**//Program for linear search.**

**#include<iostream.h>**

**int main()**

**{**

**int a[5],i,x,flag=0;**

**cout<<" Enter the array Elements=\n";**

**for(i=0;i<5;i++)**

**{**

**cin>>a[i];**

**}**

**cout<<"The Array Elements are=\n";**

**for(i=0;i<5;i++)**

**{**

**cout<<a[i]<<"\n";**

**}**

**cout<<" Enter the element to find=\n";**

**cin>>x;**

**for(i=0;i<5;i++)**

**{**

**if(x==a[i])**

**{**

**flag=1;**

**break;**

**}**

**}**

**if(flag==0)**

**{**

**cout<<"Element not present in the array=\n";**

**}**

**else**

**{**

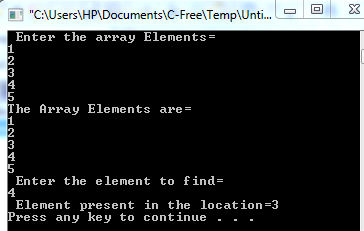
**cout<<" Element present in the location="<<i<<"\n";**

**}**

**return 0;**

**}**

**OUTPUT:**



**//Program for Binary Search.**

**#include<iostream.h>**

**int main()**

**{**

**int a[5],i,x,mid,s,e;**

**cout<<" Enter the array Elements=\n";**

**for(i=0;i<5;i++)**

**{**

**cin>>a[i];**

**}**

**cout<<"The Array Elements are=\n";**

**for(i=0;i<5;i++)**

**{**

**cout<<a[i]<<"\n";**

**}**

**cout<<" Enter the element to find=\n";**

**cin>>x;**

**s=0;**

**e=5;**

**while(s<=e)**

**{**

**mid=(s+e)/2;**

**if(x==a[mid])**

**{**

**cout<<" Element found at ="<<mid+1<<"\n";**

**break;**

**}**

**else if (x>a[mid])**

**{**

**s=mid+1;**

**}**

**else**

**{**

**e=mid-1;**

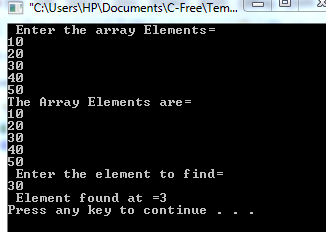
**}**

**}**

**return 0;**

**}**

**OUTPUT:**



**//Program for Bubble Sort.**

**#include<iostream.h>**

**int main()**

**{**

**int a[5],i,j,temp,n=5;**

**cout<<" Enter the 5 array Elements=\n";**

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

**{**

**cin>>a[i];**

**}**

**cout<<"The Array Elements Before Sort=\n";**

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

**{**

**cout<<a[i]<<"\n";**

**}**

**cout<<"Bubble Sort=\n";**

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

**{**

**for(j=0;j<n-1;j++)**

**if(a[j]>a[j+1])**

**{**

**temp=a[j];**

**a[j]=a[j+1];**

**a[j+1]=temp;**

**}**

**}**

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

**{**

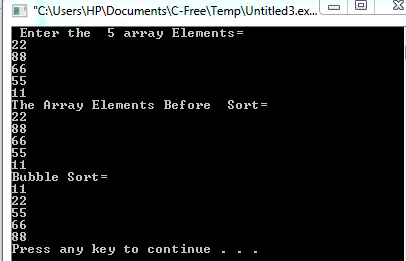
**cout<<a[i]<<"\n";**

**}**

**return 0;**

**}**

**OUTPUT:**



**//Program for Matrix addition.**

**#include<iostream.h>**

**int main()**

**{**

**int mat1[2][2], mat2[2][2], mat3[2][2],i, j;**

**cout<<"Enter matrix 1 elements :";**

**cout<<"\n";**

**for(i=0; i<2; i++)**

**{**

**for(j=0; j<2; j++)**

**{**

**cin>>mat1[i][j];**

**}**

**}**

**cout<<"Enter matrix 2 elements :";**

**cout<<"\n";**

**for(i=0; i<2; i++)**

**{**

**for(j=0; j<2; j++)**

**{**

**cin>>mat2[i][j];**

**}**

**}**

**cout<<"Addition of the two matrix is .....\n";**

**for(i=0; i<2; i++)**

**{**

**for(j=0; j<2; j++)**

**{**

**mat3[i][j]=mat1[i][j]+mat2[i][j];**

**}**

**}**

**cout<<"\n";**

**for(i=0; i<2; i++)**

**{**

**for(j=0; j<2; j++)**

**{**

**cout<<mat3[i][j]<<" ";**

**}**

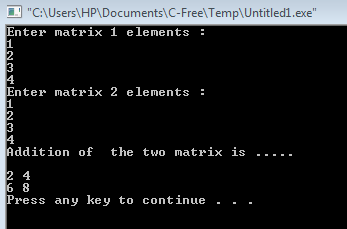
**cout<<"\n";**

**}**

**return 0;**

**}**

**OUTPUT:**



**//Program for Stack.**

**#include <iostream>**

**using namespace std;**

**int stack[100], n=100, top=-1;**

**void push(int val)**

**{**

**if(top>=n-1)**

**cout<<"Stack Overflow"<<endl;**

**else**

**{**

**top++;**

**stack[top]=val;**

**}**

**}**

**void pop()**

**{**

**if(top<=-1)**

**cout<<"Stack Underflow"<<endl;**

**else**

**{**

**cout<<"The popped element is "<< stack[top] <<endl;**

**top--;**

**}**

**}**

**void display()**

**{**

**if(top>=0)**

**{**

**cout<<"Stack elements are:";**

**for(int i=top; i>=0; i--)**

**cout<<stack[i]<<" ";**

**cout<<endl;**

**}**

**else**

**cout<<"Stack is empty";**

**}**

**int main()**

**{**

**int ch, val;**

**cout<<"1) Push in stack"<<endl;**

**cout<<"2) Pop from stack"<<endl;**

**cout<<"3) Display stack"<<endl;**

**cout<<"4) Exit"<<endl;**

**do**

**{**

**cout<<"Enter choice: "<<endl;**

**cin>>ch;**

**switch(ch)**

**{**

**case 1:**

**{**

**cout<<"Enter value to be pushed:"<<endl;**

**cin>>val;**

**push(val);**

**break;**

**}**

**case 2:**

**{**

**pop();**

**break;**

**}**

**case 3:**

**{**

**display();**

**break;**

**}**

**case 4:**

**{**

**cout<<"Exit"<<endl;**

**break;**

**}**

**default:**

**{**

**cout<<"Invalid Choice"<<endl;**

**}**

**}**

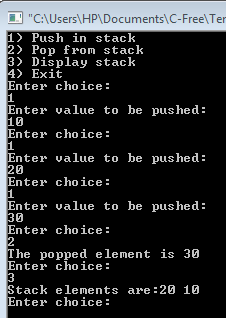
