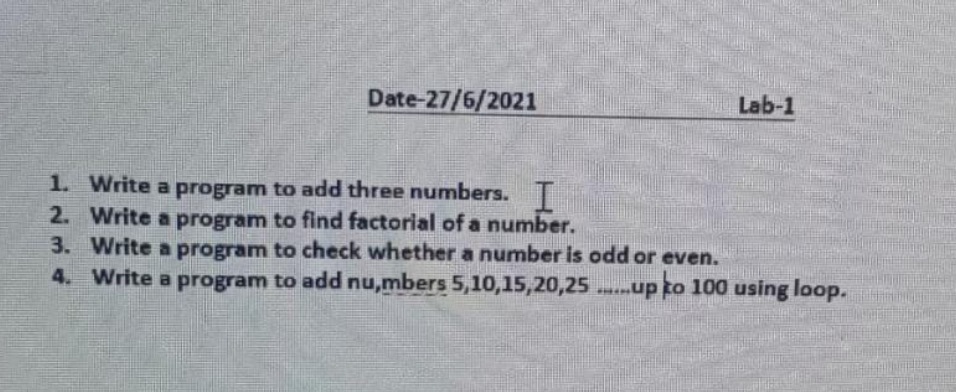
Data Structure and Algorithms

(Mid-sem submission )

**Lab 1**

**Date:- 26\_7\_2021**

**Q.**

1.

Code:-

// In 1 st lab teacher said to code in language in which we are COMFORTABLE so therefore only lab1 is in c++ language

#include<iostream> //By Anusthan Singh (20051337)

using namespace std;

int addition(int a, int b, int c);

int main(){

    float num1,num2,num3,add;

    cout<<"Enter any threee numbers =\n";

    cin>>num1>>num2>>num3;

    add=addition(num1,num2,num3);

    cout<<"Adition = "<<add;

return 0;

}

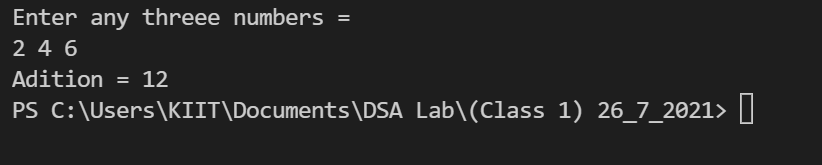
int addition(int a, int b, int c){

    float sum =(a+b+c);

    return sum;

}

Output:-



Q2.

#include<iostream>

using namespace std; //By Anusthan Singh (20051337)

int factorical(int a);

int main()

{

     int fact=1,n;

     cout<<"Enter any Number =\n";

     cin>>n;

     fact= factorical(n);

    cout<<"Factorial of  is= "<<fact<<endl;

  return 0;

}

int factorical(int a){

    int i,fact=1;

for(i=1;i<=a;i++){

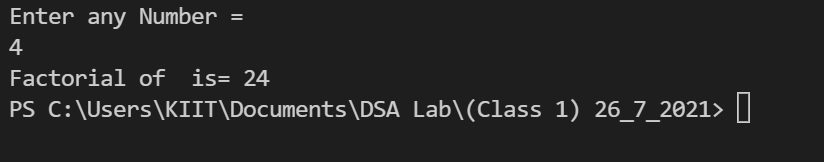
      fact=fact\*i;

     }

     return fact;

}

Output:-



Q3.

#include<iostream>//By Anusthan Singh (20051337)

using namespace std;

int even\_odd(int num);

int main(){

    int n, ans;

    cout<<"Enter any number= \n";

    cin>>n;

    ans= even\_odd(n);

    cout<<ans<<"\n (Where 0 represent even & -1 represent odd) ";

    return 0;

    }

    int even\_odd(int num){

        if(num%2==0){

            return 0;

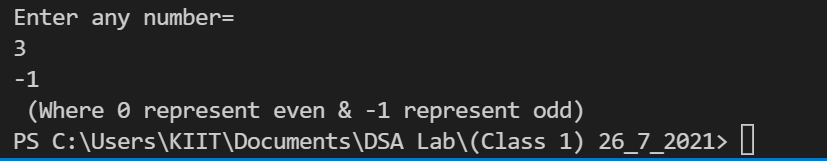
        }

        else

        return -1;

    }

Output:-



Q4.

#include<iostream>//By Anusthan Singh (20051337)

using namespace std;

int main()

{

    int sum=0,n;

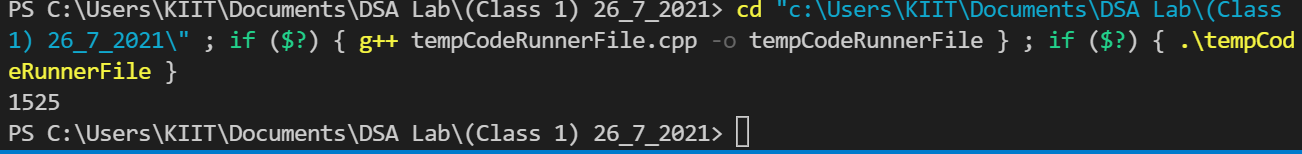
    for(int i=1;i<=100;i=i\*5){

    cout<<i;

    }

    return 0;

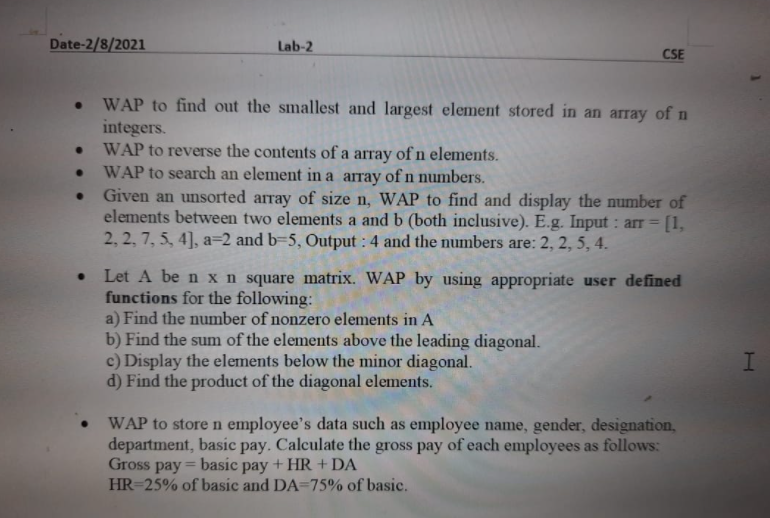
}

Output:-

Lab 2

Date:- 2/8/2021

Q



1.

Code:-

#include<stdio.h>//By Anusthan Singh (20051337)

int main()

{

int a[50],i,n,large,small;

printf("\nEnter the number of elements :\n");

scanf("%d",&n);

printf("\nInput the array elements :\n");

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

scanf("%d",&a[i]);

large=small=a[0];

 for(i=1;i<n;++i)

{

 if(a[i]>large)

large=a[i];

 if(a[i]<small)

small=a[i];

}

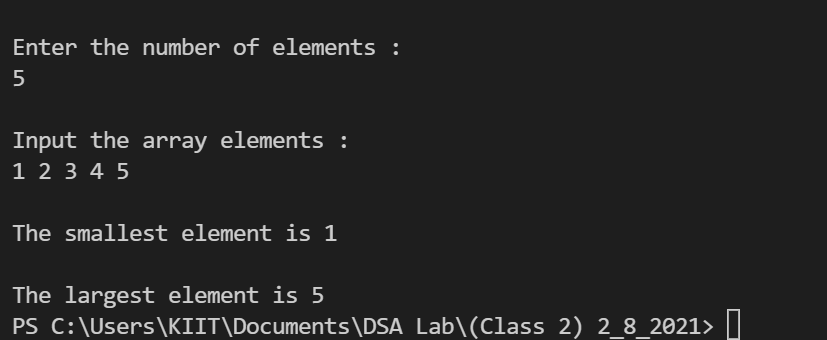
printf("\nThe smallest element is %d\n",small);

printf("\nThe largest element is %d\n",large);

return 0;

}

Output:-



Q2.

#include <stdio.h>

#include <stdlib.h>

int main(void)

{

    int n;

    printf("Enter the no of elements int the array = ");

    scanf("%d",&n);

    int arr[n];

    printf("\n Enter the elements of array = \n");

    for(int i=0;i<n;i++)

     {scanf("%d",&arr[i]);}

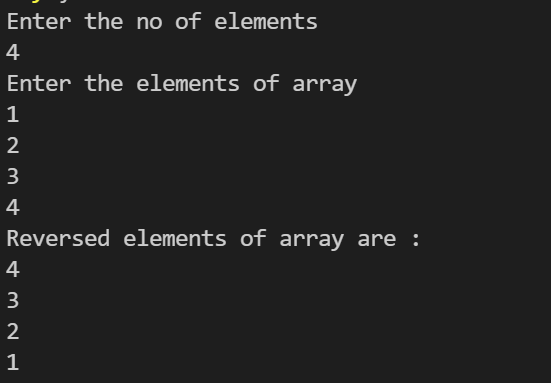
    printf("Reversed format of the  array are = \n");

    for(int i=n-1;i>=0;i--)

     {printf("%d \n",arr[i]);}

}

Output:-



Q3.

Code:-

#include <stdio.h>//By Anusthan Singh (20051337)

#include <conio.h>

int main()

{

    int a[10000],i,n,found;

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

    scanf("%d", &n);

    printf("Input  elements in array : ");

    for(i=0; i<n; i++)

    {

        scanf("%d",&a[i]);

    }

     printf("Enter the number to find : ");

    scanf("%d", &found);

    for(i=0; i<n; i++)

    {

        if(a[i]==found)

        {

            printf("Element found = ");

             printf("%d",found);

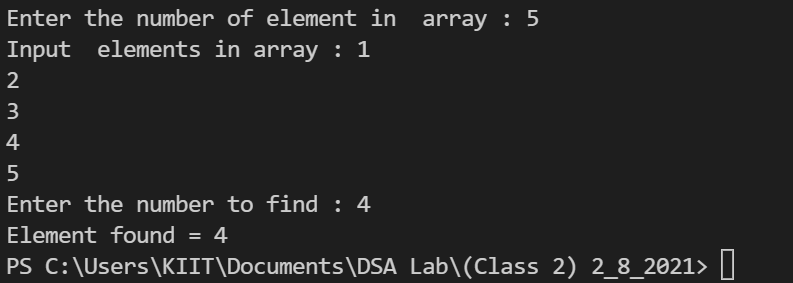
            return 0;

        }

    }

    }

Output:-



Q4.

