By-

Abhilash Tyagi

#include<iostream>

using namespace std;

//definition of structure of animal information

struct animal

{

int IC\_Number;

string Name;

int age;

};

//definition of node of linked list

struct node

{

animal data;

struct node\* next;

};

//definition of class stack

class stack

{

private:

struct node\*top;//to store top position of stack

//to swap the data of two structures of animal type

void swap(animal &a, animal &b)

{

int tempI= a.IC\_Number;

a.IC\_Number=b.IC\_Number;

b.IC\_Number=tempI;

string tempn=a.Name;

a.Name=b.Name;

b.Name=tempn;

int tempa=a.age;

a.age=b.age;

b.age=tempa;

}

public :

//constructor to initialize top as NULL

stack()

{

top=NULL;

}

//function to prompt user for animal data and push it to stack

void push()

{

int num;

string name;

int age;

//prompt user to enter details

cout<<"Enter IC number of the animal:";

cin>>num;

cout<<"Enter name of animal :";

cin>>name;

cout<<"Enter age of animal:";

cin>>age;

cout<<endl;

//dynamically allocating a node and store address in pointer

node\*t = new node;

//storing data in new node

t->data.IC\_Number=num;

t->data.Name=name;

t->data.age=age;

t-> next = NULL;

//if stack is empty

if(top==NULL)

{

top=t;

return;

}

//linking the new node to top element and changing top position

//to new node

t->next= top;

top=t;

};

//to sort by IC number

void sort\_byIC()

{

//if the stack is empty

if (top==NULL)

return ;

node\*p = top;

node\* last =NULL;

//to store whether elements are swapped or not

int swapped;

do

{

swapped=0; // initially no elements are swapped

p = top; // initializing p with top node

//loop will stop when p reaches a node from where

// all elements afterwards are sorted

while( p->next!= last)

{

//swapping with next element if next element is

// smaller than current element

if(p->data.IC\_Number> p->next->data.IC\_Number)

{

swap(p->data,p->next->data);

//as the element is swapped

// flagged to 1(true)

swapped =1;

}

//moving to next node

p=p->next;

}

//assigning node to last which indicates that after last

// all elements are sorted

last=p;

}

while(swapped!=0);//if all the elements are swapped then all elements

// are sorted

}

//To sort by name

void sort\_byname()

{

//If the stock is empty

if(top==NULL)

return;

node\* p=top;

node\* last= NULL;

int swapped;

do

{

swapped = 0;

p=top;

while (p->next!= last)

{

if (p->data.Name > p-> next->data.Name)

{

swap(p->data, p->next->data);

swapped=1;

}

p=p->next;

}

last=p;

}

while (swapped!=0);

}

//To sort by age

void sort\_byage()

{

//If the stack is empty

if(top==NULL)

return;

node\* p=top;

node\* last=NULL;

int swapped;

do

{

swapped = 0;

p=top;

while (p->next != last)

{

if (p->data.age > p->next->data.age)

{

swap(p->data, p->next->data);

swapped = 1;

}

p = p->next;

}

last = p;

}

while (swapped!=0);

}

void u\_changed\_ur\_mind()

{

string x="";

cout<<"Removing your last animal u just added";

cout<<endl;

node\* temp=top;

if(top==NULL){

cout<<"no animal added to remove";

}

else{

x=top->data.Name;

top=top->next;

}

}

void search()

{

string name;

cout<<"Enter name of the animal to be searched ";

cin>>name;

cout<<endl;

node\* p=top;

//Traversing through stack

while (p)

{

//Displaying information if name is found

if (p->data.Name ==name)

{

cout<<"Animal found! Following is the information\n";

cout<<"IC Number: "<< p->data.IC\_Number << endl;

cout<<"Name: "<< p->data.Name << endl;

cout<<"Age: "<<p->data.age << endl << endl;

break;

}

p=p->next;

}

//p reached end means name not found

if(p==NULL)

{

cout<<"Animal not found!\n";

}

}

void display()

{

//If the stock is empty

if (top== NULL)

{

cout<<"No animal is there\n";

return;

}

node\* p=top;

cout<<"\nDisplaying animal information\n";

//Traversing through stack to display information

while(p)

