**Content 52**

### Virtual Functions Examples And Rules in C++

#### Rules for virtual functions

1. They cannot be static
2. They are accessed by object pointers
3. Virtual functions can be a friend of another class
4. A virtual function in the base class might not be used.
5. If a virtual function is defined in a base class, there is no necessity of redefining it in the derived class

#### Virtual Functions Example in C++

As we have seen in the previous tutorial that how virtual functions are used to implement run-time polymorphism. In this tutorial, we will see an example of virtual functions.

#include <iostream>

#include <cstring>

using namespace std;

class CWH

{

protected:

    string name;

    float rating;

public:

    CWH(string n, float r)

    {

        name = n;

        rating = r;

    }

    void display() {}

};

class Video : public CWH

{

protected:

    float video\_length;

public:

    Video(string n, float t, float vl) : CWH(n, t)

    {

        video\_length = vl;

    }

    void display(void)

    {

        cout << "The Name is: " << name << endl;

        cout << "The Ratings are: " << rating << endl;

        cout << "The Video Length is: " << video\_length << endl;

    }

};

class Text : public CWH

{

protected:

    int num\_text;

public:

    Text(string n, float r, int nt) : CWH(n, r)

    {

        num\_text=nt;

    }

    void display(void)

    {

        cout << "The Name is: " << name << endl;

        cout << "The Ratings are: " << rating << endl;

        cout << "The Text Length is: " << num\_text << endl;

    }

};

int main()

{     string title;

    float rating, vlen;

    int words;

    // for Code With Harry Video

    title = "Django tutorial";

    vlen = 4.56;

    rating = 4.89;

    Video djVideo(title, rating, vlen);

    // for Code With Harry Text

    title = "Django tutorial Text";

    words = 433;

    rating = 4.19;

  Text djText(title, rating, words);

    CWH \*tuts[2];

    tuts[0] = &djVideo;

    tuts[1] = &djText;

    tuts[0]->display();

    tuts[1]->display();

    return 0;

}