#include <stdio.h>//By Anusthan Singh (20051337)

#include <conio.h>

int main()

{

    int a[10000],i,j,n,c,b;

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

    scanf("%d", &n);

    printf("Input  elements in array : ");

    for(i=0; i<n; i++)

    {

        scanf("%d",&a[i]);

    }

     printf("Enter the number that is your 'a' element : ");

    scanf("%d", &c);

     printf("Enter the number that is your 'b'  element : ");

    scanf("%d", &b);

    for(i=0; i<n; i++)

    {

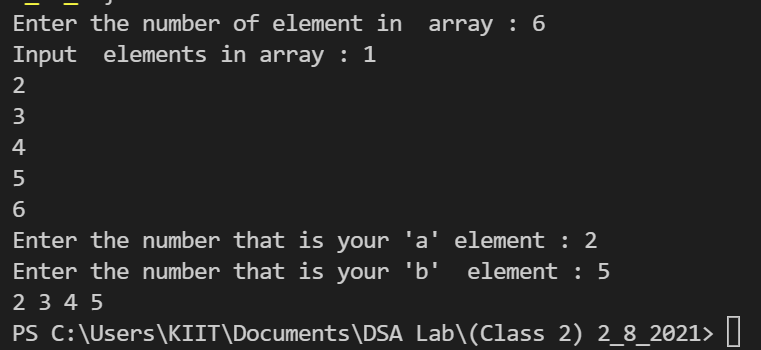
        if(a[i]>=c && a[i]<=b)

         printf("%d ", a[i]);

    }

    }

Output:-



Q5

#include<stdio.h>//By Anusthan Singh (20051337)

#include<stdlib.h>

int main()

{

    int n,diagonal\_sum=0,no\_zero=0,multiplication=1;

    printf("Enter the size of 2d array \n");

    scanf("%d",&n);

     int \*\*arr = (int \*\*)malloc(n \* sizeof(int \*));

    for (int i=0; i<n; i++)

         arr[i] = (int \*)malloc(n \* sizeof(int));

    printf("Enter the elements of the 2d array \n");

    for (int i = 0; i < n; i++)

    {

        for (int j = 0; j < n; j++)

        {

            scanf("%d",&arr[i][j]);

        }

    }

    for (int i = 0; i < n; i++)

    {

        for (int j = 0; j < n; j++)

        {

            if(arr[i][j]!=0) no\_zero++;

        }

    }

    for (int i = 0; i <n; i++)

    {

        for (int j = 0; j <n; j++)

        {

            if(j>i) diagonal\_sum+=arr[i][j];

        }

    }

    for (int i = 0,j=0; i < n; i++,j++)

    {

        multiplication\*=arr[i][j];

    }

    printf("The elements below the minor diagonal = \n");

    for (int i = 0; i < n; i++)

    {

        for (int j = 0; j < n; j++)

        {

            if(j<i) printf("%d ",arr[i][j]);

            else printf("- ");

        }

        printf("\n");

    }

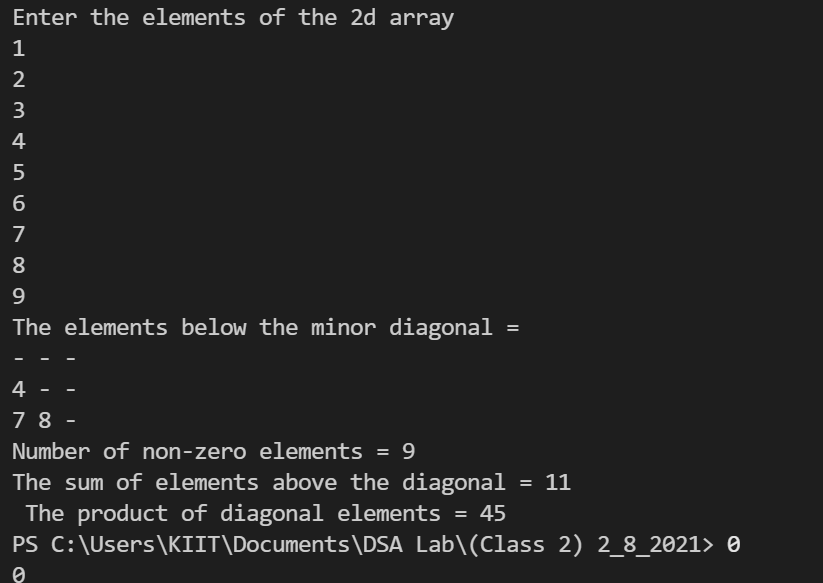
    printf("Number of non-zero elements = %d\n",no\_zero);

    printf("The sum of elements above the diagonal = %d\n ",diagonal\_sum);

      printf("The product of diagonal elements = %d\n",multiplication);

}

Output:-



Q6

#include <stdio.h>

struct empl

{

    int id,basic;

    char name[100], desig[100], dept[100];

    char gender;

    float gross;

};

int main()

{

    float da, ha;

    struct empl b[10];

    int n;

    printf("Enter No Of Employee : ");

    scanf("%d", &n);

    for (int i = 0; i < n; i++)

    {

        printf("Enter the employee id = ");

        scanf("%d", &b[i].id);

        printf("Enter the employee Name = ");

        scanf(" %[^\n]%\*c", b[i].name);

        printf("Enter the employee Gender = ");

        scanf(" %c", &b[i].gender);

        printf("Enter the employee designation = ");

        scanf(" %[^\n]%\*c", b[i].desig);

        printf("Enter employee dept = ");

        scanf(" %[^\n]%\*c", b[i].dept);

        printf("Enter basic pay = ");

        scanf("%d", &b[i].basic);

        da = (75 \* b[i].basic) / 100.0;

        ha = (25 \* b[i].basic) / 100.0;

        b[i].gross = b[i].basic + ha + da ;

        printf("\n");

    }

    int i = 0;

    for (int i = 0; i < n; i++)

    {   printf("\nEmployee id =");

        printf("%d",  b[i].id);

        printf("\nEmployee name ");

        printf("%s",b[i].name);

        printf("\nEmployee Gender ");

        printf("%c",b[i].gender);

        printf("\nEmployee desig ");

         printf("%s",b[i].desig);

         printf("\nEmployee dept");

         printf("%s",b[i].dept);

         printf("\nEmployee gross");

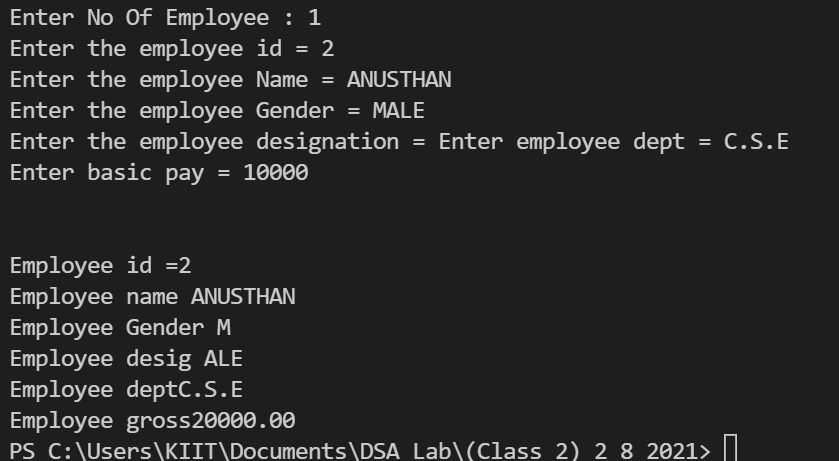
         printf("%.2f",b[i].gross);

    }

    return 0;

}

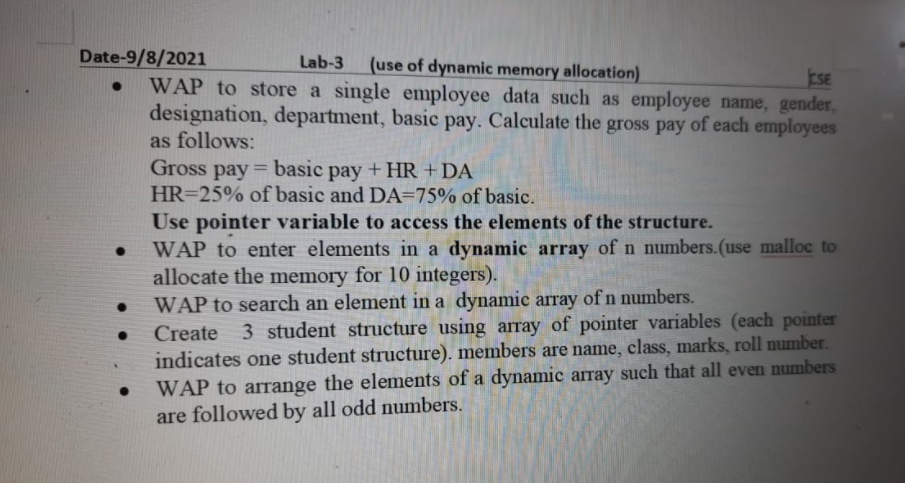
}



Output:-

Lab 3

Date – 9/8/2021

Q

Q1

//By Anusthan Singh (20051337)

#include<stdio.h>

#include<stdlib.h>

int n;

typedef struct

{

char dep[50],name[50],des[50],gen[6];

float basic;

}a;

int main()

{

    int i;

    float gp,hr,da;

 printf("Enter the number of empolyee =");

scanf("%d",&n);

 a \* x= (a\*) malloc(n\*sizeof(a));

    for(i=0;i<n;i++)

    {

        printf("Enter the employee Name = ");

        scanf("%s",x[i].name);

        printf("Enter the employee Gender =  ");

        scanf("%s",x[i].gen);

        printf("Enter the employee Department = ");

        scanf("%s",&x[i].dep);

        printf("Enter the employee Designation = ");

        scanf("%s",&x[i].des);

        printf("Enter the employee's Basic Pay = ");

        scanf("%f",&x[i].basic);

            }

        printf("Name         Gender       Department    Designation   Basic Pay    Gross Pay \n");

    for(i=0;i<n;i++)

    {

        hr= (25.0/100.0)\*x[i].basic;

        da=(75.0/100.0)\*x[i].basic;

        gp= hr+da+x[i].basic;

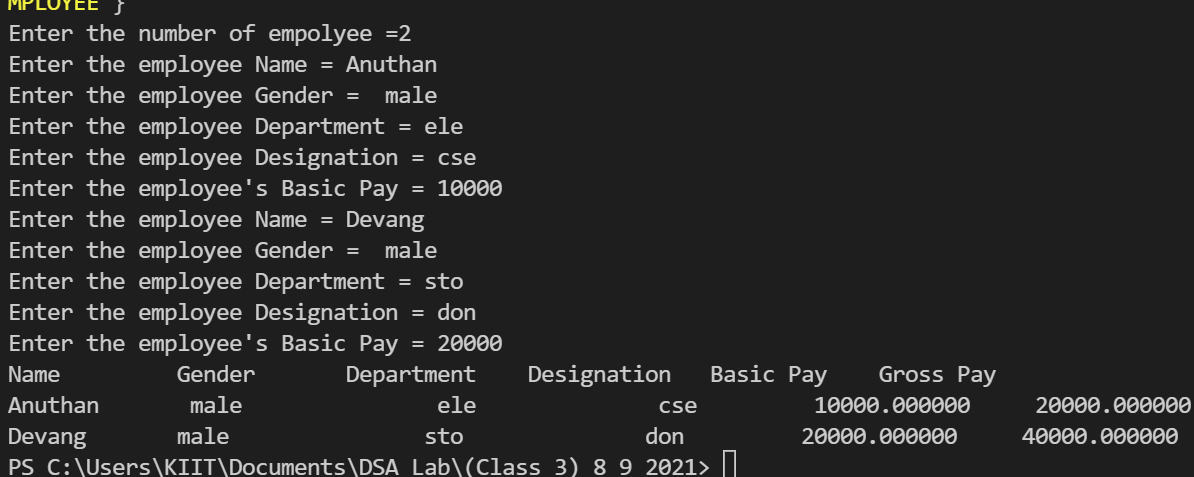
        printf("%s       %s               %s              %s         %f     %f\n",x[i].name,x[i].gen,x[i].dep,x[i].des,x[i].basic,gp);

    }

   return 0;

}

Output



Q2

//By Anusthan Singh (20051337)

#include<stdio.h>

#include<stdlib.h>

int main()

{

    int \*ptr,a,i;

    printf("\nEnter the size of array = ");

    scanf("%d",&a);

    ptr = (int\*)malloc(10 \* sizeof(int));

    printf("\n Enter its %d elements = ",a);

    for (i=0;i<a;i++)

    scanf("%d",(ptr+i));

    printf("After the allocation your array becomes =");

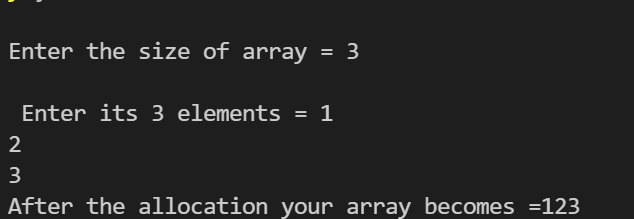
    for (i=0;i<a;i++)

    printf("%d",\*(ptr+i));

      return 0;

}

Output



Q3.