{

cout<<"IC Number: "<<p->data.IC\_Number<<endl;

cout<<"Age: "<<p->data.age<<endl<<endl;

cout<<"Name: "<<p->data.Name<<endl;

p=p->next;

}

}

};

int main()

{

cout<<"\n\n\n\t\t\t\t \_\_ \_\_ \_\_\_ | | \_\_\_ \_\_\_ \_ \_\_ \_\_\_ \_\_\_ "<<endl;

cout<<"\t\t\t\t \\ \\ /\\ / / / \_ \\ | | / \_\_| / \_ \\ | '\_ ` \_ \\ / \_ \\ "<<endl;

cout<<"\t\t\t\t \\ V V / | \_\_/ | | | (\_\_ | (\_) | | | | | | | | \_\_/ "<<endl;

cout<<"\t\t\t\t \\\_/\\\_/ \\\_\_\_| |\_| \\\_\_\_| \\\_\_\_/ |\_| |\_| |\_| \\\_\_\_| "<<endl;

cout<<" ,---. ,---. "<<endl;

cout<<" / /^`.\\.--''''-./,'^\\ \\ "<<endl;

cout<<" \\ \\ \_ \_ / / "<<endl;

cout<<" `./ / \_\_ \_\_ \\ \\,' "<<endl;

cout<<" / /\_O)\_(\_O\\ \\ "<<endl;

cout<<" | .-' \_\_\_ `-. | "<<endl;

cout<<" .--| \\\_/ |--. "<<endl;

cout<<" ,' \\ \\ | / / `."<<endl;

cout<<" / `. `--^--' ,' \\"<<endl;

cout<<" .-^^^^^-. `--.\_\_\_.--' .-^^^^^-."<<endl;

cout<<".-----------/ \\------------------/ \\--------------."<<endl;

cout<<"| .---------\\ /----------------- \\ /------------. |"<<endl;

cout<<"| | `-`--`--' `--'--'-' | |"<<endl;

cout<<"| | | |"<<endl;

cout<<"| | | |"<<endl;

cout<<"| | | |"<<endl;

cout<<"| | | |"<<endl;

cout<<"| |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_| |"<<endl;

cout<<"|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|"<<endl;

cout<<" )\_\_\_\_\_\_\_\_\_\_|\_\_|\_\_\_\_\_\_\_\_\_\_("<<endl;

cout<<" | || |"<<endl;

cout<<" |\_\_\_\_\_\_\_\_\_\_\_\_||\_\_\_\_\_\_\_\_\_\_\_\_|"<<endl;

cout<<" ),-----.( ),-----.("<<endl;

cout<<" ,' ==. \\ / .== `."<<endl;

cout<<" / ) ( \\"<<endl;

cout<<" `===========' `===========' "<<endl;

//Creating a object of class 'stack'

stack animal;

//to store choice when prompted to user

int ch;

cout<<"\n\t\t\t\t\t\@@@@@@@@@@@ HELLO MASTER!!! @@@@@@@@@\n";

cout<<"\n\t\t\t\t@@@@@@ WELCOME TO YOUR VIRTUAL ANIMAL SANCTUARY!! @@@@@@@@@\n";

cout<<"\n";

cout<<"\t\t\t\t\t\t\t\tMenu: \n";

cout<<"\t\t\t\t\t1. Add your Animal here\n\t\t\t\t\t2. Sort\_by IC Number \n\t\t\t\t\t3. Sort\_by Name\n";

cout<<"\t\t\t\t\t4. Sort\_byage\n\t\t\t\t\t5. Search\n\t\t\t\t\t6. Display\n\t\t\t\t\t7. Exit\n\t\t\t\t\t\8. remove\n\n";

do

{

//Prompt user to enter choice

cout<<"Enter your choice> ";

cin>>ch;

switch(ch)

{

case 1: animal.push();

break;

case 2: animal.sort\_byIC();

break;

case 3: animal.sort\_byname();

break;

case 4: animal.sort\_byage();

break;

case 5: animal.search();

break;

case 6: animal.display();

break;

case 7: cout << "exit"<<endl;

break;

case 8: animal.u\_changed\_ur\_mind();

break;

default: cout<<"Invalid choice\n";

}

}

while(ch!=7);

return 0;

};