**}**

**while(ch!=4);**

**return 0;**

**}**

**OUTPUT:**



**//program for Queue.**

**#include <iostream>**

**using namespace std;**

**int queue[100], n = 100, front = - 1, rear = - 1;**

**void Insert()**

**{**

**int val;**

**if (rear == n - 1)**

**cout<<"Queue Overflow"<<endl;**

**else**

**{**

**if (front == - 1)**

**front = 0;**

**cout<<"Insert the element in queue : "<<endl;**

**cin>>val;**

**rear++;**

**queue[rear] = val;**

**}**

**}**

**void Delete()**

**{**

**if (front == - 1 || front > rear)**

**{**

**cout<<"Queue Underflow ";**

**return ;**

**}**

**else**

**{**

**cout<<"Element deleted from queue is : "<< queue[front] <<endl;**

**front++;;**

**}**

**}**

**void Display()**

**{**

**if (front == - 1)**

**cout<<"Queue is empty"<<endl;**

**else**

**{**

**cout<<"Queue elements are : ";**

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

**cout<<queue[i]<<" ";**

**cout<<endl;**

**}**

**}**

**int main()**

**{**

**int ch;**

**cout<<"1) Insert element to queue"<<endl;**

**cout<<"2) Delete element from queue"<<endl;**

**cout<<"3) Display all the elements of queue"<<endl;**

**cout<<"4) Exit"<<endl;**

**do {**

**cout<<"Enter your choice : "<<endl;**

**cin>>ch;**

**switch (ch)**

**{**

**case 1: Insert();**

**break;**

**case 2: Delete();**

**break;**

**case 3: Display();**

**break;**

**case 4: cout<<"Exit"<<endl;**

**break;**

**default: cout<<"Invalid choice"<<endl;**

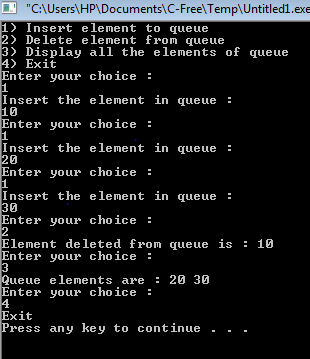
**}**

**} while(ch!=4);**

**return 0;**

**}**

**OUTPUT:**



**// program for Single Linked List.**

**#include <iostream>**

**using namespace std;**

**struct Node**

**{**

**int data;**

**struct Node \*next;**

**};**

**struct Node\* head = NULL;**

**void insert(int new\_data)**

**{**

**struct Node\* new\_node = (struct Node\*) malloc(sizeof(struct Node));**

**new\_node->data = new\_data;**

**new\_node->next = head;**

**head = new\_node;**

**}**

**void display()**

**{**

**struct Node\* ptr;**

**ptr = head;**

**while (ptr != NULL)**

**{**

**cout<< ptr->data <<" ";**

**ptr = ptr->next;**

**}**

**}**

**int main()**

**{**

**insert(3);**

**insert(1);**

**insert(7);**

**insert(2);**

**insert(9);**

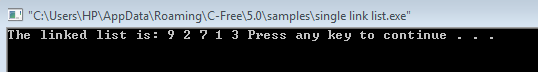
**cout<<"The linked list is: ";**

**display();**

**return 0;**

**}**

**OUTPUT**:



**// Program To Demonstrate Adjancy Matrix Representation of Graph.**

**#include<iostream>**

**#include<iomanip>**

**using namespace std;**

**void PrintMat(int mat[][20], int n)**

**{**

**int i, j;**

**cout<<"\n\n"<<setw(4)<<"";**

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

**cout<<setw(3)<<"("<<i+1<<")";**

**cout<<"\n\n";**

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

**{**

**cout<<setw(3)<<"("<<i+1<<")";**

**for(j = 0; j < n; j++)**

**{**

**cout<<setw(4)<<mat[i][j];**

**}**

**cout<<"\n\n";**

**}**

**}**

**int main()**

**{**

**int i, j, v;**

**cout<<"Enter the number of vertexes: ";**

**cin>>v;**

**int mat[20][20];**

**cout<<"\n";**

**for(i = 0; i < v; i++)**

**{**

**for(j = i; j < v; j++)**

**{**

**if(i != j)**

**{**

**cout<<"Enter 1 if the vertex "<<i+1<<" is adjacent to "<<j+1<<", otherwise 0: ";**

**cin>>mat[i][j];**

**mat[j][i] = mat[i][j];**

**}**

**else**

**mat[i][j] = 0;**

**}**

**}**

**PrintMat(mat, v);**

**}**

**OUTPUT:**