#include<stdio.h>//By Anusthan Singh (20051337)

int main()

{

    int arr[10], n, i, find, Flag;

    printf("\nEnter the size of array =  ");

    scanf("%d",&n);

    printf("\nEnter %d elements of the array= \n", n);

    for(i = 0; i < n; i++)

    {

        scanf("%d",&arr[i]);

    }

    printf("\nEnter the Element to Search  =  ");

    scanf("%d",&find);

    Flag = 0;

    for(i = 0; i < n; i++)

    {

        if(arr[i] == find)

        {

            Flag = 1;

            break;

        }

    }

    if(Flag == 1)

    {

        printf("\nfounded the Search Element was %d at at the Position %d ", find, i + 1);

    }

    else

    {

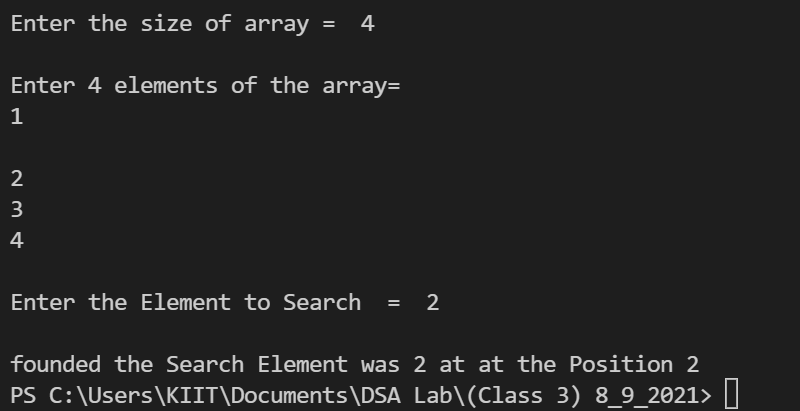
        printf("\n Not found the the Search Element %d ", find);

    }

    return 0;

}

Output



Q4.

#include <stdlib.h>

#include <stdio.h>

struct student

{

 int roll, class, mark;

 char name[50];

};

int main(void)

{

 struct student s[3], \*ptr;

 for (int i = 0; i < 3; i++)

 {

printf("\n\nEnter the roll no of the student = ");

 scanf("%d", &s[i].roll);

 printf("Enter the name  of the student = ");

 scanf("%s", &s[i].name);

 printf("Enter the class  of the student = ");

 scanf("%d", &s[i].class);

 printf("Enter the marks of the student = ");

 scanf("%d", &s[i].mark);

 }

 ptr = s;

 for (int i = 0; i < 3; i++)

 {

 printf("Name : %s\n", ptr->name);

 printf("Roll : %d\n", ptr->roll);

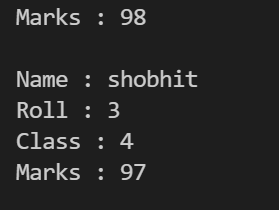
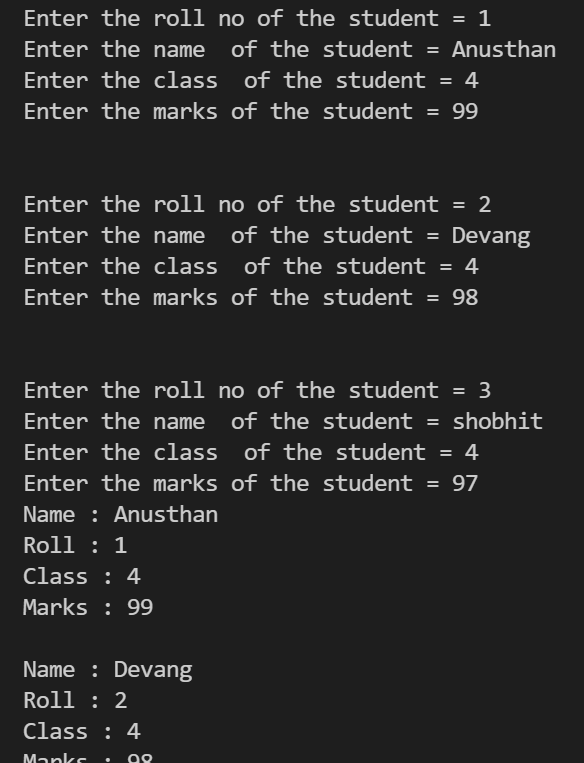
 printf("Class : %d\n", ptr->class);

 printf("Marks : %d\n\n", ptr->mark);

 ptr++;

 } }

Output:-



Q5.

#include <stdio.h>//By Anusthan Singh (20051337)

int main()

{

    int n;

    printf("Enter the size of array = ");

    scanf("%d", &n);

    int x[n], y[n], tempo = 0;

    printf("Enter the %d Elements: \n",n);

    for (int i = 0; i < n; i++)

    {

        scanf("%d", &x[i]);

    }

    for (int i = 0; i < n; i++)

    {

        if (x[i] % 2 == 0)

        {

            y[tempo] = x[i];

            tempo++;

        }

    }

    for (int i = 0; i < n; i++)

    {

        if (x[i] % 2 != 0)

        {

            y[tempo] = x[i];

            tempo++;

        }

    }

    printf("Your new array as even odd allocation = \n");

    for (int i = 0; i < n; i++)

    {

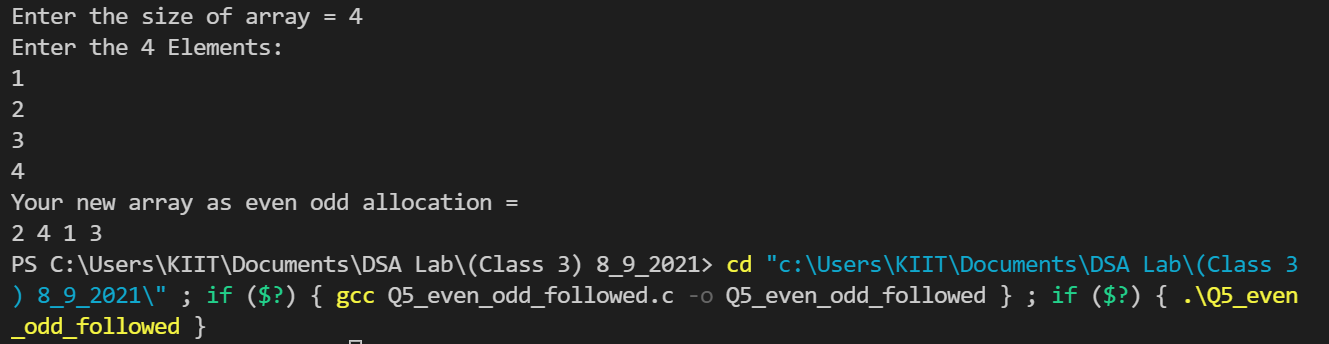
        printf("%d ", y[i]);

    }

    return 0;

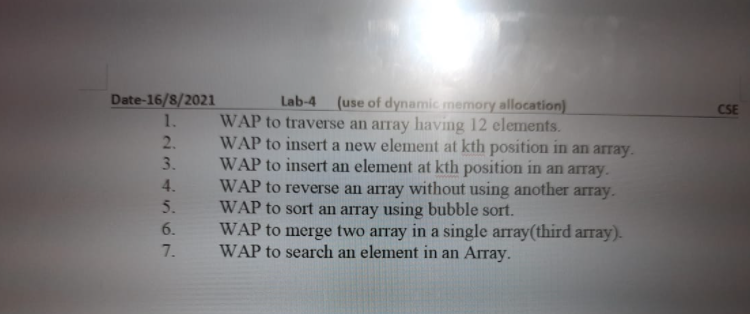
}

Output:-



Lab 4

Date - 16/8/2021

Q

Q1

#include<stdio.h>//By Anusthan Singh (20051337)

#include<conio.h>

int main()

{

    int num1;

    printf("Enter the size of the array = ");

    scanf("%d",&num1);

    int arr[num1],i;

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

    for(i=0;i<num1;i++)

    scanf("%d",&arr[i]);

    printf(" Array = ");

    for(i=0;i<num1;i++)

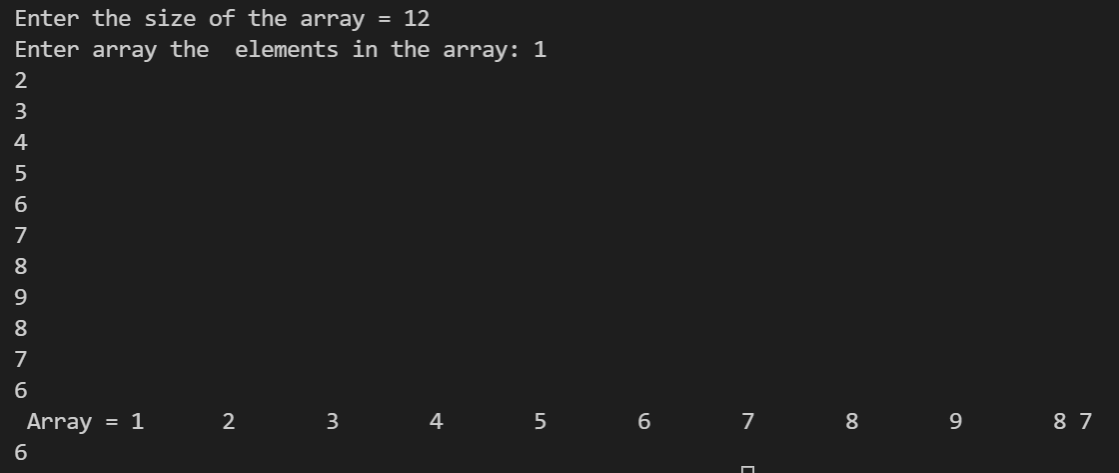
    printf("%d\t",arr[i]);

    return 0;

    getch();

}

Output:-



Q2.

#include <stdio.h>//By Anusthan Singh (20051337)

#define MAX\_SIZE 100

int main()

{

    int arr[MAX\_SIZE];

    int i, size, element, position;

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

    scanf("%d", &size);

    printf("Enter elements in array : ");

    for(i=0; i<size; i++)

    {

        scanf("%d", &arr[i]);

    }

    printf("Enter element which you want to insert : ");

    scanf("%d", &element);

    printf("Enter that  position : ");

    scanf("%d", &position);

    if(position > size+1 || position <= 0)

    {

        printf(" Not possible! Please enter position between 1 to %d", size);

    }

    else

    {

        for(i=size; i>=position; i--) // // New room by right shift kar ke

        {

            arr[i] = arr[i-1];

        }

        arr[position-1] = element;   //Insertation

        size++;

        printf(" The New Array after insertion : ");

        for(i=0; i<size; i++)

        {

            printf("%d  ", arr[i]);

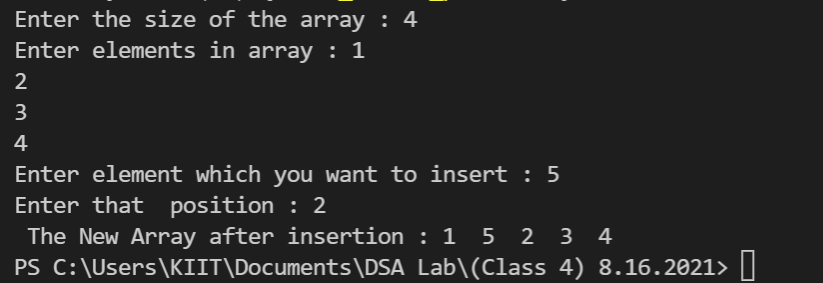
        }

    }

    return 0;

}

Output:-



Q3.

