#include <iostream> //Rohit Kumar

using namespace std; //Metallurgy\_21108053

//CREATING NODE CLASSS

class Node{

public:

//Creating class objects

string name;

int age;

Node\*next; //A node pointer for next node

Node\*prev; //A node pointer for previous node

//calling constructor

Node(string name,int age){

this->name=name;

this->age=age;

next=NULL; //initialising next as NULL

prev=NULL; //initialising prev as NULL

}

};

//FUNCTION TO INSERT DATA AT END OF LINKED LIST

void append(Node\* &head,string name,int age){

//Creating a node pointer and storing address of head in temp

Node\*temp=head;

//creating a new node and storing name and age in it

Node\*new\_node=new Node(name,age);

//Inserting node in empty list

if(temp==NULL){

head=new\_node;

}

//Inserting node in non empty list

else{

while(temp->next!=NULL){temp=temp->next;}

temp->next=new\_node;

new\_node->prev=temp;

}

}

//FUNCTION TO INSERT DATA AT START OF LINKED LIST

void insert\_at\_head(Node\*&head,string name,int age){

//Creating a Node pointer and storing address of new node in it

Node\*new\_node=new Node(name,age);

//Inserting new node at head

new\_node->next=head;

head->prev=new\_node;

head=new\_node;

}

//FUNCTION TO DIPLAY DATA FROM START OF LINKED LIST

void display\_from\_start(Node\*head){

Node\*temp=head;

while(temp!=NULL){cout<<"[Name:"<<temp->name<<" Age:"<<temp->age<<"]"<<"<=>";temp=temp->next;}

cout<<endl;

}

//FUNCTION TO DISPLAY DATA FROM END OF LINKED LIST

void display\_from\_end(Node\*head){

Node\*temp=head;

while(temp->next!=NULL){temp=temp->next;}

while(temp!=NULL){cout<<"[Name:"<<temp->name<<" Age:"<<temp->age<<"]"<<"<=>";temp=temp->prev;}

cout<<endl;

}

//FUNCTION TO DELETE ANY INDEX(0,n-1) EXCLUDING LAST ELEMENT

void delete\_ind(Node\*&head,int i){

if(i==0){

Node\*temp=head;

head=temp->next;

head->prev=NULL;

delete temp;

}

else{

Node\*temp=head;

for(int j=0;j<i;j++){temp=temp->next;}

temp->prev->next=temp->next;

temp->next->prev=temp->prev;

delete temp;

}

}

//FUNCTION TO DELETE LAST ELEMENT

void pop(Node\*&head){

Node\*temp=head;

while(temp->next!=NULL){temp=temp->next;}

temp->prev->next=NULL;

delete temp;

}

int main(){

//Initialising an empty linked list

Node\*head=NULL;

int no\_of\_family\_mem;

//Taking no of family mem as input

cout<<"Enter Number Of Family Members:";cin>>no\_of\_family\_mem;

//Inserting family members details in doubly linked list

for(int i=1;i<=no\_of\_family\_mem;i++){

string name;

int age;

cout<<"Enter Family Member "<<i<<" Name:";cin>>name;

cout<<"Enter Family Member "<<i<<" Age:";cin>>age;

//appending data name and age in doubly linked list

append(head,name,age);

}

cout<<endl;//For space

//Displaying Family details

cout<<"Doubly Linked list with my family members as elements is shown below:"<<endl;

cout<<endl;

display\_from\_start(head);

}

//Rohit Kumar

//Metallurgy\_21108053

/\*BONUS QUESTION

Q.Try to find a way to link the family members' doubly-linked list based on their relationship.

Ans.One way to link the family members doubly-linked list is by sorting the doubly linked list according to age in decending order.By doing so we will get a doubly linked list in which older generation people will be close to head in double linked list and new generation people will be close to tail in doubly linked list.

\*/