01: Simple list implement using C. Delete an item from the list.

Code :

#include<stdio.h>

int main()

{

    int arr[30];

    int n;

    int i;

    printf("Enter the size of the list  ");

    scanf("%d",&n);

    printf("\n\nEnter %d elements ", n);

    for(i=0;i<n;i++)

    {

        scanf("%d",&arr[i]);

    }

    printf("\n\nThe list is \n");

    for(i=0;i<n;i++)

    {

        printf("%d\t",arr[i]);

    }

    printf("\n");

    int k;

    printf("\nEnter the position of data which you want to delete: ");

    scanf("%d",&k);

    while(k<n)

    {

        arr[k-1]=arr[k];

        k++;

    }

    n-=1;

    printf("\nAfter deletion  \n");

    for(i=0;i<n;i++)

    {

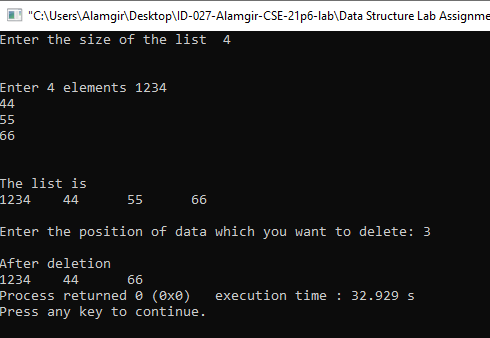
        printf("%d\t",arr[i]);

    }

    return 0;

}

Output :



02: Array implement using C. Delete an item from the Array.

Code :

#include<stdio.h>

int main()

{

    int n, i;

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

    scanf("%d",&n);

    int arr[n];

    printf("\n\nEnter %d elements ", n);

    for(i=0;i<n;i++)

    {

        scanf("%d",&arr[i]);

    }

    printf("\n\nThe array is \n");

    for(i=0;i<n;i++)

    {

        printf("%d\t",arr[i]);

    }

    printf("\n");

    int k;

    printf("\nEnter the position of data which you want to delete: ");

    scanf("%d",&k);

    while(k<n)

    {

        arr[k-1]=arr[k];

        k++;

    }

    n-=1;

    printf("\nAfter deletion  \n");

    for(i=0;i<n;i++)

    {

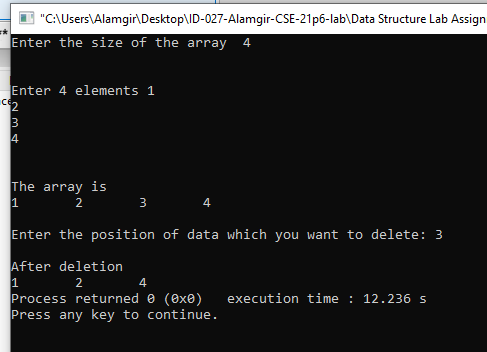
        printf("%d\t",arr[i]);

    }

    return 0;

}

Output :



03: Find the largest/maximum value and it’s location in an array.

Code :

#include<stdio.h>

int main()

{

    int n;

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

    scanf("%d",&n);

    int arr[n];

    printf("\n\nEnter %d elements ", n);

    for(int i=0;i<n;i++){

        scanf("%d",&arr[i]);

    }

    printf("\n\nThe array is \n");

    for(int i=0;i<n;i++)

    {

        printf("%d\t",arr[i]);

    }

    printf("\n");

    int max=arr[0];

    int i,  loc=1;

    for (i=0;i<n;i++)

    {

        if(arr[i]>max)

        {

            max=arr[i];

            loc=i;

        }

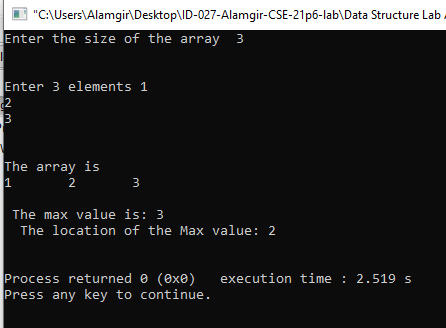
    }

    printf("\n The max value is: %d\n  The location of the Max value: %d\n\n",max,loc);

    return 0;

}

Output :



04: Implement Bhaskara’s formula.

Code :

#include <stdio.h>

#include <math.h>

int main() {

    double a, b, c, pr1,x, y;

    printf("\nInput the first number(a): ");

    scanf("%lf", &a);

    printf("\nInput the second number(b): ");

    scanf("%lf", &b);

    printf("\nInput the third number(c): ");

    scanf("%lf", &c);

    pr1 = (b\*b) - (4\*(a)\*(c));

    if(pr1 > 0 && a != 0) {

        pr1 = sqrt(pr1);

        x = (-b + pr1)/(2\*a);

        y = (-b - pr1)/(2\*a);

        printf("Root1 = %.5lf\n", x);

        printf("Root2 = %.5lf\n", y);

    }

    else

     {

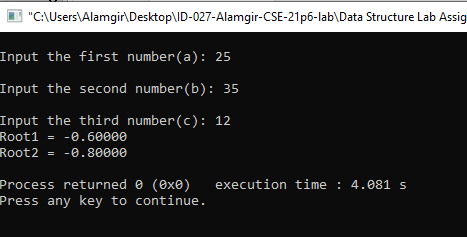
        printf("\nImpossible to find the roots.\n");

    }

    return 0;

}

Output :



05: To search an element in Array using LS\*.