#include <stdio.h>//By Anusthan Singh (20051337)

int main()

{

    int arr[100] = { 0 };

    int i, x, insert, n = 10;

    for (i = 0; i < 10; i++)

        arr[i] = i + 1;

    for (i = 0; i < n; i++)

        printf("%d ", arr[i]);

    printf("\n");

    x = 50;

    insert = 5;

    n++;

    for (i = n-1; i >= insert; i--)

        arr[i] = arr[i - 1];

    arr[insert - 1] = x;

    for (i = 0; i < n; i++)

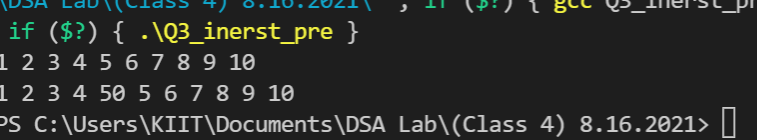
        printf("%d ", arr[i]);

    printf("\n");

    return 0;

}

Output.



Q4.

#include <stdio.h>//By Anusthan Singh (20051337)

int main()

{

    int n, i, d, a[100], b[100];

    printf("Enter the size of the array : \n");

    scanf("%d", &n);

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

    for (i = 0; i < n; i++){

        scanf("%d", &a[i]);

    }

    for (i = n - 1, d = 0; i >= 0; i--, d++)

        b[d] = a[i];

    for (i = 0; i < n; i++)

        a[i] = b[i];

    printf("Reverse array is : \n");

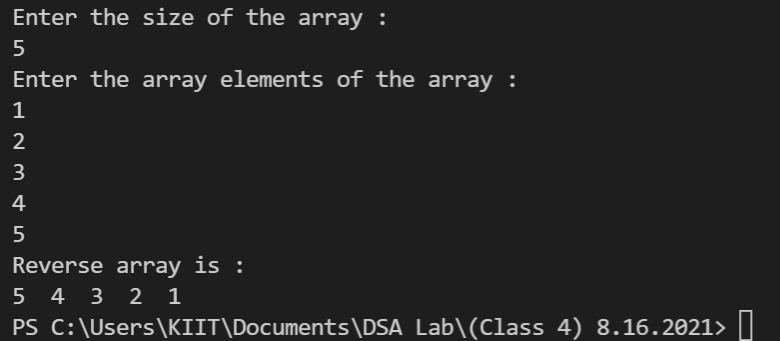
    for (i = 0; i < n; i++)

        printf("%d  ", a[i]);

    return 0;

}

Output.



Q5.

#include<stdio.h>//By Anusthan Singh (20051337)

#include<conio.h>

int main()

{

    int arr[50], i, j, n, temp;

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

    scanf("%d", &n);

    printf("\n Enter %d elements = ", n);

    for(i=0; i<n; i++)

        scanf("%d", &arr[i]);

    for(i=0; i<(n-1); i++)

    {

        for(j=0; j<(n-i-1); j++)

        {

            if(arr[j]>arr[j+1])

            {

                temp = arr[j];

                arr[j] = arr[j+1];

                arr[j+1] = temp;

            }

        }

    }

    printf("Array elements are sorted in ascending order = \n");

    for(i=0; i<n; i++)

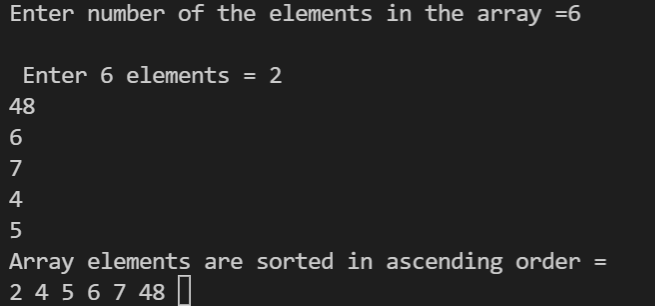
        printf("%d ", arr[i]);

    getch();

    return 0;

}

Output.



Q6.

a#include<stdio.h>//By Anusthan Singh (20051337)

#include<conio.h>

int main()

{

      int n,m;

    printf("Enter the size of the  1st array  = ");

    scanf("%d",&n);

    printf("\nEnter the size of the  2nd array = ");

    scanf("%d",&m);

    int arr1[n],arr2[m],arr3[m+n],i,j;

    printf("\nEnter elements for the 1st array  = ");

    for(i=0;i<n;i++)

    scanf("%d",&arr1[i]);

    printf("\nEnter elements for the 2nd array = ");

    for(i=0;i<m;i++)

    scanf("%d",&arr2[i]);

    printf("\n Array after merged = ");

    for(i=0,j=0;i<(m+n);i++){

        if(i<n)

        arr3[i] = arr1[i];

        else{

        arr3[i] = arr2[j];

        j++;

        }

        printf("%d ",arr3[i]);

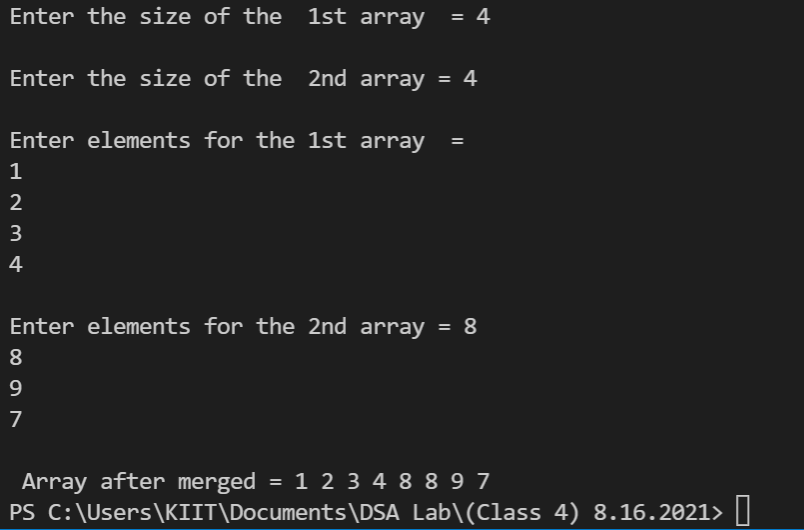
    }

    return 0;

    getch();

}

Output.



Q7.

#include<stdio.h>//By Anusthan Singh (20051337)

#include<conio.h>

int main()

{

     int n,count=0;

    printf("Enter the size of array = ");

    scanf("%d",&n);

    int arr[n],i,element;

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

    for(i=0;i<n;i++)

    scanf("%d",&arr[i]);

    printf("Enter the elemnts to search in array : ");

    scanf("%d",&element);

    for(i=0;i<n;i++){

        if(arr[i]==element){

            count++;

            printf("Element is founded at %d position ",i+1);

            break;

        }

    }

    if(count==0)

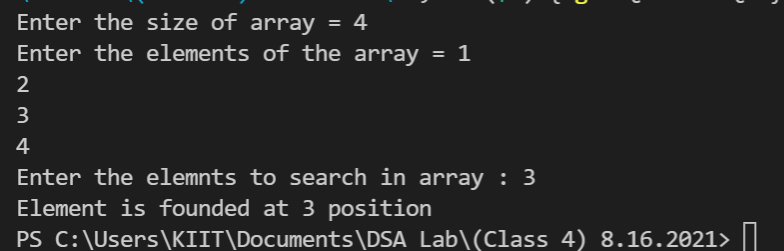
    printf("Element is not found ");

    return 0;

    getch();

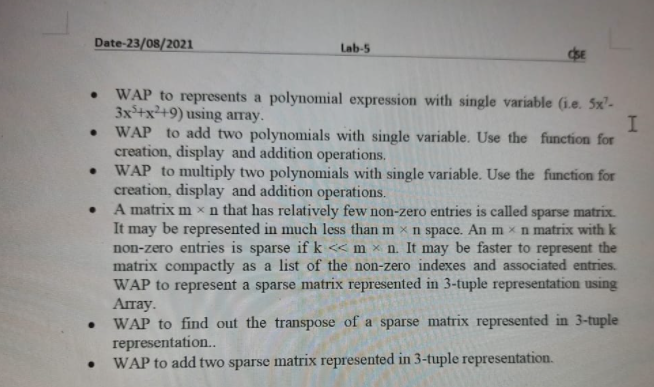
}

Output.



Lab – 5

Date – 23/8/2021

Q

Q1.

#include<stdio.h>//By Anusthan Singh (20051337)

#include<conio.h>

int main()

{

    int x;

    printf("Enter no. of terms in the expression : ");

    scanf("%d",&x);

    int coe[x],exp[x];

    for(int i=0;i<x;i++){

        printf("Enter the coefficient of %d term : ",i+1);

        scanf("%d",&coe[i]);

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

        scanf("%d",&exp[i]);

    }

    printf("The polynomial would be : \n");

    for(int i=0;i<x-1;i++){

        printf("%dx^%d+",coe[i],exp[i]);

    }

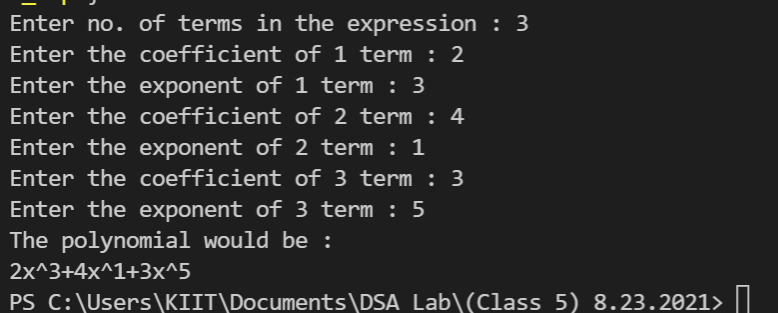
    printf("%dx^%d",coe[x-1],exp[x-1]);

    return 0;

    getch();

}

Output .



Q2.

