Innovative Apporach: Case-based Teaching Based & Flipped Classroom in C++

Among programming languages, C++ programming language is relatively difficult to learn. In order to help students, learn C++ language better, two solutions are proposed by the Teacher.

The first solution is to introduce the idea of flipped classroom into C++ teaching. Because of the many difficulties in C++, students can watch and discuss with their classmates anytime and anywhere through the video explanation of the flipped classroom. Teachers explain the key points and difficulties in offline classrooms, which will help students quickly master C++ knowledge points.

The second solution is to design a case based on a real project and then integrate the case into daily C++ teaching. In this way, students can quickly transition from the level of knowledge mastery to the level of knowledge application, which helps to cultivate students' coding ability based on real project.

Introduction

Among programming languages, C++ programming language is relatively difficult to learn. Because it not only has C++ language pointers, but also includes an understanding of object-oriented modeling. So it is very necessary for the research on C++ teaching. In traditional C++ teaching, the C++ textbooks on the market are written according to the knowledge points. Generally speaking, each knowledge point in C++ textbooks is presented with a small example. The advantage of doing so is that students can have a better grasp of the specific application of the knowledge point, but the disadvantage is that it is difficult for students to grasp the role of the knowledge point in specific practice from a more general and macroscopic perspective.

Our solution to the above problem in this article is to use a case to run through all the knowledge points of C++. In this way, students can understand the usage of each knowledge point in specific actual projects.This is more conducive for students to master the knowledge points.

On the other hand, in order to ensure the effect of students' learning C + +, we introduce flipped classroom into our daily teaching. Before class, students can learn the video of the course first. In the class, the teacher introduces project examples, puts forward the specific needs of the project according to the knowledge points of the class, guides the students to think step by step, and finally completes the mastery of the knowledge points while completing the specific project. This teaching mode has two advantages. The first is that flipped classroom gives students more preparation time for learning before class, and the second is to strengthen students' understanding of knowledge points in specific practice with the method of case teaching. Practice has proved that this is a better method.

Introduction to Flipped Classroom Teaching Method

The idea of a flipped classroom was initially attributed to two chemistry teachers in American high schools, Jon Bergmann and Aaron Sams[4]. Around 2007, they encountered the situation that students were unable to attend class for a long time due to their illness. In order to help these students who can't come to school for a long time to keep up with the teaching progress. The two teachers made videos of the course and uploaded them to the Internet for students to watch. After a period of time, many other students also like this way of learning by watching videos.

The two teachers changed their teaching methods so that students in the class watched videos after class. During class, the teacher was responsible for completing homework or difficult problems in the experiment. In this way, the traditional teaching habit of "teachers explain in class, students do homework after class" has changed to "students preview by video before class and teachers guide students to complete homework or experiment in class". This is the origin of flipped classroom.

The development of the flipped classroom is divided into two stages.

The first stage is related to the NesoAcademy Videos.

https://www.youtube.com/watch?v=s0g4ty29Xgg

The second stage of flipped classroom development is closely related to the rise of NPTEL, SWAYAM- MOOC. The full name of NPTEL" National Programme on Technology Enhanced Learning "MOOC is "Massive Open Online Courses". SWAYAM MOOCs platform is World's Largest Online Free E-Learning Platform Portal designed to achieve the three cardinal principles of Education Policy viz., Access, Equity and Quality

The difference between the MOOC and the previous online courses is more emphasis on two points:

one is to emphasize "interaction and feedback"; the other is to advocate the establishment of an "online learning community". Before the MOOC, teachers put video on the Internet for students to browse and learn. Basically, there is no interaction teachers and students and feedback from student.

The video in the MOOC encourages students to add interactive questions and quizzes in the video. At the same time, students are encouraged to learn deeply by searching various resources on the Internet.

Introduction to Case Teaching Methods

Case teaching method means that teachers carefully design case that can cover all the knowledge points, and then guide students to discuss the knowledge points involved in the case, thus students can master the knowledge points and skillfully use them in specific actual case.

The transformation of students from knowledge transfer to knowledge understanding and then to knowledge application is completed. Case teaching can string all the knowledge points of the course together, so that students have a general grasp of what they have learned. Because the cases used in case teaching are real examples in reality, it is of great help to improve students' practical ability

The Application of Flipped Classroom In C++ Teaching

In the teaching of C++, we use the video of the flipped classroom from the national-level excellent course Online. The C++ excellent course is divided into two courses, one is basic knowledge course and the other is advanced knowledge course. Limited to the teaching plan and content of the course, part of the C++ content taught in our class is in the basic knowledge part of the excellent course, and part in the advanced knowledge part.

Generally speaking, since the textbooks of the excellent course are basically the same as the content of the textbooks used in our class, which makes it more convenient for students to use the textbooks we used for learning.

The essence of flipped classroom is from the traditional teacher-centered learning model to student-centered, which makes students' learning time more flexible, and students can even use fragmented time to learn, which greatly improves students' freedom. In addition, students can pause while listening to the video, and then go to the Internet to search for information related to the knowledge point and have a detailed in-depth understanding.