Code :

#include <stdio.h>

void main()

{  int num;

    int i,searchNumber, found = 0;

    printf("Enter the number of elements ");

    scanf("%d", &num);

    int array[num];

    printf("Enter the elements one by one \n");

    for (i = 0; i < num; i++)

    {

        scanf("%d", &array[i]);

    }

    printf("Enter the element to be searched ");

    scanf("%d", &searchNumber);

    /\*  Linear search begins \*/

    for (i = 0; i < num ; i++)

    {

        if (searchNumber == array[i] ) {

            found = 1;

            break;

        }

    }

    if (found == 1)

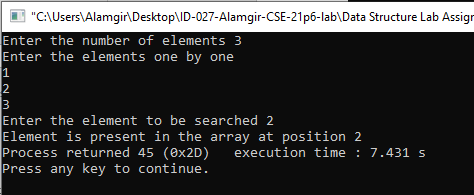
        printf("Element is present in the array at position %d",i+1);

    else

        printf("Element is not present in the array\n");

}

Output :



06: To search an element in a 2D array using LS\*.

Code :

#include<stdio.h>

int main(){

  int rows, columns, srchElement,i,j, count=0;

  printf("Enter the number of Rows and Columns: \n");

  scanf("%d %d", &rows, &columns);

  int array[rows][columns];

  printf("Enter %d elements: \n", (rows\*columns));

  for( i=0; i<rows; i++){

    for(j=0; j<columns; j++){

      scanf("%d", &array[i][j]);

    }

  }

 printf("\nThe array is :\n");

    for(i=0;i<rows;i++){

        for(j=0;j<columns;j++){

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

        }

        printf("\n");

    }

  printf("Enter the element to get the position: \n");

  scanf("%d", &srchElement);

  for(i=0; i<rows; i++){

    for(int j=0; j<columns; j++){

      if(array[i][j] == srchElement){

        printf("(The element is in the row number %d,  and column number %d) \n", i, j);

        count++;

      }

    }

  }

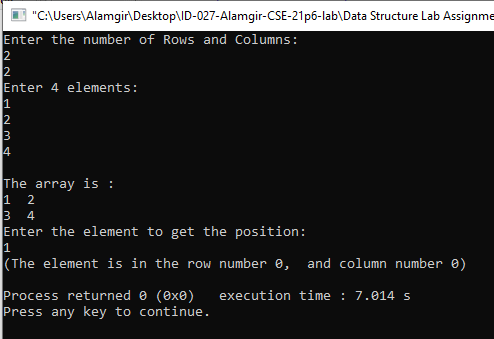
  if(count==0)

    printf("Not found \n");

  return 0;

}

Output :



07: Implement a simple Queue.

Code :

#include <stdio.h>

#define MAX 50

void insert();

void delete();

void display();

int queue\_array[MAX];

int rear = - 1;

int front = - 1;

main(){

    int choice;

    while (1){

        printf("1.Insert element to queue \n");

        printf("2.Delete element from queue \n");

        printf("3.Display all elements of queue \n");

        printf("4.Quit \n");

        printf("Enter your choice : ");

        scanf("%d", &choice);

        switch (choice)

        {

            case 1:

            insert();

            break;

            case 2:

            delete();

            break;

            case 3:

            display();

            break;

            case 4:

            exit(1);

            default:

            printf("Wrong choice \n");

        }

    }

}

void insert()

{

    int add\_item;

    if (rear == MAX - 1)

    printf("Queue Overflow \n");

    else

        {

        if (front == - 1)

        front = 0;

        printf("Inset the element in queue : ");

        scanf("%d", &add\_item);

        rear = rear + 1;

        queue\_array[rear] = add\_item;

    }

}

void delete()

{

    if (front == - 1 || front > rear)

    {

        printf("Queue Underflow \n");

        return ;

    }

    else

    {

        printf("Element deleted from queue is : %d\n", queue\_array[front]);

        front = front + 1;

    }

}

void display()

{

    int i;

    if (front == - 1)

        printf("Queue is empty \n");

    else

    {

        printf("Queue is : \n");

        for (i = front; i <= rear; i++)

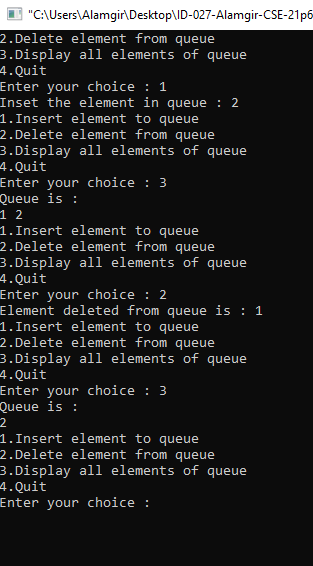
            printf("%d ", queue\_array[i]);

        printf("\n");

    }

}

Output :



08: Design, develop and implement a menu driven program in c for the following Array operations (Don’t use built in functions).

a. Creating an Array of N integer elements.

b. Display of Array elements with suitable headings.

c. Inserting an element (ele) at a given valid position (pos).

d. Deleting an element at a given valid position (pos).

e. Exit.