#include<stdio.h>//By Anusthan Singh (20051337)

#include<math.h>

struct poly

{

  float coeff;

  int exp;

};

struct poly a[20],b[20],c[20],d[20];

int main()

{

 int i;

 int deg1,deg2;

 int k=0,l=0,m=0;

 printf("Enter the highest degree of the poly1:");

 scanf("%d",&deg1);

 for(i=0;i<=deg1;i++)

 {

    printf("\nEnter the coeff of x^%d :",i);

    scanf("%f",&a[i].coeff);

  a[k++].exp = i;

 }

 printf("\nEnter the highest degree of poly2:");

 scanf("%d",&deg2);

 for(i=0;i<=deg2;i++)

 {

       printf("\nEnter the coeff of x^%d :",i);

       scanf("%f",&b[i].coeff);

     b[l++].exp = i;

 }

  printf("\nExpression 1 = %.1f",a[0].coeff);

  for(i=1;i<=deg1;i++)

  {

    printf("+ %.1fx^%d",a[i].coeff,a[i].exp);

  }

  printf("\nExpression 2 = %.1f",b[0].coeff);

   for(i=1;i<=deg2;i++)

    {

      printf("+ %.1fx^%d",b[i].coeff,b[i].exp);

    }

 if(deg1>deg2)

    {

     for(i=0;i<=deg2;i++)

      {

      c[m].coeff = a[i].coeff + b[i].coeff;

      c[m].exp = a[i].exp;

      m++;

      }

      for(i=deg2+1;i<=deg1;i++)

      {

      c[m].coeff = a[i].coeff;

      c[m].exp = a[i].exp;

      m++;

      }

    }

 else

  {

    for(i=0;i<=deg1;i++)

     {

       c[m].coeff = a[i].coeff + b[i].coeff;

       c[m].exp = a[i].exp;

       m++;

     }

  for(i=deg1+1;i<=deg2;i++)

    {

      c[m].coeff = b[i].coeff;

      c[m].exp = b[i].exp;

      m++;

    }

  }

  printf("\nExpression after additon  = %.1f",c[0].coeff);

  for(i=1;i<m;i++)

  {

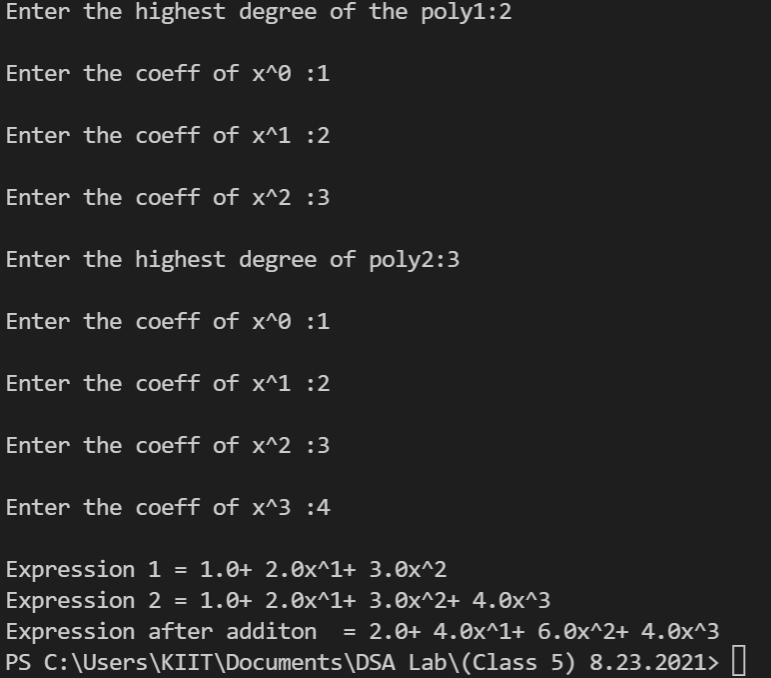
     printf("+ %.1fx^%d",c[i].coeff,c[i].exp);

   }

  return 0;

}

Output.



Q3.

#include<stdio.h>//By Anusthan Singh (20051337)

#include<conio.h>

struct poly

{

  int degree;

  int coeff;

};

void main ()

{

  struct poly poly1[10], poly2[10], multi[100];

  int n1, n2, count = -1;

  int i, j;

  printf ("\nEnter Size Of the  1st Polynomial: ");

  scanf ("%d", &n1);

  for (i = 0; i < n1; i++)

    {

      printf ("\nEnter its Degree: ");

      scanf ("%d", &poly1[i].degree);

      printf ("\nEnter its Coefficient: ");

      scanf ("%d", &poly1[i].coeff);

    }

  printf ("\nEnter Size of the Of 2nd Polynomial: ");

  scanf ("%d", &n2);

  for (i = 0; i < n2; i++)

    {

      printf ("\nEnter its Degree: ");

      scanf ("%d", &poly2[i].degree);

      printf ("\nEnter its Coefficient: ");

      scanf ("%d", &poly2[i].coeff);

    }

  for (i = 0; i < n1; i++)

    {

      for (j = 0; j < n2; j++)

  {

    multi[++count].degree = poly1[i].degree + poly2[j].degree;

    multi[count].coeff = poly1[i].coeff \* poly2[j].coeff;

  }

    }

  printf ("\nThe Multiplication Of the Two Polynomialsi is = \n");

  for (i = 0; i <= count; i++)

    {

      if (multi[i].degree == 0)

  printf ("%d ", multi[i].coeff);

      else if (multi[i].degree == 1)

  printf ("%dx ", multi[i].coeff);

      else

  {

    printf ("%dx^%d ", multi[i].coeff, multi[i].degree);

  }

      if (i != count)

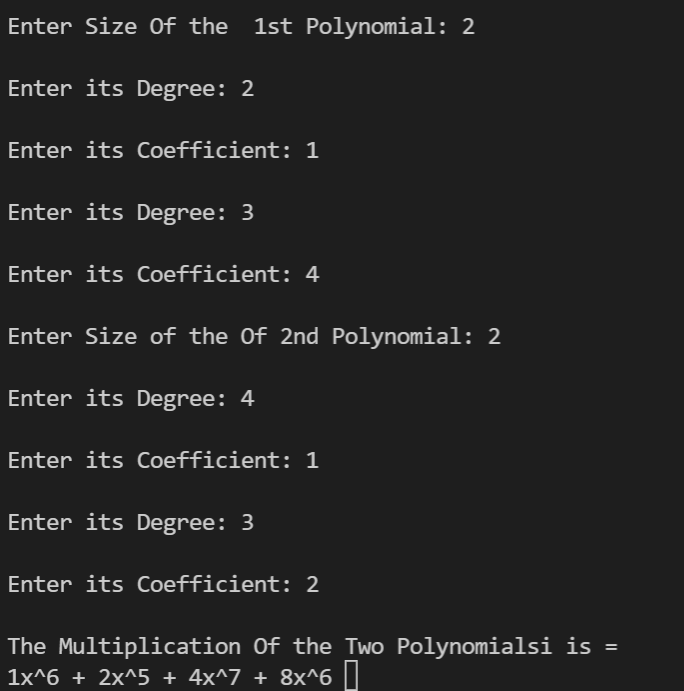
  printf ("+ ");

    }

  getch ();

}

Output.



Q4.

#include<stdio.h>//By Anusthan Singh (20051337)

int main()

{

    int sparseMatrix[4][5] =

    {

        {0 , 0 , 3 , 0 , 4 },

        {0 , 0 , 5 , 7 , 0 },

        {0 , 0 , 0 , 0 , 0 },

        {0 , 2 , 6 , 0 , 0 }

    };

    int size = 0;

    for (int i = 0; i < 4; i++)

        for (int j = 0; j < 5; j++)

            if (sparseMatrix[i][j] != 0)

                size++;

    int compactMatrix[3][size];

    int k = 0;                       // yeha se naya matrix banega

    for (int i = 0; i < 4; i++)

        for (int j = 0; j < 5; j++)

            if (sparseMatrix[i][j] != 0)

            {

                compactMatrix[0][k] = i;

                compactMatrix[1][k] = j;

                compactMatrix[2][k] = sparseMatrix[i][j];

                k++;

            }

    for (int i=0; i<3; i++)

    {

        for (int j=0; j<size; j++)

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

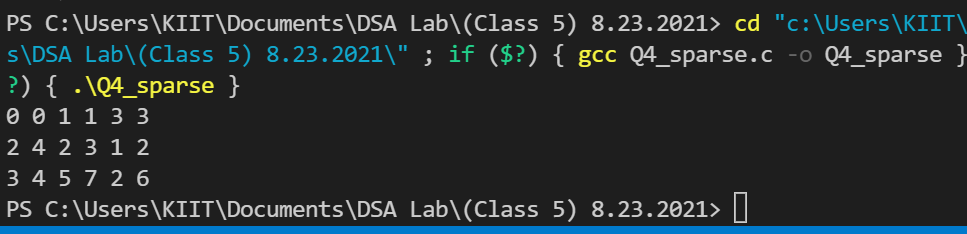
        printf("\n"); // printing machine

    }

    return 0;

}

Output.



Q5

#include<stdio.h>//By Anusthan Singh (20051337)

#include<conio.h>

#include<string.h>

#include<math.h>

int k=1,i,j,m,num,n,p[10][10],q[10][10],ct=0,R=1,C,r1,r2,c1,c2;

void read(int a[10][10],int r,int c)

{

    k=1;

    R=1;

    ct=0;

    printf("Enter the matrix:");

    for(i=0;i<r;i++)

    {

        for(j=0;j<c;j++)

        {

            scanf("%d",&num);

            if(num!=0)

            {

                a[R][C]=i;

                a[R][++C]=j;

                a[R][++C]=num;

                ct++;

                R++;

                C=0;

            }

        }

    }

    a[0][0]=r;

    a[0][1]=c;

    a[0][2]=ct;

}

void trans()

{

    printf("Enter row & coloumn of matrix to transpose: ");

    scanf("%d %d",&r1,&c1);

    read(p,r1,c1);

    printf("IT'S TRANSPOSE : ");

    k=1;

    for(i=0;i<r1;i++)

    {

        for(j=0;j<c1;j++)

        {

            if(p[k][0]==i&&p[k][1]==j)

            {

                q[j][i]=p[k][2];

                k++;

            }

            else

                q[j][i]=0;

        }

    }

    for(i=0;i<c1;i++)

    {

        for(j=0;j<r1;j++)

        {

            printf(" %d ",q[i][j]);

        }

        printf("");

    }

}

void main()

{

    char ch;

    fflush(stdin);

    do

    {

        ct=0;

        fflush(stdin);

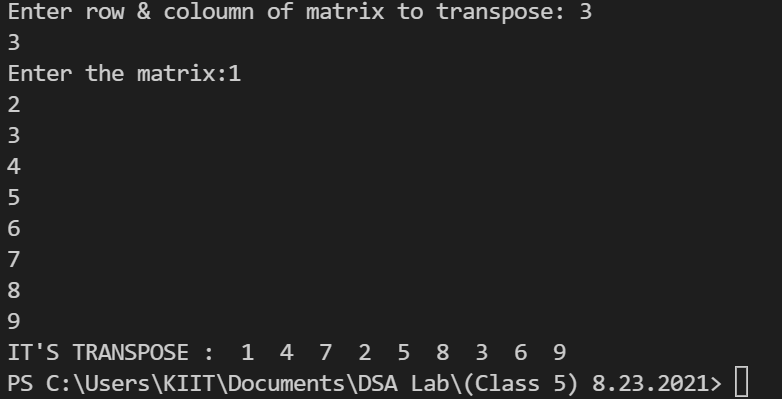
        fflush(stdout);

         trans();

    }while(ch=='y');

}

Output.



Q6.

#include<stdio.h>//By Anusthan Singh (20051337)

#include<conio.h>

#include<string.h>

#include<math.h>

int k=1,i,j,m,num,n,p[10][10],q[10][10],ct=0,R=1,C,r1,r2,c1,c2;

void read(int a[10][10],int r,int c)

{

    k=1;

    R=1;

    ct=0;

    printf("Enter the matrix:");

    for(i=0;i<r;i++)

    {

        for(j=0;j<c;j++)

        {

            scanf("%d",&num);

            if(num!=0)

            {

                a[R][C]=i;

                a[R][++C]=j;

                a[R][++C]=num;

                ct++;

                R++;

                C=0;

            }

        }

    }

    a[0][0]=r;

    a[0][1]=c;

    a[0][2]=ct;

}

void add()

{

    printf("Enter the row & coloumn of 1st matrix: ");

    scanf("%d %d",&r1,&c1);

    printf("Enter the row & coloumn of 2nd matrix: ");

    scanf("%d %d",&r2,&c2);

    if(r1==r2&&c1==c2)

    {

        read(p,r1,c1);

        read(q,r2,c2);

        k=1;

        printf("Sum Matrix:");

        for(i=0;i<r1;i++)

        {

            for(j=0;j<c1;j++)

            {

                if((p[k][0]==i&&p[k][1]==j)&&(q[k][0]==i&&q[k][1]==j))

                {

                    printf(" %d ",p[k][2]+q[k][2]);

                    k++;

                }

                else if(p[k][0]==i&&p[k][1]==j)

                {

                    printf(" %d ",p[k][2]);

                    k++;

                }

                else if(q[k][0]==i&&q[k][1]==j)

                {

                    printf(" %d ",q[k][2]);

                    k++;

                }

                else

                    printf(" 0 ");

        }

        printf("");

    }

 }

 else

    printf("Addition not possible!");

}

void main()

{

    char ch;

    fflush(stdin);

    do

    {

        ct=0;

        fflush(stdin);

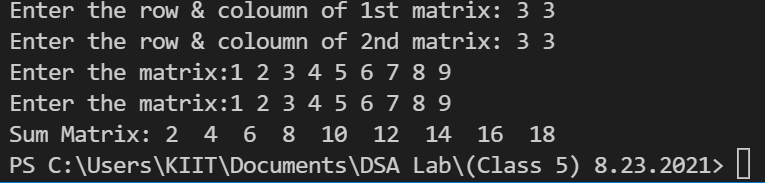
        fflush(stdout);

            add();

    }while(ch=='y');

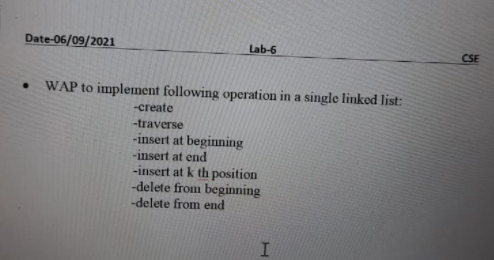
}

Output .



