TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PURWANCHAL CAMPUS

DHARAN

LABSHEET 4 : [CONSTRUCTORS AND DESTRUCTORS]

BY

Rabin Poudel **[PUR079BCT056]**

DEPARTMENT OF COMPUTER & ELECTRONICS ENGINEERING

PURWANCHAL CAMPUS

DHARAN, NEPAL

JULY, 2024

0.1 Question 1 :

Write a program that has a class to represent time. The class should have constructors to initialize data members hour, minute, and second to 0 and to initialize them to values passed as arguments. The class should have a member function to add time objects and return the result as a time object. There should be functions to display time in 12-hour and 24-hour format.

Code:

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

class Time
{
private:
    int h, m, s;
public:
    Time()
    {
        h = 0;
        m = 0;
        s = 0;
    }
    Time(int dh, int dm, int ds)
    {
        h = dh;
        m = dm;
        s = ds;
    }
    Time add_time(Time t1, Time t2)
    {
        t1.s = t1.s + t2.s;
        if (t1.s > 60)
        {
            t1.m++;
            t1.s = t1.s − 60;
        }
```

```
                t1.m = t1.m + t2.m;
                if (t1.m > 60)
                {
                        t1.h++;
                        t1.m = t1.m − 60;
                }
                t1.h = t1.h + t2.h;
                return t1;
        }
        void disp_result()
        {
                cout << "Result = " << h << ’:’ << m << ’:’ << s;
        }
};

int main()
{
        Time t1(6, 35, 17);
        Time t2(7, 19, 59);
        Time disp;
        disp = disp.add_time(t1, t2);
        disp.disp_result();
        return 0;
}
```

0.2 Question 2:

Write a program that has a class with a dynamically allocated character array as its data member. One object should contain "Engineers are" and another should contain " Creatures of logic". Member function join() should concatenate two strings by passing two objects as arguments. Display the concatenated string through a member function. Use constructors to allocate and initialize the data member. Also, write a destructor to free the allocated memory for the character array. Make your own function for the concatenation of two strings.

Code:

```
#include <iostream>
#include <cstring>
using namespace std;
class str
{
	char  *sent;
	int length;

public:
	str()
	{
		length = 0;
		sent = new char[length + 1];
	}
	str(char s[])
	{
		length = strlen(s);
		sent = new char[length + 1];
		strcpy(sent, s);
	}
	void join(str &s1, str &s2)
	{
		length = s1.length + s2.length;
		delete sent;
		sent = new char[length + 1];
		strcpy(sent, s1.sent);
```

```
        strcat(sent, s2.sent);
    }
    void display()
    {
        cout << sent << endl;
    }
};
int main()
{
    str s1("Engineers are"), s2(" Creatures of logic"), d;
    d.join(s1, s2);
    d.display();
}
```

0.3 Question 3:

Write a class that can store Department ID and Department Name with constructors to initialize its members. Write destructor member in the same class and display the message "Object n goes out of the scope". Your program should be made such that it should show the order of constructor and destructor invocation.

Code:

```
#include <iostream>
using namespace std;
class Department
{
private:
        int deptID;
        string deptName;

public:
        Department(int id, string name)
        {
                deptID = id;
                deptName = name;
                cout << "Object " << deptName << " is created" << endl;
        };
        ~Department()
        {
                cout << "Object " << deptName << " goes out of the scope" <<
                    endl;
        };
};

int main()
{
        Department d1(1, "CSE");
        Department d2(2, "ECE");
        Department d3(3, "ME");
        return 0;
}
```

0.4 Question 4:

Assume that one constructor initializes data member say num-vehicle, hour, and rate. There should be a 10 percent discount if num-vehicle exceeds 10. Display the total charge. Use two objects and show a bit-by-bit copy of one object to another (make your own copy constructor).

Code:

```
#include <iostream>
using namespace std;
class Vehicle
{
private:
        int num_vehicle;
        int hour;
        int rate;

public:
        Vehicle(int num_vehicle, int hour, int rate);
        Vehicle(const Vehicle &v);
};

Vehicle::Vehicle(int vn, int h, int r)
{
        this->num_vehicle = vn;
        this->hour = h;
        this->rate = r;
        if (num_vehicle > 10)
        {
                cout << "Total charge: " << (num_vehicle * hour * rate) *
                      0.9 << endl;
        }
        else
        {
                cout << "Total charge: " << num_vehicle * hour * rate <<
                      endl;
        }
};
```

```
Vehicle::Vehicle(const Vehicle &v)
{
	num_vehicle = v.num_vehicle;
	hour = v.hour;
	rate = v.rate;
	cout << "Bit−by−bit copy of one object to another" << endl;
	cout << "Number of vehicles: " << num_vehicle << endl;
	cout << "Hour: " << hour << endl;
	cout << "Rate: " << rate << endl;
}

int main()
{
	Vehicle v1(50, 1, 10);
	Vehicle v2(v1);
	return 0;
}
```

0.5 Question 5:

Write a program that illustrates the following relationship and comment on them.

i) const_object.non_const_mem_function

ii) const_object.const_mem_function

iii) non_const_object.non_const_mem_function

iv) non_const_object.const_mem_function

Code:

```
#include <iostream>

using namespace std;
class Class
{
private:
        const int id;

public:
        Class(int i) : id(i){};

        int const_get() const
        {
                return id;
        }
        int get()
        {
                return id;
        }
};
int main()
{
        Class a(1);
        cout << "non const object calling non const mem func " <<
               a.get() << endl;          // non const object. non const mem func
        cout << "non const object calling const mem func " <<
        a.const_get() << endl; // non const object. const mem
    const Class c(2);
    cout << "const object calling const mem func " << c.const_get();
        // const object.cont mem func
    //    cout << c.get(); cannot be called
    return 0;
}
```

0.6 Question 6:

Create a class with a data member to hold a "serial number" for each object created from the class. That is, the first object created will be numbered 1, the second 2, and so on by using the basic concept of static data members. Use static member function if it is useful in the program. Otherwise, make a separate program that demonstrates the use of static member function.gs.

Code:

```
#include <iostream>
using namespace std;
class SerialNumber
{
private:
	static int nextSerialNumber;
	int serialNumber;

public:
	SerialNumber()
	{
		serialNumber = nextSerialNumber++;
	}

	int getSerialNumber() const
	{
		return serialNumber;
	}
	static int getNextSerialNumber()
	{
		return nextSerialNumber;
	}
};

int SerialNumber::nextSerialNumber = 1;

int main()
```

```
{
        SerialNumber obj1;
        SerialNumber obj2;
        SerialNumber obj3;
        cout << "Object 1 Serial Number: " << obj1.getSerialNumber() <<
               endl;
        cout << "Object 2 Serial Number: " << obj2.getSerialNumber() <<
               endl;
        cout << "Object 3 Serial Number: " << obj3.getSerialNumber() <<
               endl;
        cout << "Next Serial Number: " <<
               SerialNumber::getNextSerialNumber() << endl;
        return 0;
}
```