EXPERIMETN 1

|  |
| --- |
| #include<iostream> |
|  |  |
|  | using namespace std; |
|  |  |
|  | void insert\_ele( int \*arr,int &n) |
|  | { int pos,data; |
|  | cout<<"Enter the specific position:"<<endl; |
|  | cin>>pos; |
|  | cout<<"Enter Element:"<<endl; |
|  | cin>>data; |
|  | for(int i=n-1;i>=pos;i--) |
|  | { |
|  | arr[i+1] = arr[i]; |
|  | } |
|  | arr[pos] = data; |
|  | n = n+1; |
|  | } |
|  | void delete\_ele\_value(int \*arr,int &n) |
|  | { |
|  | int data,index; |
|  |  |
|  | cout<<"Enter element to be deleted"<<endl; |
|  | cin>> data; |
|  | for(int i = 0;i<n;i++) |
|  | { |
|  | if(data == arr[i]) |
|  | { |
|  | index = i; |
|  | break; |
|  | } |
|  | } |
|  |  |
|  | for(int i = index ;i<n;i++) |
|  | { |
|  | arr[i]= arr[i+1]; |
|  | } |
|  | n--; |
|  |  |
|  | } |
|  |  |
|  | void delete\_ele\_pos(int \*arr,int &n) |
|  | { |
|  | int index; |
|  | cout<<"Enter position of element to be deleted"<<endl; |
|  | cin>> index; |
|  | for(int i = index ;i<n;i++) |
|  | { |
|  | arr[i]= arr[i+1]; |
|  | } |
|  | n--; |
|  |  |
|  | } |
|  |  |
|  |  |
|  | void find\_loc(int \*arr,int &n) |
|  | { |
|  | int ele,i=0; |
|  | cout<<"Enter element:\n"; |
|  | cin>>ele; |
|  | while(arr[i]!= ele && i<n) |
|  | { |
|  | i++; |
|  | } |
|  | if(ele==arr[i]) |
|  | cout<<"Location: "<<i<<endl; |
|  | else |
|  | cout<<"Element not found\n"; |
|  |  |
|  | } |
|  |  |
|  | void display(int \*arr,int &n) |
|  | { |
|  | for(int i=0;i<n;i++) |
|  | { |
|  | cout<<arr[i]<<" "; |
|  | } |
|  | cout<<endl; |
|  |  |
|  | } |
|  |  |
|  | int main() |
|  | { |
|  | int n; |
|  | cout<<"Enter number of elements in an array:"<<endl; |
|  | cin>>n; |
|  | int arr[100]; |
|  | cout<<"Enter Array Elements:"<<endl; |
|  | for(int i=0;i<n;i++) |
|  | { |
|  | cin>>arr[i]; |
|  | } |
|  | int choice = 0; |
|  |  |
|  | do |
|  | { |
|  | cout<<"1.Insert a new element at a specified position."<<endl; |
|  | cout<<"2.Delete an element from value."<<endl; |
|  | cout<<"3.Delete an element from position."<<endl; |
|  | cout<<"4.Find the location of a given element."<<endl; |
|  | cout<<"5.Display the elements of the linear array."<<endl; |
|  | cout<<"6.Exit\n"; |
|  |  |
|  | cin >> choice; |
|  |  |
|  | switch(choice) |
|  | { |
|  | case 1 : insert\_ele(arr,n); |
|  | break; |
|  | case 2 : delete\_ele\_value(arr,n); |
|  | break; |
|  | case 3 : delete\_ele\_pos(arr,n); |
|  | break; |
|  | case 4 : find\_loc(arr,n); |
|  | break; |
|  | case 5 : display(arr,n); |
|  | break; |
|  | case 6 : break; |
|  | default : cout<<"Wrong Input"<<endl; |
|  | } |
|  |  |
|  | }while(choice != 6); |
|  |  |
|  |  |
|  | return 0; |
|  | } |

EXPERIMENT 2

|  |
| --- |
| #include <iostream> |
|  | using namespace std; |
|  |  |
|  | int main() |
|  | { |
|  | int n,ele; |
|  | cout<<"Enter size of array: \n"; |
|  | cin>>n; |
|  | int arr[n]; |
|  |  |
|  | for(int i=0;i<n;i++) |
|  | { |
|  | cin>>arr[i]; |
|  | } |
|  |  |
|  | cout<<"Enter element to search : \n"; |
|  | cin>>ele; |
|  | int pos = -1; |
|  | for(int i=0;i<n;i++) |
|  | { |
|  | if(arr[i]== ele) |
|  | { |
|  | pos=i; |
|  | break; |
|  | } |
|  | } |
|  |  |
|  | if(pos==-1) |
|  | { |
|  | cout<<"Element Not Found "; |
|  | } |
|  | else |
|  | { |
|  | cout<<"Element "<<ele<<" is at "<<pos+1<<" position"; |
|  | } |
|  | return 0; |
|  | } |

