National University of Computer and Emerging Sciences



**Lab Exercise 14**

For

# Object Oriented Programming Lab

|  |  |
| --- | --- |
| Name | Muhammad Zain |
| Roll No | 19F-0228 |
| Course Instructor(s) | Dr. Danish |
| Lab Instructor(s) | Mr. Mughees Ismail |
| Semester | Spring 2020 |

**FAST School of Computing**

**Question#1**

Write a template function swap to perfrom swap between any two values.

# Source Code:

#include<iostream>

using namespace std;

template <class t1>

void Swap(t1 a, t1 b)

{

cout << "Before Swaping " << endl;

cout << "1st number = " << a << endl;

cout << "2nd number = " << b << endl;

t1 temp;

temp = a;

a = b;

b = temp;

cout << "After Swaping " << endl;

cout << "1st number = " << a << endl;

cout << "2nd number = " << b << endl;

}

int main()

{

float value1;

float value2;

cout << "Input 2 values " << endl;

cout << "Value 1 : ";

cin >> value1;

cout << "Value 2 : ";

cin >> value2;

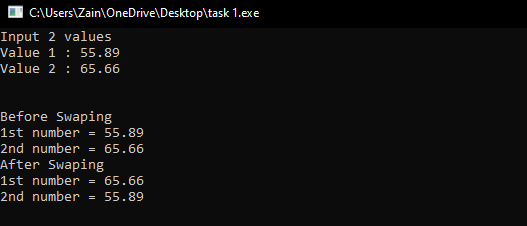
cout << endl << endl;

Swap(value1, value2);

system("pause>0");

}

# Snip:



**Question#2**

Write a **template** function **sort()** to sort any array. The parameters will be array and its size.

# Source Code:

#include<iostream>

using namespace std;

template<class t1>

void sort(t1 Array[], int size)

{

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

{

for (int j = 0; j<size; j++)

{

if (Array[j]>Array[i])

{

t1 temp;

temp = Array[i];

Array[i] = Array[j];

Array[j] = temp;

}

}

}

}

template<class t2>

void output(t2 Array[], int size)

{

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

{

cout << i + 1 << " element of array = " << Array[i] << endl;

}

}

int main()

{

double \*Array;

int size;

cout << "Input the size of array" << endl;

cin >> size;

Array = new double[size];

cout << endl << "Input the elements of Array" << endl;

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

{

cout << i + 1 << " element = ";

cin >> Array[i];

}

cout << endl << endl << "Unsorted Array " << endl;

output(Array, size);

cout << endl << endl << "Sorted Array " << endl;

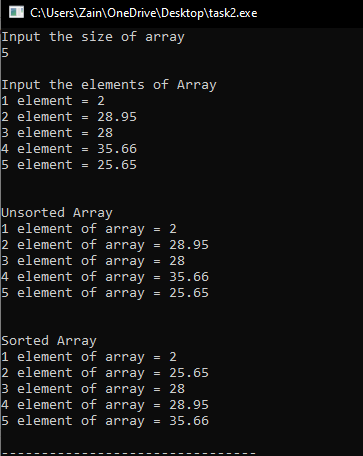
sort(Array, size);

output(Array, size);

system("pause>0");

}

# Snip:



## Question#3

Write a template class **Calculator** which will perform the functionalities of calculator having two private attributes and having functions add, subtract, multiply, divide.

# Source Code:

#include<iostream>

using namespace std;

template <class t1>

class Calculator

{

private:

t1 value1;

t1 value2;

public:

Calculator() {}

Calculator(t1 a, t1 b)

{

value1 = a;

value2 = b;

}

t1 Addition()

{

return (value1 + value2);

}

t1 Subtraction()

{

return (value1 - value2);

}

t1 Multiplication()

{

return (value1\*value2);

}

t1 Division();//just to learn the syntax of definig function outside class

};

template <class t1> t1 Calculator<t1>::Division()

{

return value1 / value2;

}

int main()

{

cout << "\t\tCalculator" << endl;

double int1;

double int2;

cout << "Input 2 values " << endl;

cout << "Value 1 : ";

cin >> int1;

cout << "Value 2 : ";

cin >> int2;

Calculator <double>obj(int1, int2);

cout << endl << endl;

cout << "Addition : " << obj.Addition() << endl;

cout << "Subtraction : " << obj.Subtraction() << endl;

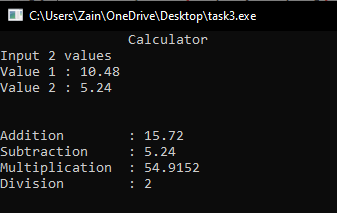
cout << "Multiplication : " << obj.Multiplication() << endl;

cout << "Division : " << obj.Division() << endl;

system("pause>0");

}

# Snip:



## Question#4

# Source Code:

#include<iostream>

using namespace std;

template <class t1>

class Vector {

t1 capacity;

int Size;

t1\* Array;

public:

Vector() { }

Vector(int n)//parametrized constructor

{

Size = n;

Array = new t1[n];

capacity = 1;

}

//Copy Constructor

Vector(const Vector& Obj\_1) {

this->ptr = NULL;

this->size = Obj\_1.Size;

this->ptr = new t1[Size];

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

this->ptr[i] = Obj\_1.ptr[i];

}

}

void setArray(int n)

{

cout << endl << "Input the elements of Array" << endl;

for (int i = 0; i<n; i++)

{

cout << i + 1 << " element = ";

cin >> Array[i];

}

}

bool empty()

{

for (int i = 0; i < Size; i++)

if (Array[i] != 0)

return false;

return true;

}

int Front()

{

return Array[0];

}

int Last()

{

return Array[Size];

}

void push(int data) //- It push\_back the elements into a vector from the back

{

if (Size == capacity)

{

int\* temp = new int[2 \* capacity];

for (int i = 0; i < capacity; i++)

{

temp[i] = Array[i];

}

delete[] Array;

capacity \*= 2;

Array = temp;

}

Array[Size] = data;

Size++;

}

t1 mplace()// - It extends the container by inserting new element at position.