Lab-6

Date – 6/9/2021

Q

Q1.

#include <stdio.h>//By Anusthan Singh (20051337)

#include <stdlib.h>

struct node {

    int data;//data

    struct node \*next; // Address

}\*head;

void creatingList(int n);

void insertNodeAtBeginning(int NUMBER);

void insertNodeAtheEnd(int NUMBER);

void insertNodeAtheMiddle(int NUMBER, int pos);

void deletethefirstnode();

void deletethelastnode();

void displaythatList();

int main()

{

    int n, data, position, choice;

      //Create a singly linked list of n nodes

    printf("Enter the total number of nodes in the list = ");

    scanf("%d", &n);

    creatingList(n);

    printf("\n Now Data in the list is \n");

    displaythatList();

     // Insert data at the beginning of the singly linked list

    printf("\nEnter that data which you wantto insert at beginning of the list: ");

    scanf("%d", &data);

    insertNodeAtBeginning(data);

    printf("\nNow Data in the list is \n");

    displaythatList();

     // Insert data at the end of the singly linked list

    printf("\nEnter that data which you want to insert at end of the list: ");

    scanf("%d", &data);

    insertNodeAtheEnd(data);

    printf("\nNow Data in the list is \n");

    displaythatList();

     // Insert data at middle of the singly linked list

    printf("nEnter data to insert at the middle of the list: ");

    scanf("%d", &data);

    printf("Enter the position for that insert new node: " );

    scanf("%d", &position);

    insertNodeAtheMiddle(data, position);

    printf("\nNow Data in the list is \n");

    displaythatList();

    printf("\nPress 1 to delete the first node: ");

    scanf("%d", &choice);

    /\* Delete first node from list \*/

    if(choice == 1)

        deleteFirstNode();

    printf("\nNow Data in the list is \n");

    displaythatList();

    printf("\nPress 2 to delete last node: ");

    scanf("%d", &choice);

    /\* Delete last node from list \*/

    if(choice == 2)

        deleteLastNode();

    printf("\nNow Data in the list is \n");

    displaythatList();

    return 0;

}

void creatingList(int n)//n node

{

    struct node \*newNode, \*temp;

    int data, i;

    head = (struct node \*)malloc(sizeof(struct node));

    if(head == NULL)

    {

        printf("Unable to allocate memory.");

    }

    else

    {

        printf("Enter the data of node 1: ");

        scanf("%d", &data);

        head->data = data; // Link data field to data

        head->next = NULL; // Link address field to NULL

        temp = head;

        /\*

         \* Create n nodes and adds to linked list

         \*/

        for(i=2; i<=n; i++)

        {

            newNode = (struct node \*)malloc(sizeof(struct node));

            if(newNode == NULL)

            {

                printf("Unable to allocate memory.");

                break;

            }

            else

            {

                printf("Enter the data of node %d: ", i);

                scanf("%d", &data);

                newNode->data = data; // Link data field of newNode with data

                newNode->next = NULL; // Link address field of newNode with NULL

                temp->next = newNode; // Link previous node i.e. temp to the newNode

                temp = temp->next;

            }

        }

        printf("SINGLY LINKED LIST CREATED SUCCESSFULLY\n");

    }

}

//1 insert new node at beginning(create new node)

void insertNodeAtBeginning(int data)

{

    struct node \*newNode;

    newNode = (struct node\*)malloc(sizeof(struct node));

    if(newNode == NULL)

    {

        printf("Unable to allocate memory.");

    }

    else

    {

        newNode->data = data; // Link data part

        newNode->next = head; // Link address part

        head = newNode;

        printf("DATA INSERTED SUCCESSFULLY\n");

    }

}

 // inserts at the end of the linked list.(create new node)

void insertNodeAtheEnd(int data)

{

    struct node \*newNode, \*temp;

    newNode = (struct node\*)malloc(sizeof(struct node));

    if(newNode == NULL)

    {

        printf("Unable to allocate memory.");

    }

    else

    {

        newNode->data = data; // Link the data part

        newNode->next = NULL;

        temp = head;

        // Traverse to  last (node)

        while(temp != NULL && temp->next != NULL)

            temp = temp->next;

        temp->next = newNode; // Link address part

        printf("DATA INSERTED SUCCESSFULLY\n");

    }

}

 //inserts at middle of the linked list.

void insertNodeAtheMiddle(int data, int position)

{

    int i;

    struct node \*newNode, \*temp;

    newNode = (struct node\*)malloc(sizeof(struct node));

    if(newNode == NULL)

    {

        printf("Unable to allocate memory.");

    }

    else

    {

        newNode->data = data; // Link data part

        newNode->next = NULL;

        temp = head;

        for(i=2; i<=position-1; i++) // Traverse to the n-1 position

        {

            temp = temp->next;

            if(temp == NULL)

                break;

        }

        if(temp != NULL)

        {

            /\* Link address part of new node \*/

            newNode->next = temp->next;

            /\* Link address part of n-1 node \*/

            temp->next = newNode;

            printf("DATA INSERTED SUCCESSFULLY\n");

        }

        else

        {

            printf("UNABLE TO INSERT DATA AT THE GIVEN POSITION\n");

        }

    }

}

//Deletes the first node

void deleteFirstNode()

{

    struct node \*toDelete;

    if(head == NULL)

    {

        printf("List is already empty.");

    }

    else

    {

        toDelete = head;

        head = head->next;

        printf("\nData deleted = %d\n", toDelete->data);

                             // Clears the memory OF first node

        free(toDelete);

        printf("DELETETION DONE FOR FIRST NODE  FROM LIST\n");

    }

}

 // Delete last node of the linked list

void deleteLastNode()

{

    struct node \*toDelete, \*secondLastNode;

    if(head == NULL)

    {

        printf("List is already empty.");

    }

    else

    {

        toDelete = head;

        secondLastNode = head;

        // Traverse to the last node

        while(toDelete->next != NULL)

        {

            secondLastNode = toDelete;

            toDelete = toDelete->next;

        }

        if(toDelete == head)

        {

            head = NULL;

        }

        else

        {

            secondLastNode->next = NULL;                                         /\* Disconnect link of second last node with last node \*/

        }

        free(toDelete);

        printf("DELETETION DONE FOR LAST NODE OF LIST \n");

    }

}

/\*

 \* Display entire list

 \*/

void displaythatList()

{

    struct node \*temp;

    /\*

     \* If the list is empty i.e. head = NULL

     \*/

    if(head == NULL)

    {

        printf("List is empty.");

    }

    else

    {

        temp = head;

        while(temp != NULL)

        {

            printf("Data = %d\n", temp->data); // Print data of current node

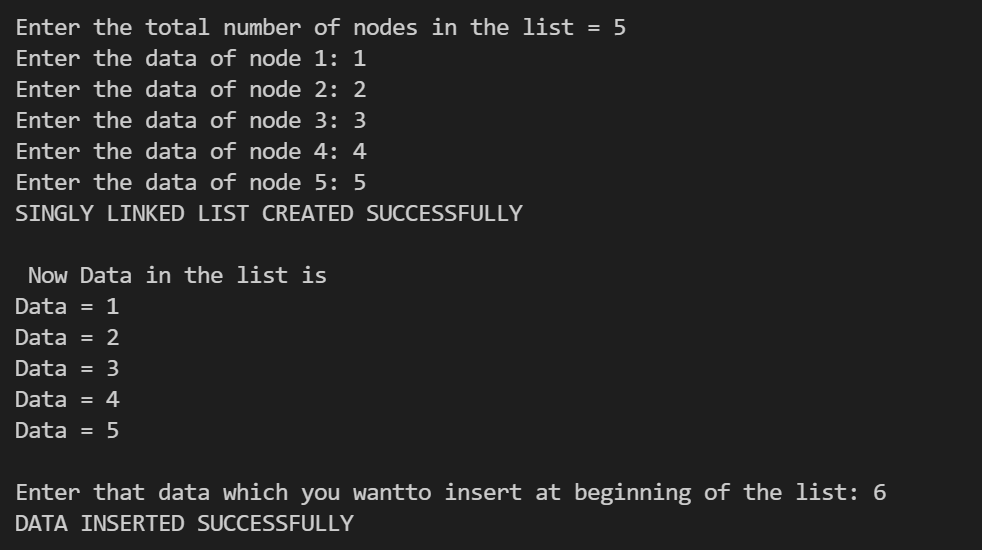
            temp = temp->next;                 // Move to next node