EXPERIMENT 3

|  |
| --- |
| #include <iostream> |
|  | using namespace std; |
|  |  |
|  | int main() |
|  | { |
|  | int n,ele; |
|  |  |
|  | cout<<"Enter size of array: \n"; |
|  | cin>>n; |
|  | int arr[n]; |
|  | for(int i=0;i<n;i++) |
|  | { |
|  | cin>>arr[i]; |
|  | } |
|  |  |
|  | for(int i=0;i<n-1;i++) |
|  | { |
|  | for(int j=0;j<n-i-1;j++) |
|  | { |
|  | if(arr[j]>arr[j+1]) |
|  | { |
|  | int temp=arr[j]; |
|  | arr[j]=arr[j+1]; |
|  | arr[j+1]=temp; |
|  | } |
|  | } |
|  | } |
|  |  |
|  | cout<<"Sorted Array: \n"; |
|  | for(int i=0;i<n;i++) |
|  | { |
|  | cout<<arr[i]<<" "; |
|  | } |
|  |  |
|  | cout<<"\nEnter element to search: \n"; |
|  | cin>>ele; |
|  |  |
|  | int beg,end,mid,pos = -1; |
|  | beg=0; |
|  | end=n-1; |
|  |  |
|  | mid=(beg+end)/2; |
|  | while(beg<=end) |
|  | { |
|  | if(arr[mid]==ele) |
|  | { |
|  | pos=mid; |
|  | break; |
|  | } |
|  | else if(arr[mid]<ele) |
|  | { |
|  | beg=mid+1; |
|  | } |
|  | else if(arr[mid]>ele) |
|  | { |
|  | end=mid-1; |
|  | } |
|  | mid=(beg+end)/2; |
|  | } |
|  |  |
|  | if(pos==-1) |
|  | { |
|  | cout<<"Element not found \n"; |
|  | } |
|  | else |
|  | { |
|  | cout<<"Element "<<ele<<" is at "<<pos+1<<" position\n"; |
|  | } |
|  |  |
|  | return 0; |
|  | } |

