|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | |  | |
|  | | | |  | |
|  | | | |  | |
|  | | CSPC-205Data Structures and Algorithms | | |
|  | | PARTICIPANTHariom Dhaked19103040Cse 3rd SemesterCOURSE CO-ORDINATORDr Rajneesh Rani | | | | | | | |
|  | | |  | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | |  |
|  | Phone Directory Application (Using Linked List): Phone Directory Application is a system of phone contact information.  This project demonstrate the working of contact book applications using linked list.  Typically, phone book management encompasses searching, sorting, and deleting operations.  A distinctive feature of the search queries here is that the user sees suggestions from the contact list after entering each character. | | |  |
|  | The Program :– | |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  | #include<iostream>  #include <stdlib.h>  #include<string.h>  using namespace std;  struct node  {  string name , number;  node \*next;  };  node \*head = NULL , \*newnode , \*temp;  int len = 0 ;  void C\_node ()  {  newnode = new node;  cout<<"Enter Name ";  cin>>newnode->name;  cout<<"Enter number ";  cin>>newnode->number;  newnode->next = NULL;  if(head == NULL)  {  head = newnode;  temp = newnode;  }  else  {  temp->next= newnode;  temp = newnode;  }  }  void display ()  {  if(head == NULL)  {  cout<<"Contact list is Empty "<<endl;  }  else  {  node \*trav = head ;  while (trav != NULL)  {  cout<<"\n\t\tName :"<<trav->name<<endl;  cout<<"\t\tPhone Number :"<<trav->number<<endl;  trav= trav->next;  len++;  }  cout<<"Total contacts in the list = "<<len<<endl;  }  }  void search\_contact ()  {  node \*search\_node = head;  string srch;  cout<<"Enter your desired contact you want to search ";  cin>>srch;  bool found = false;  if(head == NULL)  {  cout<<"\nList is Empty "<<endl;  }  else  {  while (search\_node != NULL)  {  if(srch == search\_node->name || srch == search\_node->number)  {  cout<<"\n\t\tName: "<<search\_node->name<<endl;  cout<<"\t\tPhone number: "<<search\_node->number<<endl;  found =true;  }  search\_node = search\_node->next;  }  }  if(found == true)  {  cout<<"\t\tContact found"<<endl;  }  else  {  cout<<"Not fount "<<endl;  }  }  void at\_given () ///deletion function  {  int pos ;  node \*next\_node;  temp = head;  cout<<"Enter your desired position from where you want to delete contact "<<endl;  cin>>pos;  if(head == NULL)  {  cout<<"List is Empty "<<endl;  }  else if (pos > len)  {  cout<<"Invalid Position "<<endl;  }  else if(pos == 0)  {  temp = head;  head = head->next;  delete temp;  cout<<"Contact has been deleted "<<endl;  }  else  {  for (int i = 1 ; i<pos ; i++)  {  temp = temp->next;  }  next\_node = temp->next;  temp->next = next\_node->next;  delete next\_node;  cout<<"Contact has been deleted "<<endl;  }  }  void clear\_all ()  {  if(head == NULL)  {  cout<<"List is Empty "<<endl;  }  else  {  while (head != NULL)  {  temp = head ;  delete temp;  head = head->next;  }  cout<<"\n\t\tALL contact list has been deleted "<<endl;  }  }  void my\_swap (node \*node\_1, node \*node\_2)  {  string temp = node\_1->name;  node\_1->name = node\_2 -> name;  node\_2 -> name = temp;  }  void bubble\_sort()  {  int swapped;  node \*lPtr;  node \*rPrt = NULL;  do  {  swapped = 0;  lPtr = head;  while(lPtr->next != rPrt)  {  if (lPtr->name > lPtr->next->name)  {  my\_swap(lPtr, lPtr->next);  swapped = 1;  }  lPtr = lPtr->next;  }  rPrt = lPtr;  }while(swapped);  }  void suggestions ()  {  node \*search\_node = head;  string srch;  cout<<"Enter string you want to search ";  cin>>srch;  bool found = false;  if(head == NULL)  {  cout<<"Contact list is Empty "<<endl;  }  else  {  while (search\_node != NULL)  {  if(search\_node->name.find(srch,0) == 0 )  {  cout<<"\n\t\tName: "<<search\_node->name<<endl;  cout<<"\t\tPhone number: "<<search\_node->number<<endl;  found =true;  }  search\_node = search\_node->next;  }  }  if(found == false)  {  cout<<"Not found "<<endl;  }  }  void menu ()  {  cout<<"Enter 1 to add contact "<<endl;  cout<<"Enter 2 to display all contact "<<endl;  cout<<"Enter 3 to search contact "<<endl;  cout<<"Enter 4 to delete contact from where you want "<<endl;  cout<<"Enter 5 to clear All record "<<endl;  cout<<"Enter 6 to sort "<<endl;  cout<<"Enter 7 to search by suggestion"<<endl;  cout<<"Enter 0 to exit"<<endl;  }  int main ()  {  cout<<" \*\*\*\*\*PHONEBOOK\*\*\*\*\*\n"<<endl;  int op;  while (true )  {  menu();  cin>>op;  switch (op)  {  case 1:  C\_node();  break;  case 2:  len = 0;  display();  break;  case 3:  search\_contact();  break;  case 4:  at\_given();  break;  case 5:  clear\_all();  break;  case 6:  bubble\_sort();  break;  case 7:  suggestions();  break;  case 0:  exit(0);  break;  default:  cout<<"Invalid Option "<<endl;  }  }} |  |  |
|  | OUTPUT:- C:\Users\HP\Pictures\Screenshots\Screenshot (167).png  C:\Users\HP\Pictures\Screenshots\Screenshot (168).png  C:\Users\HP\Pictures\Screenshots\Screenshot (169).png  C:\Users\HP\Pictures\Screenshots\Screenshot (174).png  C:\Users\HP\Pictures\Screenshots\Screenshot (172).png  C:\Users\HP\Pictures\Screenshots\Screenshot (173).png |  |  |
|  | THIS IS THE END. | |  |