**NAME : KHUSHI PANWAR**

**ROLL NO: 2021334**

**DATA STRUCTURES PRACTICAL**

**2. WAP using templates to sort a list of elements. Give user the option to perform sorting using Insertion sort, bubble sort and selection sort.**

#include <iostream>

using namespace std;

const int size=5;

template <class t> class sorting{

t arr[size];

t sorted[size];

public:

void addElement();

void bubbleSort();

void insertionSort();

void selectionSort();

void display();

void displaySorted();

sorting(){

}

};

template<class t> void sorting<t>::addElement(){

cout<<"\n-> Add the elements in the list : ";

for (int i=0; i<size; i++){

cin>>arr[i];

}

}

template<class t> void sorting<t>::display(){

cout<<"-> Sorted List : ";

for (int i=0; i<size; i++){

cout<<"\t"<<arr[i];

}

}

template<class t> void sorting<t>::bubbleSort(){

for (int i=0; i<size; i++){

for (int j=i+1; j<size; j++){

if (arr[i]>arr[j]){

t temp=arr[j];

arr[j]=arr[i];

arr[i]=temp;

}

}

}

}

template<class t> void sorting<t>::insertionSort(){

t key;

for (int i=0; i<size; i++){

key=arr[i];

int j=i-1;

while(j>=0 && arr[j]>key){

arr[j+1]=arr[j];

--j;

}

arr[j+1]=key;

}

}

template<class t> void sorting<t>::selectionSort(){

int mn;

for (int i=0; i<size-1; i++){

mn=i;

for (int j=i+1; j<size; j++){

if(arr[j]<arr[mn]){

mn=j;

}

t temp=arr[mn];

arr[mn]=arr[i];

arr[i]=temp;

}

}

}

void operationList(){

cout<<endl<<endl<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<"\t --> OPERATIONS AVAILABLE : "<<endl;

cout<<"1. Bubble sort"<<endl;

cout<<"2. Selection sort "<<endl;

cout<<"3. Insertion Sort"<<endl;

cout<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl<<endl;

}

template <class t> void performOperations(sorting<t> obj){

int choice;

char ch='y';

obj.addElement();

operationList();

cout<<endl<<"-> Enter your choice (1/2/3) : ";

cin>>choice;

switch(choice){

case 1: cout<<" \n \t OPERATION CHOOSEN :: BUBBLE SORT "<<endl;

obj.bubbleSort();

obj.display();

break;

case 2: cout<<" \n \t OPERATION CHOOSEN :: SELECTION SORT "<<endl;

obj.selectionSort();

obj.display();

break;

case 3: cout<<" \n \t OPERATION CHOOSEN :: INSERTION SORT "<<endl;

obj.insertionSort();

obj.display();

break;

default: cout<<"!! Invalid Operation !!"<<endl;

}

return ;

}

int main()

{

cout << endl<<"\t \t \*\* CREATE MENU DRIVEN PROGRAM OF SORTING USING TEMPLATES \*\* \t \t" << endl;

int ch;

char chh='y';

while(chh=='y' || chh=='Y'){

cout<<"\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \n-> What type of list do you want to create? \n1. String \n2. Integer \n3. Float \nEnter your choice : ";

cin>>ch;

if (ch==1){

sorting<string> obj;

performOperations(obj);

}else if (ch==2){

sorting<int> obj;

performOperations(obj);

}else if (ch==3){

sorting<float> obj;

performOperations(obj);

}else{

cout<<" ~!! INVALID CHOICE : TRY AGAIN LATER !!~"<<endl<<endl;

}

cout<<"\n \n Do you want to continue (y/n) :";

cin>>chh;

cout<<endl;

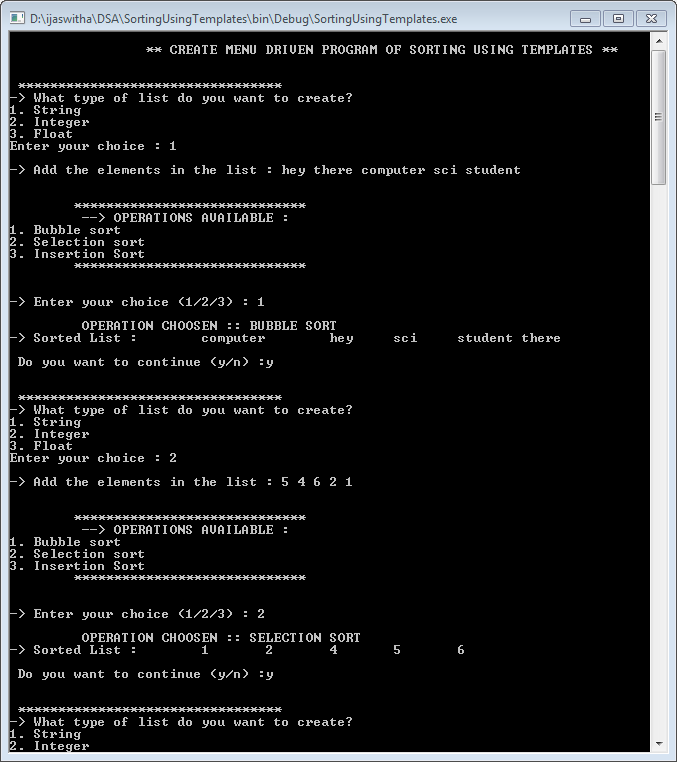
}

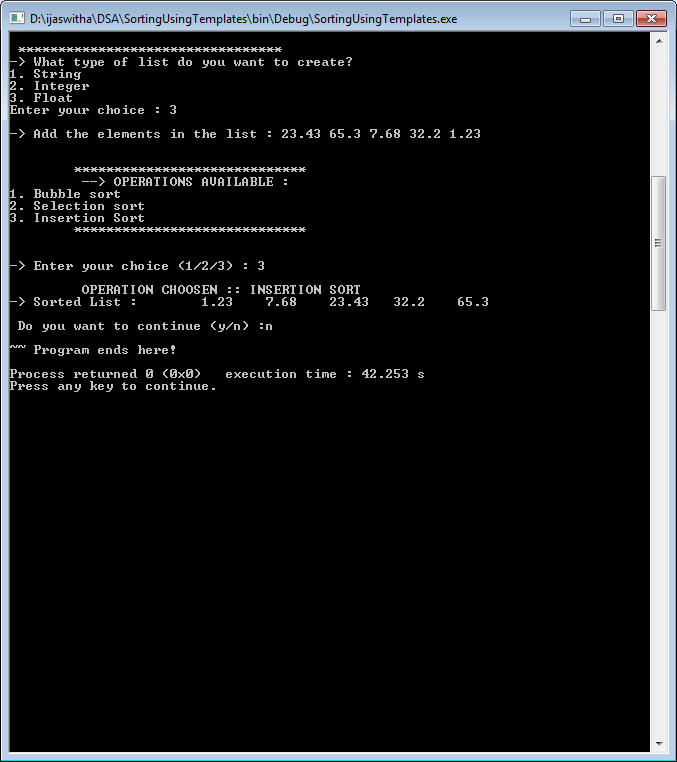
cout<<"~~ Program ends here!"<<endl;

return 0;

}

**Output:**

****

****

**s**

**16. WAP to reverse the order of elements in a stack using additional stack :**