Code :

#include<stdio.h>

int arr[100],i,size,choice,pos,elem;

void create();

void display();

void insert();

void Delete();

int main()

{ int size;

    while(1){

        printf("1.Create array\n2.Display array\n3.Insert element in array\n4.Delete element of array\n5.Exit the program\nEnter your choice number: ");

    scanf("%d",&choice);

    switch(choice)

    {

    case 1:

        create();

        break;

    case 2:

        display();

        break;

    case 3:

        insert();

        break;

    case 4:

        Delete();

        break;

    case 5:

        exit(0);

        break;

    default:

        printf("invalid input. please enter the number from 1 to 5 \n");}

    printf("\n======================================\n");}

    return 0;

}

void create(){

        int i;

        printf("Enter the array size: ");

        scanf("%d",&size);

        printf("Enter the array element: ");

        for(i=0; i<size; i++){

            scanf("%d",&arr[i]); }

    }

    void display(){

        printf("Resultant array is: \n");

        for(i=0; i<size; i++){

            printf("%d ",arr[i]); }

    }

   void insert(){

        printf("Enter the position number where you insert element: ");

        scanf("%d",&pos);

        if(pos>=0 && pos<=size){

            printf("Enter the value of new element: ");

        scanf("%d",&elem);

        size++;

        for(i=size;i>=pos;i--){

            arr[i] = arr[i-1];

        }

        arr[pos-1] = elem;

        printf("After insertion ");

        display();}

        else{

            printf("Invalid position!Please enter valid position: ");} }

    void Delete(){

    printf("Enter the position number which element you want to delete: ");

    scanf("%d",&pos);

    size--;

    if(pos>=0 && pos<=size){

        for(i=pos-1;i<size;i++){

            arr[i] = arr[i+1];

        }

        printf("After deletion ");

        display();}

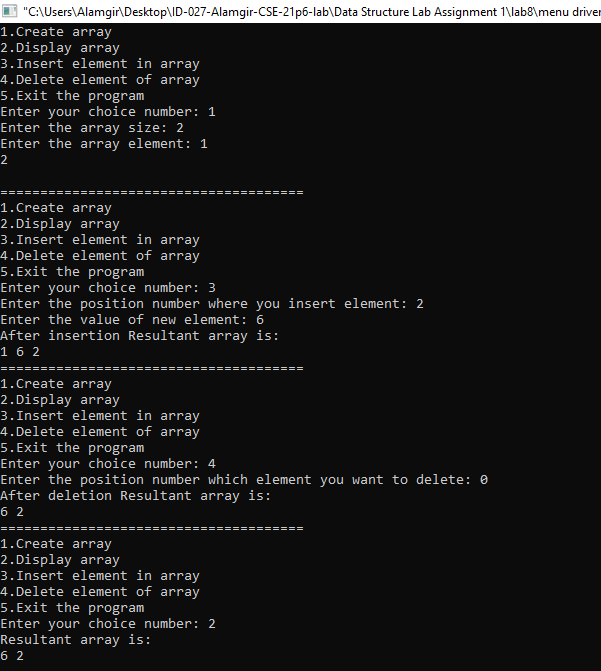
    else{

        printf("Invalid position!Please enter valid position: ");

    }

    }

Output :



09: Design, develop and implement a menu driven program in c for the following operations on strings (Don’t use built in functions).

a. Read a main string (str), a pattern string (pat) and a replacement string

(rep).

b. Perform pattern matching operation: Find and replace all occurrences of

pat in str with rep if pat exists on str. Report with suitable messages in case

pat does not exist in str.

Code :

#include<stdio.h>

char str[50], pat[20], rep[20], nstr[50];

int i, j, k, c, m, flag=0;

void main()

{

    printf("Enter Main string: ");

    gets(str);

    printf("\nEnter a pattern string: ");

    gets(pat);

    printf("\nEnter a replace string: ");

    gets(rep);

    match();

    if(flag==1)

        printf("\n\nThe new string is: %s\n\n", nstr);

    else

        printf("\n\nPattern not found\n\n");

}

void match()

{

    i=m=c=j=0;

    while(str[c]!='\0')

    {

        if(str[m]==pat[i])

        {

            i++;m++;

            if(pat[i]=='\0')

            {

                flag = 1;

                for(k=0;rep[k]!='\0';k++,j++)

                    nstr[j]=rep[k];

                i=0;

                c=m;

            }

        }

        else

        {

            nstr[j]=str[c];

            j++;

            c++;

            i=0;

            m=c;

        }

    }

    nstr[j]='\0';

}

Output :