EXPERIMENT 4

|  |
| --- |
| #include <iostream> |
|  | using namespace std; |
|  |  |
|  | struct node |
|  | { struct node \*prev; |
|  | int data; |
|  | struct node \*next; |
|  | }; |
|  | struct linked\_list |
|  | { |
|  | node \*head,\*tail; |
|  | linked\_list() |
|  | { |
|  | head=NULL; |
|  | tail=NULL; |
|  | } |
|  | node \*createnode(int); |
|  | void insert\_beg(); |
|  | void display(); |
|  | void insert\_end(int a); |
|  | void insert\_after(); |
|  | void insert\_before(); |
|  | void delete\_beg(); |
|  | void delete\_end(); |
|  | void delete\_after(); |
|  | void delete\_before(); |
|  | void search\_element(); |
|  |  |
|  | }; |
|  | node \*linked\_list :: createnode(int x) |
|  | { |
|  | node \*temp = new node; |
|  | temp->prev=NULL; |
|  | temp->data=x; |
|  | temp->next=NULL; |
|  | return temp; |
|  | } |
|  |  |
|  | void linked\_list :: insert\_beg() |
|  | { |
|  | int ele; |
|  | node \*p,\*ptr; |
|  | cout<<"enter element "; |
|  | cin>>ele; |
|  | p=createnode(ele); |
|  | if(head==NULL) |
|  | { |
|  | head=p; |
|  | } |
|  | else |
|  | { |
|  | p->next=head; |
|  | head->prev=p; |
|  | head=p; |
|  | } |
|  | } |
|  | void linked\_list :: insert\_before() |
|  | { |
|  | int data2,x; |
|  | node \*p,\*ptr; |
|  | cout<<"enter element to insert "; |
|  | cin>>data2; |
|  | p=createnode(data2); |
|  | cout<<"enter node to insert before "; |
|  | cin>>x; |
|  | ptr=head; |
|  | while(ptr->data!=x) |
|  | { |
|  | ptr=ptr->next; |
|  | } |
|  | if(ptr->prev==NULL) |
|  | { |
|  | p->next=ptr; |
|  | ptr->prev=p; |
|  | head=p; |
|  | } |
|  | else |
|  | { |
|  | ptr->prev->next=p; |
|  | p->prev=ptr->prev; |
|  | p->next=ptr; |
|  | ptr->prev=p; |
|  | } |
|  |  |
|  |  |
|  | } |
|  | void linked\_list :: insert\_after() |
|  | { |
|  | int data1,x; |
|  | node \*p,\*ptr; |
|  | cout<<"Enter element to insert: \n"; |
|  | cin>>data1; |
|  | p=createnode(data1); |
|  | cout<<"Enter after which node to insert: \n"; |
|  | cin>>x; |
|  | ptr=head; |
|  | while(ptr->data!=x) |
|  | { |
|  | ptr=ptr->next; |
|  | } |
|  | if(ptr->next==NULL) |
|  | { |
|  | ptr->next=p; |
|  | p->prev=ptr; |
|  | } |
|  | else |
|  | { |
|  | p->next=ptr->next; |
|  | ptr->next->prev=p; |
|  | ptr->next=p; |
|  | p->prev=ptr; |
|  |  |
|  | } |
|  |  |
|  |  |
|  | } |
|  | void linked\_list :: insert\_end(int a) |
|  | { |
|  |  |
|  | node \*n,\*ptr; |
|  | n=createnode(a); |
|  | if(head==NULL) |
|  | { |
|  | head=n; |
|  | } |
|  | else |
|  | { |
|  | ptr=head; |
|  | while(ptr->next!=NULL) |
|  | { |
|  | ptr=ptr->next; |
|  | } |
|  | ptr->next=n; |
|  | n->prev=ptr; |
|  | } |
|  |  |
|  | } |
|  |  |
|  |  |
|  | void linked\_list :: display() |
|  | { |
|  | tail=head; |
|  | if(head==NULL) |
|  | { |
|  | cout<<"Linked List is Empty.\n"; |
|  | } |
|  | while(tail!=NULL) |
|  | { |
|  | cout<<tail->data<<" <-> "; |
|  | tail=tail->next; |
|  | } |
|  | cout<<"NULL\n"; |
|  | } |
|  | void linked\_list :: delete\_beg() |
|  | { |
|  | if(head==NULL) |
|  | { |
|  | cout<<"doubly link list is empty.\n"; |
|  | } |
|  | else if(head->next==NULL) |
|  | { |
|  | head=NULL; |
|  | cout<<"No element left in linked list.\n"; |
|  | } |
|  | else |
|  | { |
|  | head=head->next; |
|  | head->prev=NULL; |
|  | } |
|  |  |
|  | } |
|  | void linked\_list :: delete\_end() |
|  | { |
|  | node \*ptr; |
|  | ptr=head; |
|  | if(head==NULL) |
|  | { |
|  | cout<<"Doubly link list is empty\n"; |
|  | } |
|  | else if(head->next==NULL) |
|  | { |
|  | head=NULL; |
|  | cout<<"No element left in ll\n"; |
|  | } |
|  | else |
|  | { |
|  | while(ptr->next!=NULL) |
|  | { |
|  | ptr=ptr->next; |
|  | } |
|  | ptr->prev->next=NULL; |
|  | } |
|  |  |
|  | } |
|  | void linked\_list :: delete\_after() |
|  | { int x; |
|  | node \*ptr; |
|  | cout<<"Enter a node element after which to delete.\n"; |
|  | cin>>x; |
|  | ptr=head; |
|  | while(ptr->data!=x) |
|  | { |
|  | ptr=ptr->next; |
|  | } |
|  | if(ptr->next==NULL) |
|  | { |
|  | cout<<"After can't be performed\n"; |
|  | } |
|  | else if(ptr->next->next==NULL) |
|  | { |
|  | ptr->next=NULL; |
|  | } |
|  | else |
|  | { |
|  | ptr->next=ptr->next->next; |
|  | ptr->next->prev=ptr; |
|  | } |
|  |  |
|  | } |
|  | void linked\_list :: delete\_before() |
|  | { int x; |
|  | node \*ptr; |
|  | cout<<"Enter a node element before which to delete\n"; |
|  | cin>>x; |
|  | ptr=head; |
|  | while(ptr->data!=x) |
|  | { |
|  | ptr=ptr->next; |
|  | } |
|  | if(x==head->data) |
|  | { |
|  | cout<<"Before can't be performed\n"; |
|  | } |
|  | else if(ptr->prev->prev==NULL) |
|  | { |
|  | head=ptr; |
|  | head->prev=NULL; |
|  | } |
|  | else{ |
|  | ptr->prev=ptr->prev->prev; |
|  | ptr->prev->next=ptr; |
|  | } |
|  |  |
|  | } |
|  | void linked\_list :: search\_element() |
|  | { |
|  | int ele; |
|  | int flag=0; |
|  | cout<<"enter element to search : "; |
|  | cin>>ele; |
|  | node \*ptr; |
|  | ptr=head; |
|  | while(ptr!=NULL) |
|  | { |
|  | if(ptr->data==ele) |
|  | { |
|  | flag=1; |
|  | break; |
|  | } |
|  | ptr=ptr->next; |
|  | } |
|  | if(flag==1) |
|  | { |
|  | cout<<"element found in link list\n"; |
|  | } |
|  | else |
|  | { |
|  | cout<<"element not found in link list\n"; |
|  | } |
|  | } |
|  | int main() |
|  | { |
|  | linked\_list l; |
|  | int ch,n,data; |
|  | cout<<"enter size\n"; |
|  | cin>>n; |
|  | cout<<"enter elements\n"; |
|  | for(int i=0;i<n;i++) |
|  | { |
|  | cin>>data; |
|  | l.insert\_end(data); |
|  | } |
|  | cout<<"enter :\n1 to insert at beg\n2 to display\n3 to insert at end\n4 to insert after\n5 to insert before\n6 to delete beg\n7 to delete end\n8 to delete\_after\n9 to delete before\n10 search element\n11 exit\n"; |
|  | cin>>ch; |
|  | while(ch!=11) |
|  | { |
|  | switch(ch) |
|  | { |
|  | case 1 : l.insert\_beg(); |
|  | break; |
|  | case 2 : l.display(); |
|  | break; |
|  | case 3 : |
|  | { |
|  | cout<<"enter element : "; |
|  | cin>>data; |
|  | l.insert\_end(data); |
|  | break; |
|  | } |
|  | case 4 : l.insert\_after(); |
|  | break; |
|  | case 5 : l.insert\_before(); |
|  | break; |
|  | case 6 : l.delete\_beg(); |
|  | break; |
|  | case 7 : l.delete\_end(); |
|  | break; |
|  | case 8 : l.delete\_after(); |
|  | break; |
|  | case 9 : l.delete\_before(); |
|  | break; |
|  | case 10 : l.search\_element(); |
|  | break; |
|  | } |
|  | cout<<"enter :\n1 to insert at beg\n2 to display\n3 to insert at end\n4 to insert after\n5 to insert before\n6 to delete beg\n7 to delete end\n8 to delete\_after\n9 to delete before\n10 search element\n11 exit\n"; |
|  | cin>>ch; |
|  | } |
|  | return 0; |
|  | } |