At the same time, we set up a QQ group for course discussion to guide students to discuss in the group. Students can help each other, and teachers can also participate in student discussions and explain the key and difficult points of the course. When conducting classroom teaching, teachers should organize classroom teaching well, guide students to group discussion on key points and difficulties. As for grouping, it is recommended to set a limit on the number of students about every group first, and then let the students choose group freely.

When grouping, the students should know that the teacher should evaluate the situation of the group discussion in the future. The teacher will set up a group discussion assessment form, and then score the contents of the form according to the group discussion, which can stimulate the students' learning enthusiasm.

In this way, the original teacher-centered teaching model can be changed to a new student-centered teaching model, which can greatly stimulate students' independent learning ability and increase students' enthusiasm for participating in classroom interaction.

Design Ideas of Case Teaching Method in C++

Object Oriented Programming With A Real-World Scenario Object Oriented Programming (OOP) concept theoretically but are unable to link it with real world & programming world. We write programs to solve our problems and get our work done.

Object Oriented Programming is considered as a design methodology for building non-rigid software. In OOPS, every logic is written to get our work done, but represented in form of Objects. OOP allows us to break our problems into small unit of work that is represented via objects and their functions. We build functions around objects.

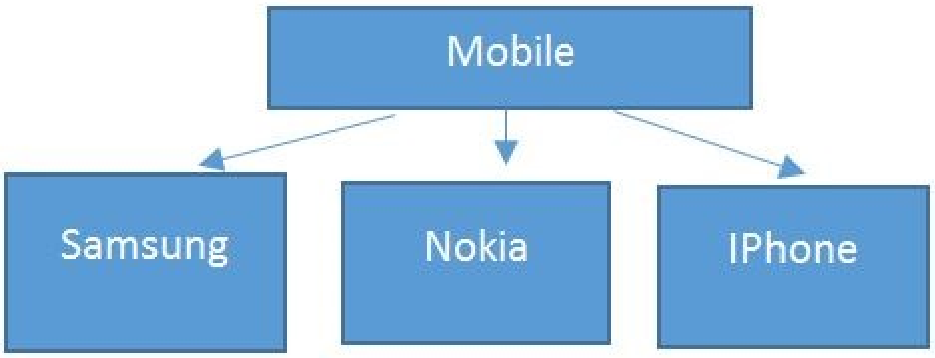
There are mainly four pillars (features) of OOP. If all of these four features are presented in programming, the programming is called perfect Object Oriented Programming.

1. Abstraction
2. Encapsulation
3. Inheritance
4. Polymorphism

Let's consider an example explaining each of these pillars so you can better understand Object Oriented Programming.

Before that, we need to know something.

When we think of a mobile phone as an object, its basic functionality for which it was invented were Calling & Receiving a call & Messaging. But now a days thousands of new features and models were added and the features and number of models are still growing.



In above diagram, each brand (Samsung, Nokia, IPhone) have their own list of features along with basic functionality of dialing, receiving a call and messaging.

Objects

Any real world entity which can have some characteristics or which can perform some tasks is called as Object. This object is also called an instance i.e. a copy of entity in programming language. If we consider the above example, a mobile manufacturing company can be an object. Each object can be different based on their characteristics. For example, here are two objects.

```
Mobile mbl1 = new Mobile ();
Mobile mbl2 = new Mobile ();
```

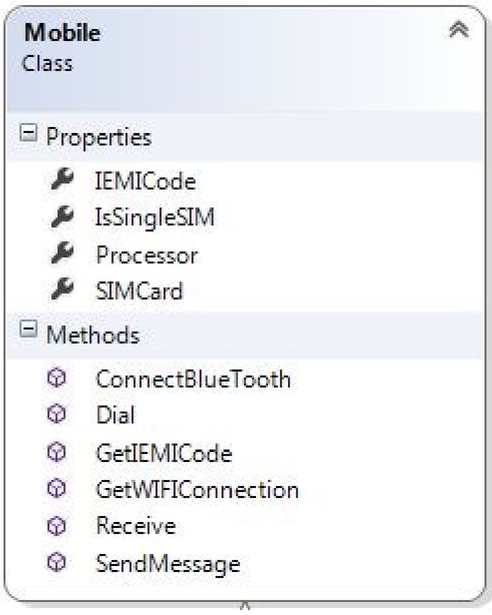
Class

A class in OOP is a plan which describes the object. We call it a blueprint of how the object should be represented. Mainly a class would consist of a name, attributes, and operations. Considering the above example, the Mobile can be a class, which has some attributes like Profile Type, IMEI Number, Processor, and some more. It can have operations like Dial, Receive and SendMessage.

There are some OOPS principle that need to be satisfied while creating a class. This principle is called as SOLID where each letter has some specification. I won't be going into this points deeper. A single line of each explanation may clear you with some points.

