

    for (int i = 0; i < row; i++)

    {

        for (int j = 0; j < col; j++)

        {

            if (i < j)

            {

                if (a[i][j] != 0)

                {

                    flag = -1;

                }

            }

            if (i > j)

            {

                if (a[i][j] != 0)

                {

                    flag = -1;

                }

            }

        }

}

    if (flag == 0)

    {

        printf("The matrix is Tri-diagonal Matrix.");

    }

    else

    {

        printf("The matrix is not a tri-diagonal Matrix.");

    }

    printf("\n");

};

int main()

{

    int col, row;

    printf("Enter the number of row: ");

    scanf("%d", &row);

    printf("Enter the number of coloumn: ");

    scanf("%d", &col);

    int a[row][col];

    printf("Enter the Matrix : \n");

    for (int i = 0; i < row; i++)

    {

        for (int j = 0; j < col; j++)

        {

            scanf("%d", &a[i][j]);

        }

    }

    lower(row, col, a);

    upper(row, col, a);

    tri(row, col, a);

    return 0;

}

Output:-