{

cout << "Enter the max size of the array " << endl;

int max;

cin >> max;

return max;

}

void insert(int input, int index) //It assigns new value to the vector elements by replacing old ones

{

if (index == capacity)

push(input);

else

Array[index] = input;

}

void assign(int input, int index) //It assigns new value to the vector elements by replacing old ones

{

if (index == capacity)

push(input);

else

{

for (int i = 0; i<input; i++)

{

Array[input] = index;

}

}

}

int at(int g) //- Returns a reference to the element at position 'g' in the vector

{

if (g < Size)

return Array[g];

}

int\* getat(int i)

{

return &Array[i];

}

void pop\_back() //-It is used to pop or remove elements from a vector from the back.

{

Size--;

}

int getSize()

{

return Size;

}

int getcapacity()

{

return capacity;

}

void output()

{

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

cout << Array[i] << " ";

}

cout << endl;

}

};

int main()

{

int x;

cout << "Input the size of array" << endl;

cout << "Size = "; cin >> x;

Vector <int>obj\_v(x);

obj\_v.setArray(x);

cout << "Input the values at last of the array ....If u want to terminate press -1" << endl;

int temp;

while (1)

{

cout << "Value : ";

cin >> temp;

if (temp == -1)

break;

obj\_v.push(temp);

}

cout << "Maximum Array \n" << obj\_v.mplace() << endl;

cout << "The Size of vector is = " << obj\_v.getSize() << endl;

cout << "The Capacity of the vector is = " << obj\_v.getcapacity() << endl;

cout << "The Elements of Vector are = "; obj\_v.output(); cout << endl << endl;

cout << "Input the position on which you want to insert value" << endl;

int pos;

cout << " Position = "; cin >> pos;

int val;

cout << "Enter the value you want to replace " << endl;

cout << " Value = "; cin >> val;

obj\_v.insert(val, pos);

cout << "After updating " << pos << " position" << endl;

cout << "Vector elements : ";

obj\_v.output();

cout << endl;

cout << "Input the position from which you want to assing value" << endl;

int po;

cout << " Position = "; cin >> pos;

int va;

cout << "Enter the value you want to replace " << endl;

cout << " Value = "; cin >> val;

obj\_v.assign(va, po);

cout << "After updating " << endl;

cout << "Vector elements : ";

obj\_v.output();

cout << endl;

int any;

cout << endl << "Input the index of which you want to get the value" << endl;

cout << "Index number = ";

cin >> any;

cout << "The value at this index is = " << obj\_v.at(any) << endl;

obj\_v.pop\_back();

cout << endl;

cout << "After deleting last element" << endl;

cout << "The new Size of vector is = " << obj\_v.getSize() << endl;

cout << "The Capacity of the vector is = " << obj\_v.getcapacity() << endl;

cout << "The updated Elements of Vector are = "; obj\_v.output(); cout << endl;

cout << "The Refrence of First index is = " << obj\_v.Front() << endl;

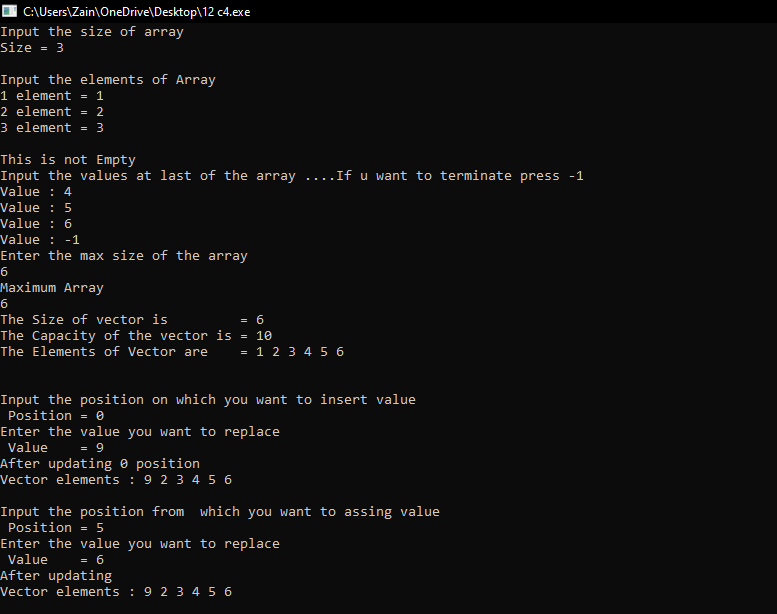
cout << "The Refrence of Last index is = " << obj\_v.Last() << endl;

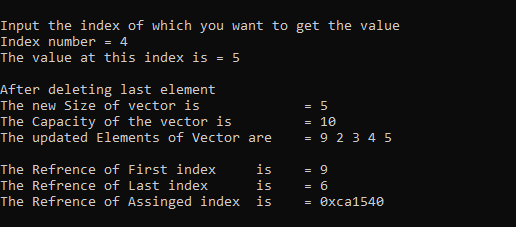
cout << "The Refrence of Assinged index is = " << obj\_v.getat(any) << endl;

system("Pause>0") ;

# }

# Snip:





# Happy Coding 😊