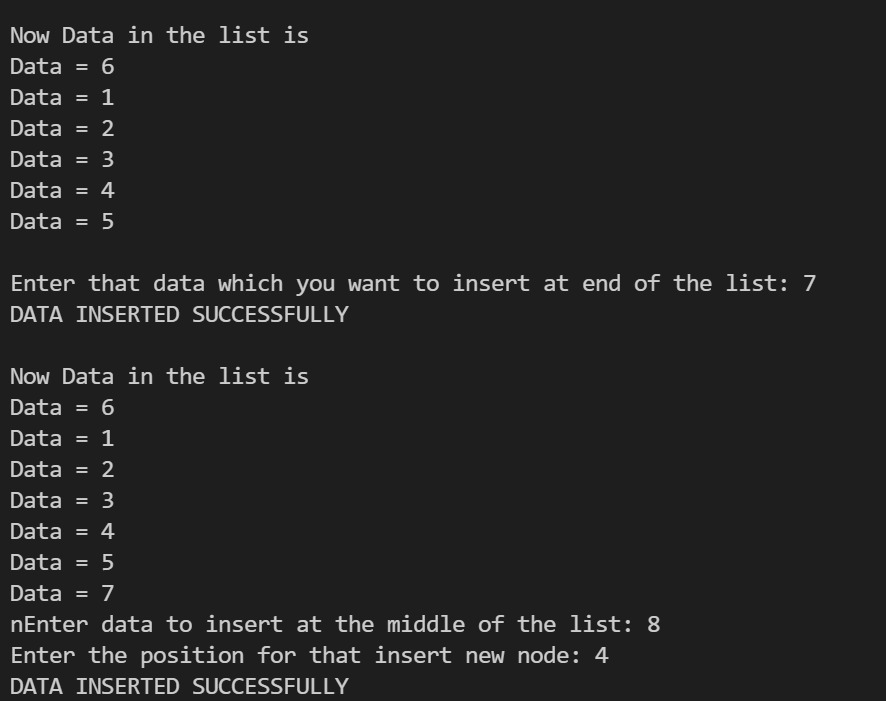
        }

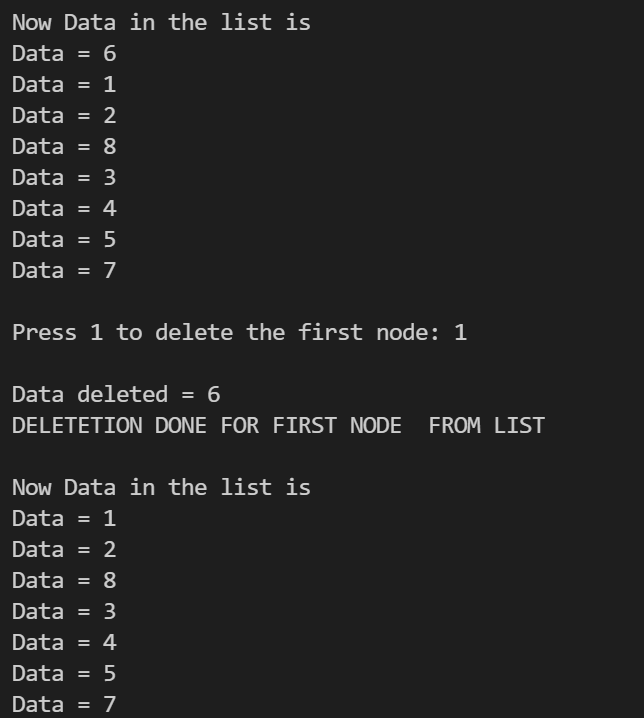
    }

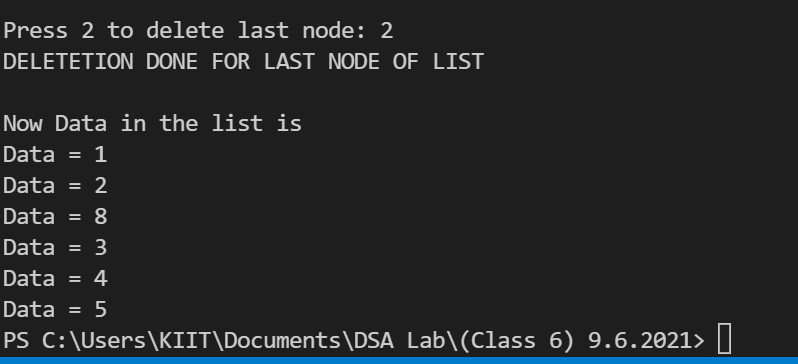
}

Output .









Q// Same question with switch case(simple)

#include <stdio.h>

#include <stdlib.h>

struct node{

int data;

struct node \*next;

}\*head=NULL;

void create(){

struct node \*ptr,\*last;

head=(struct node\*)malloc(sizeof(struct node));

printf("Enter 1st element: ");

scanf("%d",&head->data);

head->next=NULL;

last=head;

while(1){

 int temp;

 printf("Enter the next element(press 0 to end): ");

 scanf("%d",&temp);

 if(temp==0)

 break;

 else{

 ptr=(struct node\*)malloc(sizeof(struct node));

 ptr->data=temp;

 last->next=ptr;

 ptr->next=NULL;

 last=ptr;

 }

 }

printf("\nLINKED LIST IS CREATED SUCCESFULLY\n");

}

void displaythatList(){

struct node \*ptr;

ptr=head;

printf("\nLINKED LIST = \n");

while(ptr!=NULL){

 printf("%d\t",ptr->data);

 ptr=ptr->next;

}

}

void search(){

struct node\* ptr;

int count=0,temp;

printf("\nEnter element to search: ");

scanf("%d",&temp);

ptr=head;

while(ptr!=NULL && temp!=ptr->data){

 count++;

 ptr=ptr->next;

}

 printf("\nelement found at %d position\n",count+1);

}

void insert(){

struct node \*ptr,\*temp;

int s;

printf("\nEnter element to add after = ");

scanf("%d",&s);

ptr=head;

while(ptr!=NULL && s!=ptr->data){

 ptr=ptr->next;

}

 temp=(struct node\*)malloc(sizeof(struct node));

 printf("\nEnter element to insert: ");

 scanf("%d",&temp->data);

 temp->next=ptr->next;

 ptr->next=temp;

 printf("\nElement inserted\n");

}

void dlt(){

 struct node \*ptr,\*temp;

 int s;

 printf("\nEnter element to delete: ");

 scanf("%d",&s);

 ptr=head;

 while(s!=(ptr->next)->data){

 ptr=ptr->next;

 }

 ptr->next=ptr->next->next;

}

int main(void){

 int caseno;

 printf("\n0\tExit\n1)\tCreate\n2)\tDispaly\n3)\tInsert\n4)\tDelete\n");

 printf("Enter operation: ");

 scanf("%d",&caseno);

 switch (caseno){

 case 0:

 exit(1);

 case 1:

 create();

 main();

 case 2:

 displaythatList();

 main();

 case 3:

 insert();

 main();

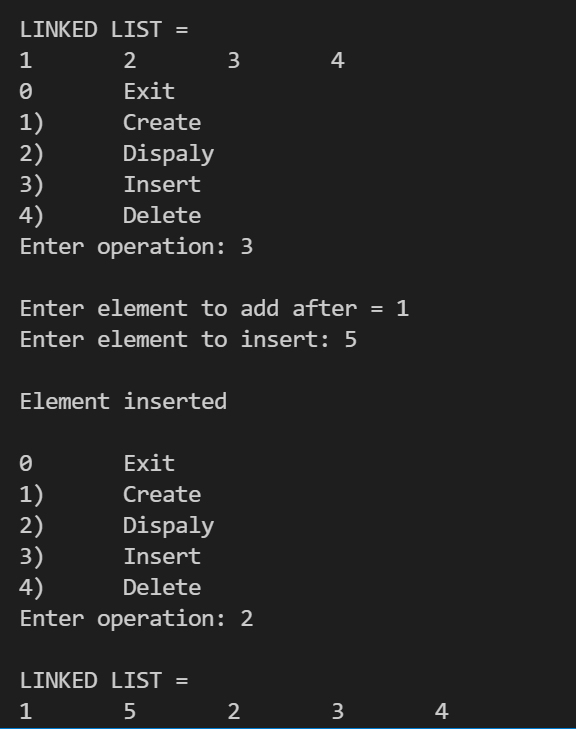
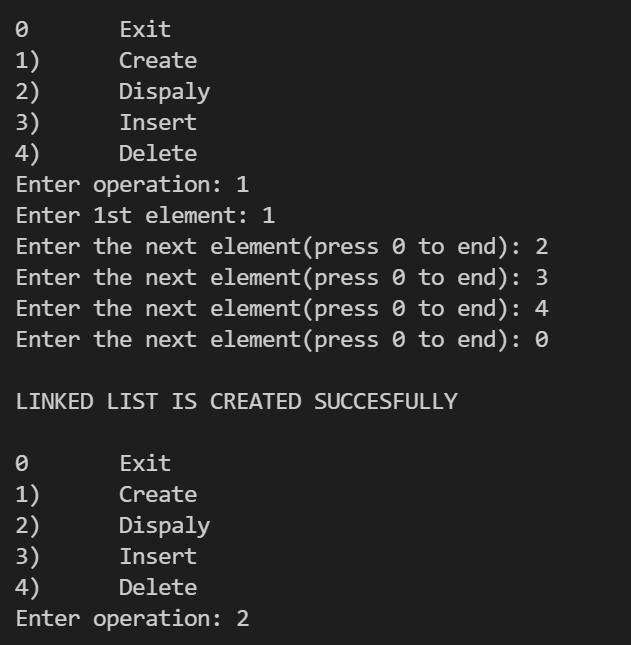
 case 4:

 dlt();

 main();

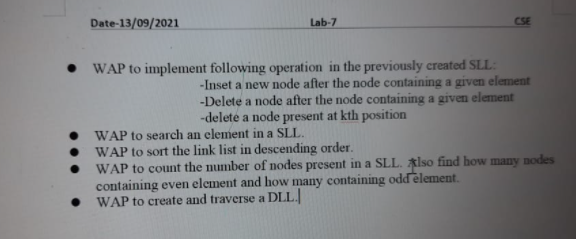
 } }

Output:-



Lab-7

Date – 13/8/2021

Q

Q1.// Without switch case

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

typedef struct Node

{

     int data;

     struct Node \*next;

}node;

node \*head;

node\* createnewlinkedlist(int);

void traverselinkedlist (node \*head);

void insertNode();

void deleteNode();

void DeletedNodeatK();

int main()

{

    int n;

    printf("Enter the no of nodes in the linked list = ");

    scanf("%d",&n);

    head = createnewlinkedlist(n);

    traverselinkedlist(head);

    insertNode();

    deleteNode();

    DeletedNodeatK();

    return 0;

    getch();

}

node\* createnewlinkedlist(int n){

    node \*head=NULL,\*temp,\*p;

    for(int i=0;i<n;i++){

        temp= (node\*)malloc(sizeof(node));

        printf("Enter data in %d node : ",i+1);

        scanf("%d",&(temp->data));

        temp->next=NULL;

        if(head==NULL)

        head = temp;

        else{

            p=head;

            while(p->next != NULL)

            p = p->next;

            p->next = temp;

        }

    }

    return head;

}

void traverselinkedlist(node \*head){

    node \*temp = head;

    if(temp == NULL)

    printf("Linked list is empty \n");

    else{

        printf("\nCreated linked list is = ");

        while(temp != NULL){

        printf("%d -> ",temp->data);

        temp = temp->next;

        }

    }

}

void insertNode(){

    int ele;

    printf("\nEnter element from linked list to insert new node after that : ");

    scanf("%d",&ele);

    node \*newNode=NULL,\*temp,\*p;

    newNode = (node\*)malloc(sizeof(node));

    printf("Enter data of new node : ");

    scanf("%d",&(newNode->data));

    newNode->next= NULL;

    temp = head;

    while(temp->next != NULL){

        if(temp->data == ele){

            newNode->next = temp->next;

            temp->next= newNode;

            break;

        }

        else

        temp=temp->next;

    }

    if(temp->next == NULL)

    {

         newNode->next = temp->next;

            temp->next= newNode;

    }

    traverselinkedlist(head);

}

void deleteNode(){

    int delete;

    printf("\nEnter element from linked list to delete node after that : ");

    scanf("%d",&delete);

    node \*p,\*q;

    p = head;

    while(p->next != NULL){

        if(p->data == delete){

           break;

        }

        else

        p = p->next;

    }

           q = p->next;

           p->next = q->next;

           q->next = NULL;

           free(q);

    traverselinkedlist(head);

}

void DeletedNodeatK(){

    int k;

    printf("\nEnter the kth position to delete the node : ");

    scanf("%d",&k);

   node \*p,\*q;

   p = head;

   while(k>2){

   p = p->next;

   --k;

   }

   q = p->next;

   p->next = q->next;

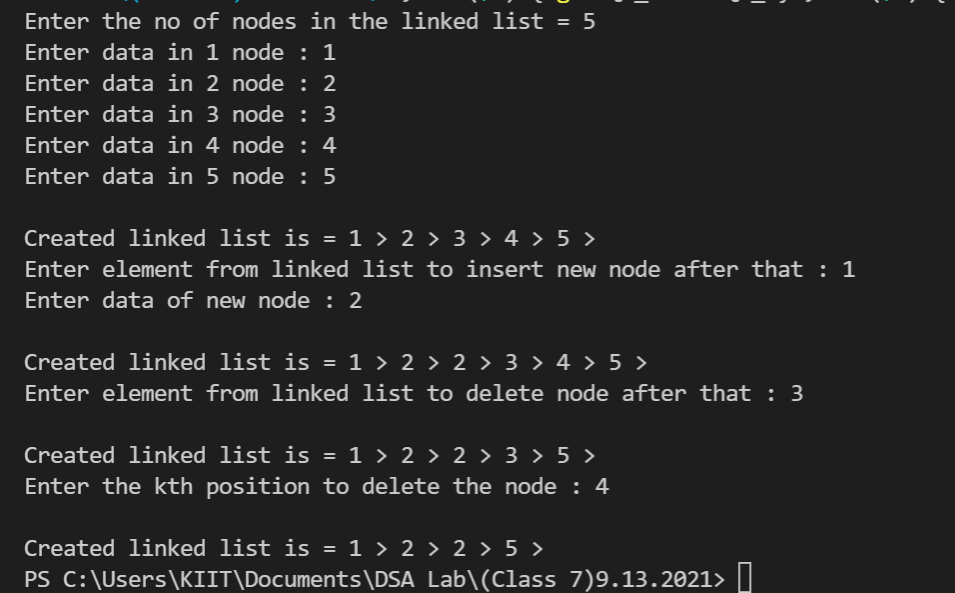
   q->next = NULL;

   free(q);

    traverselinkedlist(head);

}

Output:-



Q2.