EXPERIMENT 5

|  |
| --- |
| #include <iostream> |
|  |  |
|  | using namespace std; |
|  | struct node |
|  | { |
|  | int data; |
|  | struct node \*next; |
|  | }; |
|  | class link |
|  | { |
|  | private : |
|  | node \*top; |
|  | public: |
|  | link() |
|  | { |
|  | top=NULL; |
|  | } |
|  | void push(int n) |
|  | { |
|  | node \*temp=new node; |
|  | temp->data=n; |
|  | temp->next=NULL; |
|  | if(top==NULL) |
|  | { |
|  | top=temp; |
|  | } |
|  | else |
|  | { |
|  | temp->next=top; |
|  | top=temp; |
|  | } |
|  | } |
|  | void pop() |
|  | { |
|  | node \*ptr=top; |
|  | top=top->next; |
|  | delete(ptr); |
|  |  |
|  | } |
|  | void display() |
|  | { |
|  | if(top==NULL) |
|  | { |
|  | cout<<"link list is empty\n"; |
|  | } |
|  | else{ |
|  | cout<<top->data<<" "; |
|  | } |
|  |  |
|  | } |
|  |  |
|  | }; |
|  | int main() |
|  | { int n,num,ch; |
|  | link l; |
|  | cout<<"enter size "; |
|  | cin>>n; |
|  | for(int i=0;i<n;i++) |
|  | { |
|  | cin>>num; |
|  | l.push(num); |
|  | } |
|  |  |
|  |  |
|  | do{ |
|  | cout<<"\n1. Push\n2. Pop\n3. Display\n4. Exit\n"; |
|  | cin>>ch; |
|  | switch(ch) |
|  | { |
|  | case 1: |
|  | { |
|  | cout<<"Enter element to push: "; |
|  | cin>>num; |
|  | l.push(num); |
|  | break; |
|  | } |
|  | case 2: l.pop(); |
|  | break; |
|  | case 3: l.display(); |
|  | break; |
|  | } |
|  |  |
|  | } while(ch!=4); |
|  | return 0; |
|  | } |