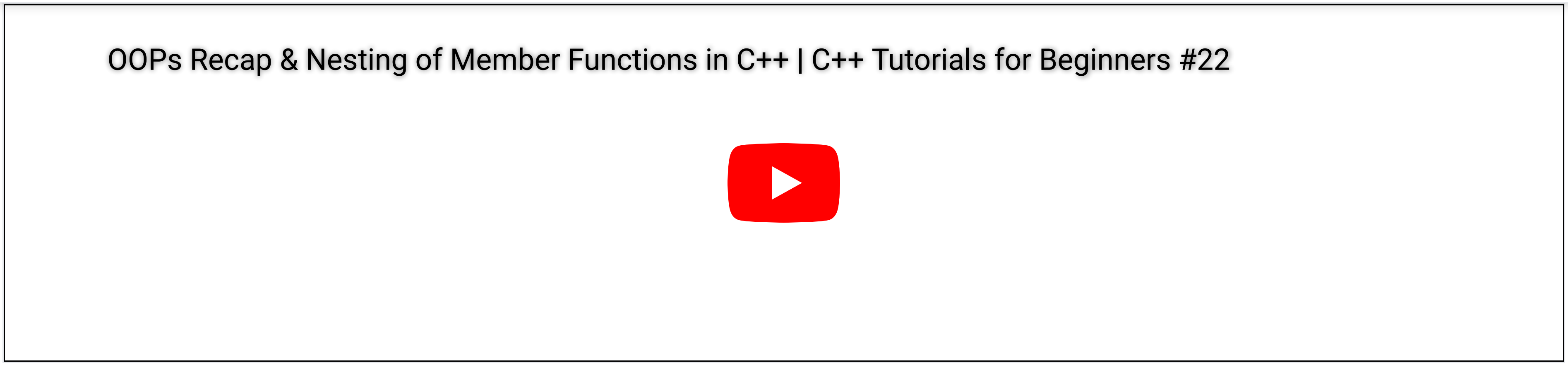
OOPs Recap & Nesting of Member Functions in C++ | C++ Tutorials for Beginners #22

OOPs Recap & Nesting of Member Functions in C++ | C++ Tutorials for Beginners #22

In this tutorial, we will discuss the nesting of a member function in C++

Object-Oriented programming Recap

- Stroustrup initially named C++ language as C with classes because C++ language was almost the same as C language but they added a new concept of classes in it.
- Classes are the extension of structures in C language.
- Structures had limitations such as; members are public and no methods.
- Classes have some additional futures than structures such as; classes that can have methods and

properties.

- Classes have a feature to make class members as public and private.
- In C++ objects can be declared along with class deceleration as shown in Code Snippet 1.

```
class Employee{
            // Class definition
} harry, rohan, lovish;
```

Code Snippet 1: Declaring Objects with Class Declaration

Nesting of Member Functions

If one member function is called inside the other member function of the same class it is called nesting of a member function. A program to demonstrate the nesting of a member function is shown below.

```
class binary
{
private:
    string s;
    void chk_bin(void);

public:
    void read(void);
    void ones_compliment(void);
    void display(void);
};
```

<u>Code Snippet 2: Binary Class</u>

As shown in Code Snippet 2, we created a binary class that has, “s” string variable and “chk_bin” void function as private class members; and “read” void function, “ones_compliment” void function, and “display” void function as public class members. The definitions of these functions are shown below.

```
void binary::read(void)
{
    cout << "Enter a binary number" << endl;
    cin >> s;
}
```

<u>Code Snippet 3: Read Function</u>

As shown in Code Snippet 3, we have created a “read” function. This function will take input from the user at runtime.

```
void binary::chk_bin(void)
{
    for (int i = 0; i < s.length(); i++)
    {
        if (s.at(i) != '0' && s.at(i) != '1')
        {
            cout << "Incorrect binary format" << endl;
            exit(0);
        }
    }
}
```

Code Snippet 4: Check Binary Function

As shown in Code Snippet 4 we have created a "chk_bin" function. This "for" loop in the function will run till the length of the string and "if" condition in the body of the loop will check the whole string that if there are any values in the string other than "1" and "0". If there are values other than "1" and "0" this function will output "Incorrect binary format".

```
void binary::ones_compliment(void)
{
    chk_bin();
    for (int i = 0; i < s.length(); i++)
    {
        if (s.at(i) == '0')
        {
            s.at(i) = '1';
        }
       else
        {
            s.at(i) = '0';
        }
    }
}
```

Code Snippet 5: One's Compliment

As shown in Code Snippet 5, in the body of the "ones_compliment" function; the "chk_bin" function is called, and as we have discussed above that if one member function is called inside the other member function of the same class it is called **nesting of a member function**. The "for" loop inside the "ones_compliment" functions runs till the length of the string and the "if" condition inside the loop replaces the number "0" with "1" and "1" with "0".

```
void binary::display(void)
{
    cout<<"Displaying your binary number"<<endl;
    for (int i = 0; i < s.length(); i++)
    {
        cout << s.at(i);
    }
    cout<<endl;
}
```

Code Snippet 6: Display Function

As shown in Code Snippet 6, the "for" loop inside display function runs till the length of the string and prints each value of the sting.

```
int main()
{
    binary b;
    b.read();
    // b.chk_bin();
    b.display();
    b.ones_compliment();
    b.display();

    return 0;
}
```

Code Snippet 7: Main Function

As shown in Code Snippet 7, we created an object “b” of the binary data type, and the functions “read”, “display”, “ones_compliment”, and “display” are called. The main thing to note here is that the function ”chk_bin” is the private access modifier of the class so we cannot access it directly by using the object, it can be only accessed inside the class or by the member function of the class.

Code as described/written in the video

```
// OOPs - Classes and objects

// C++ --> initially called --> C with classes by stroustroup
// class --> extension of structures (in C)
// structures had limitations
//      - members are public
//      - No methods
// classes --> structures + more
// classes --> can have methods and properties
// classes --> can make few members as private & few as public
// structures in C++ are typedefed
// you can declare objects along with the class declarion like this:
/* class Employee{
```

```
                // Class definition
            } harry, rohan, lovish; */
// harry.salary = 8 makes no sense if salary is private

// Nesting of member functions

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

class binary
{
private:
    string s;
    void chk_bin(void);

public:
    void read(void);
    void ones_compliment(void);
    void display(void);
};

void binary::read(void)
{
    cout << "Enter a binary number" << endl;
    cin >> s;
```

```
}

void binary::chk_bin(void)
{
    for (int i = 0; i < s.length(); i++)
    {
        if (s.at(i) != '0' && s.at(i) != '1')
        {
            cout << "Incorrect binary format" << endl;
            exit(0);  → error code 0.
        }
    }
}

void binary::ones_compliment(void)
{
    chk_bin();
    for (int i = 0; i < s.length(); i++)
    {
        if (s.at(i) == '0')
        {
            s.at(i) = '1';
        }
       else
        {
            s.at(i) = '0';
        }
    }
}

void binary::display(void)
{
    cout<<"Displaying your binary number"<<endl;
    for (int i = 0; i < s.length(); i++)
    {
        cout << s.at(i);
    }
    cout<<endl;
}

int main()
{
    binary b;
    b.read();
    // b.chk_bin();
    b.display();
    b.ones_compliment();
    b.display();

    return 0;
}
```

Previous Next