#include<stdio.h>

#include<malloc.h>//By Anusthan Singh (20051337)

struct node

{

     int data;

     struct node \*next;

}

first, \*new;

int search(int item)

{

     int count=1;

     new=&first;

     while(new->next!=NULL)

     {

          if(new->data==item)

               break;

          else

               count++;

          new=new->next;

     }

     return count;

}

int main()

{

     int number,i,searching,position;

     first.next=NULL;

     new=&first;

     printf("Enter the no. of nodes  in linked list = ");

     scanf("%d",&number);

     printf("\n");

     for(i=0;i< number;i++)

     {

          new->next=(struct node \*)malloc(sizeof(struct node));

          printf("Enter element in the node %d: ",i+1);

          scanf("%d",&new->data);

          new=new->next;

     }

     new->next=NULL;

     printf("\nElements in linked list = \n\n");

     new=&first;

     while(new->next!=NULL)

     {

          printf("%d  ",new->data);

          new=new->next;

     }

     printf("  ");

     printf("\nEnter element to be searched : ");

     scanf("%d",&searching);

     position=search(searching);

     if(position<=number)

          printf("\n'%d' is found at node = %d",searching,position);

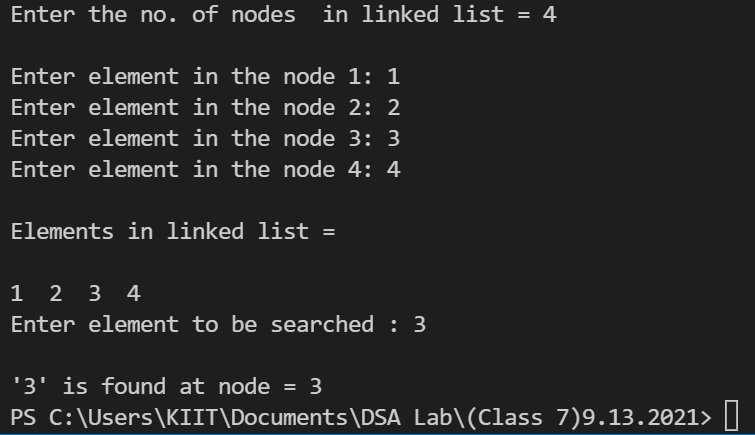
     else

          printf("The number '%d' is not in linked list.",searching);

     return 0;

}

Output .



Q3.

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

struct node

{

    int data;

    struct node \*next;

}\*head=NULL;

void arranging(struct node \*h)

{

    struct node \*current,\*ptr;

    int i,j,n;

    for(ptr=h;ptr!=NULL;ptr=ptr->next)

    {

        for(current=ptr->next;current!=NULL;current=current->next)

        {

            if(ptr->data<current->data)

            {

                n=ptr->data;

                ptr->data=current->data;

                current->data=n;

            }

        }

    }

}

void desending(struct node \*h)

{

    struct node \*current;

    current=h;

    printf("\nOrder of the elements : ");

    while(current!=NULL)

    {

        printf("%d ",current->data);

        current=current->next;

    }

}

void main()

{

    struct node \*current,\*ptr;

    int n,i,ch,d=1;

    printf("Enter the no. of nodes in the link list : ");

    scanf("%d",&n);

    for(i=0;i<n;i++)

    {

       current=malloc(sizeof(struct node));

       printf("Enter the elements of the  %d th postion : ",(i+1));

       scanf("%d",&current->data);

       current->next=NULL;

       if(head==NULL)

       {

          head=current;

          ptr=current;

       }

       else

       {

           ptr->next=current;

           ptr=current;

       }

    }

    desending(head);

    arranging(head);

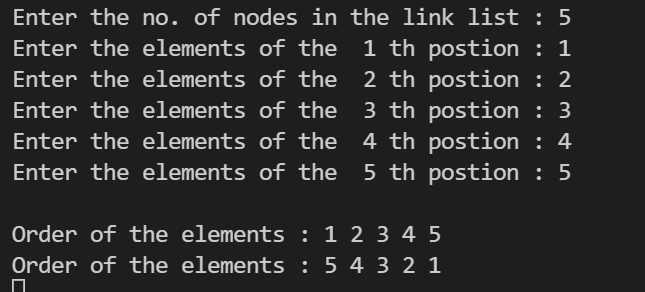
    desending(head);

    printf("\n");

   getch();

}

Output:-



Q4

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

struct node

{

    int data;

    struct node \*next;

}\*head=NULL;

void count(struct node \*h)

{

    struct node \*current,\*ptr;

    int count=0,elements=0;

    for(ptr=h;ptr!=NULL;ptr=ptr->next)

    {

        count++;

        if(ptr->data%2==0)

            elements++;

    }

    printf("\n");

    printf("\nThe No. Of Node  in the linklist =  %d ",count);

    printf("\nThe total even elements in the linklist = %d ",elements);

    printf("\nTHE total odds elements in the linklist = %d ",count-elements);

}

void traverse(struct node \*h)

{

    struct node \*current;

    current=h;

    printf("\n Your node is : ");

    while(current!=NULL)

    {

        printf("%d ",current->data);

        current=current->next;

    }

}

void main()

{

    struct node \*cur,\*ptr;

    int n,i,ch,d=1;

    printf("Enter the no of nodes in the linklist : ");

    scanf("%d",&n);

    for(i=0;i<n;i++)

    {

       cur=malloc(sizeof(struct node));

       printf("Enter the elements of node  %d : ",(i+1));

       scanf("%d",&cur->data);

       cur->next=NULL;

       if(head==NULL)

       {

          head=cur;

          ptr=cur;

       }

       else

       {

           ptr->next=cur;

           ptr=cur;

       }

    }

    traverse(head);

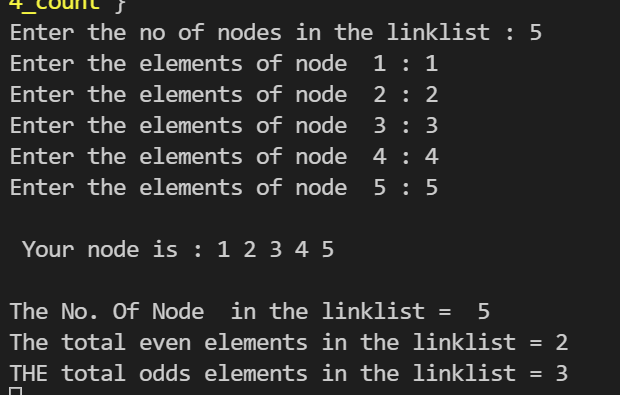
    count(head);

    printf("\n");

   getch();

}

Output:-



Q5 ,

#include <stdio.h>//By Anusthan Singh (20051337)

#include <stdlib.h>

struct node {

    int data;

    struct node \* prev;

    struct node \* next;

}\*head, \*last;

void createList(int n);

void displayListFromStarting();

void displayListFromEnding();

int main()

{

    int n, choice;

    head = NULL;

    last = NULL;

    printf("Enter the number of nodes you want to create: ");

    scanf("%d", &n);

    createList(n);

    printf("\nPress 1 to display list from First ");

    printf("\nPress 2 to display list from End(Transverse) : ");

    scanf("%d", &choice);

    if(choice==1)

    {

        displayListFromStarting();

    }

    else if(choice == 2)

    {

        displayListFromEnding();

    }

    return 0;

}

void createList(int n)

{

    int i, data;

    struct node \*newNode;

    if(n >= 1)

    {

        head = (struct node \*)malloc(sizeof(struct node));

        if(head != NULL)

        {

            printf("Enter data of 1 node: ");

            scanf("%d", &data);

            head->data = data;

            head->prev = NULL;

            head->next = NULL;

            last = head;

            for(i=2; i<=n; i++)

            {

                newNode = (struct node \*)malloc(sizeof(struct node));

                if(newNode != NULL)

                {

                    printf("Enter data of %d node: ", i);

                    scanf("%d", &data);

                    newNode->data = data;

                    newNode->prev = last; // Link new node to the previous node

                    newNode->next = NULL;

                    last->next = newNode; // vice versa Link previous node to the new node

                    last = newNode;    // now new node bcome  last/previous node

                }

                else

                {

                    printf("Unable to allocate memory.");

                    break;

                }

            }

            printf("\nDOUBLY LINKED LIST CREATED \n");

        }

        else

        {

            printf("Unable to allocate memory");

        }

    }

}

void displayListFromStarting()

{

    struct node \* temp;

    int n = 1;

    if(head == NULL)

    {

        printf("List is empty.");

    }

    else

    {

        temp = head;

        printf("\n\nDATA IN THE LIST:\n");

        while(temp != NULL)

        {

            printf("DATA of %d node = %d\n", n, temp->data);

            n++;

            temp = temp->next;                                                    // Move the current pointer to next node

        }

}

}

void displayListFromEnding()

{

    struct node \* temp;

    int n = 0;

    if(last == NULL)

    {

        printf("List is empty.");

    }

    else

    {

        temp = last;

        printf("\n\nDATA IN THE LIST:\n");

        while(temp != NULL)

        {

            printf("DATA of last-%d node = %d\n", n, temp->data);

            n++;

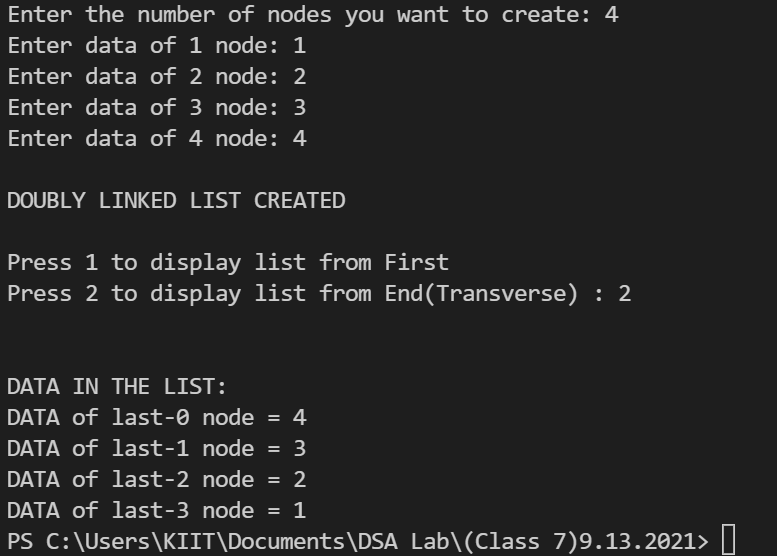
            temp = temp->prev;                                                   // Move the current pointer to previous node

        }

    }

}

Output.



\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*The End\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

By:- Anusthan Singh(20051337)

