UNIVERSITY OF ENGINEERING AND TECHNOLOGY

(NAROWAL CAMPUS)

Object-Oriented Programming Lab Manual

Created by: Muhammad Abdullah

Registration Number: 2022-CS-525

Topics: Postfix and Prefix increment Operator Overload,

Template Function, Template Class, Exception Handling (No argument),

& Exception Handling (Multi Argument)

Lab Manual

(Object-Oriented Programming Lab)

- **Task 1:**

Overload a Postfix and Prefix increment operator for a user-defined datatype in a C++ program.

Program:

```cpp
#include <iostream>
using namespace std;

class Counter
{
private:
    unsigned int count;

public:
    Counter() : count(0)
    {
    }
    Counter(int c) : count(c)
    {
    }
    unsigned int get_count() const
    {
        return count;
    }
    Counter operator++()
    {
        return Counter(++count);
    }
    Counter operator++(int)
    {
        return Counter(count++);
    }
};

int main()
{
    Counter c1, c2;
    cout << "\nc1=" << c1.get_count();
    cout << "\nc2=" << c2.get_count();
    ++c1;
    c2 = ++c1;
    cout << "\nc1=" << c1.get_count();
    cout << "\nc2=" << c2.get_count();
    c2 = c1++;

    cout << "\nc1=" << c1.get_count();
    cout << "\nc2=" << c2.get_count() << endl;
    return 0;
}
```

Output:

```
c1=0
c2=0
c1=2
c2=2
c1=3
c2=2

D:\UET Narowal\2nd Semester\Object Oriented Programming\Lab\OOP Lab\x64\Debug\OOP Lab.exe (process 19632) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

- ## Task 2:

Create a Template Function in a C++ program.

Program:

```cpp
#include <iostream>
using namespace std;

template <class T>
T abs(T n)
{
    return (n < 0) ? -n : n;
}

int main()
{
    int int1 = 5;
    int int2 = -6;
    long lon1 = 70000L;
    long lon2 = -80000L;
    double dub1 = 9.95;
    double dub2 = -10.15;

    cout << "abs(" << int1 << ")=" << abs(int1);
    cout << "\nabs(" << int2 << ")=" << abs(int2);
    cout << "\nabs(" << lon1 << ")=" << abs(lon1);
    cout << "\nabs(" << lon2 << ")=" << abs(lon2);
    cout << "\nabs(" << dub1 << ")=" << abs(dub1);
    cout << "\nabs(" << dub2 << ")=" << abs(dub2);
    cout << endl;
    return 0;
}
```

Output:

```
abs(5)=5
abs(-6)=6
abs(70000)=70000
abs(-80000)=80000
abs(9.95)=9.95
abs(-10.15)=10.15

D:\UET Narowal\2nd Semester\Object Oriented Programming\Lab\OOP Lab\x64\Debug\OOP Lab.exe (process 17000) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

- **Task 3:**

Create a Template Class in the C++ program.

Program:

```cpp
#include <iostream>
using namespace std;
const int MAX = 100;

template <class Type>
class Stack
{
private:
    Type st[MAX];
    int top;

public:
    Stack()
    {
        top = -1;
    }
    void push(Type var)
    {
        st[++top] = var;
    }
    Type pop()
    {
        return st[top--];
    }
};

int main()
{
    Stack<float> s1;
    s1.push(1111.1F);
    s1.push(2222.2F);
    s1.push(3333.3F);
    cout << "1: " << s1.pop() << endl;
    cout << "2: " << s1.pop() << endl;
    cout << "3: " << s1.pop() << endl;
    Stack<long> s2;
    s2.push(123123123L);
    s2.push(234234234L);
    s2.push(345345345L);
    cout << "1: " << s2.pop() << endl;
    cout << "2: " << s2.pop() << endl;
    cout << "3: " << s2.pop() << endl;
    return 0;
}
```

Output:

```
1: 3333.3
2: 2222.2
3: 1111.1
1: 345345345
2: 234234234
3: 123123123

D:\UET Narowal\2nd Semester\Object Oriented Programming\Lab\OOP Lab\x64\Debug\OOP Lab.exe (process 19824) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

- **Task 4:**

Handle the Error of a class using exception handling.

Program:

```
#include <iostream>
using namespace std;
const int MAX = 3;
class Stack
{
private:
    int st[MAX];
    int top;

public:
    class Range
    {
    };
    Stack()
    {
        top = -1;
    }
    void push(int var)
    {
        if (top >= MAX - 1)
            throw Range();
        st[++top] = var;
    }
    int pop()
    {
        if (top < 0)
            throw Range();
        return st[top--];
    }
};

```

```
int main()
{
    Stack s1;
    try
    {
        s1.push(11);
        s1.push(22);
        s1.push(33);

        cout << "1: " << s1.pop() << endl;
        cout << "2: " << s1.pop() << endl;
        cout << "3: " << s1.pop() << endl;
        cout << "4: " << s1.pop() << endl;
    }
    catch (Stack::Range)
    {
        cout << "Exception: Stack Full or Empty" << endl;
    }
    cout << "Arrive here after catch (or normal exit)" << endl;
    return 0;
}
```

Output:

```
1: 33
2: 22
3: 11
Exception: Stack Full or Empty
Arrive here after catch (or normal exit)

D:\UET Narowal\2nd Semester\Object Oriented Programming\Lab\OOP Lab\x64\Debug\OOP Lab.exe (process 20300) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

- **Task 5:**

Handle an error of a class and detect the member function which throws an error by giving multiple parameters.

Program:

```
#include <iostream>
#include <string>
using namespace std;
class Distance
{
private:
    int feet;
    float inches;
public:
    class InchesEx
    {
    public:
        string origin;
        float iValue;
        InchesEx(string ori, float in)
        {
            origin = ori;
            iValue = in;
        }
    };
```

```
    Distance()
    {
        feet = 0;
        inches = 0.0;
    }
    Distance(int ft, float in)
    {
        if (in >= 12.0)
            throw InchesEx("2-arg constructor", in);
        feet = ft;
        inches = in;
    }
    void getdist()
    {
        cout << "\nEnter feet: ";
        cin >> feet;
        cout << "Enter inches: ";
        cin >> inches;
        if (inches >= 12.0)
            throw InchesEx("getdist() function", inches);
    }
    void showdist()
    {
        cout << feet << " feet ," << inches << " inches ";
    }
};
```

```
int main()
{
    try
    {
        Distance dist1(17, 3.5);
        Distance dist2;
        dist2.getdist();
        cout << "\ndist1 = ";
        dist1.showdist();
        cout << "\ndist2 = ";
        dist2.showdist();
    }
    catch (Distance::InchesEx ix)
    {
        cout << "\n Initialization error in " << ix.origin<< ".\n Inches value of " << ix.iValue<< " is too large.";
    }
    cout << endl;
    return 0;
}
```

Output:

```
Enter feet: 12
Enter inches: 34

 Initialization error in getdist() function.
 Inches value of 34 is too large.

D:\UET Narowal\2nd Semester\Object Oriented Programming\Lab\OOP Lab\x64\Debug\OOP Lab.exe (process 4880) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```