1. SRP (The Single Responsibility Principle) - A class should have one, and only one responsibility
2. OCP (The Open Closed Principle) - You should be able to extend a classes behavior, without modifying it. (Inheritance)
3. LSP (The Liskov Substitution Principle) - Derived classes must be substitutable for their base classes. (Polymorphism)
4. ISP (The Interface Segregation Principle) -Make fine chopped interface instead of huge interface as client cannot be forced to implement an interface which they don't use.
5. DIP (The Dependency Inversion Principle) - Depend on abstractions, not on concretions. (Abstraction)

Now, let's take a look at our class. A typical class based on the above discussion looks like the following in Visual Studio:



In C++ programming language, a class has members. These members are called properties, methods, fields, constructors, destructors, events, and so on.
In code, the Mobile class looks like the following:

```
public class Mobile
{
     private string IEMICode { get; set; }
     public string SIMCard { get; set; }
     public string Processor { get; }
     public int InternalMemory { get; }
     public bool IsSingleSIM { get; set; }

     public void GetIEMICode()
            {
                Console.WriteLine("IEMI Code - IEDF34343435235");
            }

            public void Dial()
            {
                Console.WriteLine("Dial a number");
            }
            public void Receive()
            {
                Console.WriteLine("Receive a call");
            }
            public virtual void SendMessage()
            {
                Console.WriteLine("Message Sent");
            }
       }
```

Abstraction

Abstraction allows us to expose limited data and functionality of objects publicly and hide the actual implementation. It is the most important pillar in OOPS. In our example of Mobile class and objects like Nokia, Samsung, IPhone.

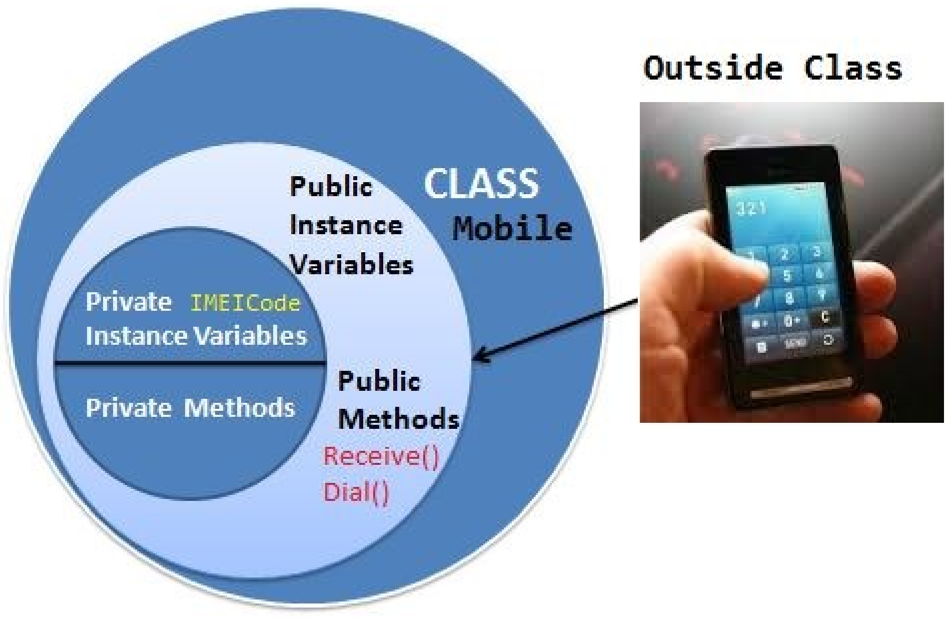
Some features of mobiles,

1. Dialing a number call some method internally which concatenate the numbers and displays it on screen but what is it doing we don't know.
2. Clicking on green button actual send signals to calling person's mobile but we are unaware of how it is doing.

This is called abstraction. In classes, we can create methods that can be called and used by the users of the class but users will have no idea what these methods do internally.

```
public void Dial()
{
   //Write the logic
   Console.WriteLine("Dial a number");
}
```

Encapsulation



Encapsulation is defined as the process of enclosing one or more details from outside world through access right. It says how much access should be given to particular details. Both Abstraction & Encapsulation works hand in hand because Abstraction says what details to be made visible and Encapsulation provides the level of access right to that visible details. i.e. – It implements the desired level of abstraction.

Talking about Bluetooth which we usually have it in our mobile. When we switch on a Bluetooth, I am able to connect to another mobile or bluetooth enabled devices but I'm not able to access the other mobile features like dialing a number, accessing inbox etc. This is because, Bluetooth feature is given some level of abstraction.

Another point is when mobile A is connected with mobile B via Bluetooth whereas mobile B is already connected to mobile C then A is not allowed to connect C via B. This is because of accessibility restriction.

In C#, a class has access modifiers such as public, private, protected, and internal. These access modifiers allows properties and methods to be exposed or restricted to the outside world.

```
private string IMEICode = "76567556757656";
```

Polymorphism

Polymorphism can be defined as the ability of using the same name for doing different things. More precisely we say it as 'many forms of single entity'. This play a vital role in the concept of OOPS.

Let's say Samsung mobile has a 5MP camera available i.e. – it is having a functionality of CameraClick(). Now same mobile is having Panorama mode available in camera, so functionality would be same but with mode. This type is said to be Static polymorphism or Compile time polymorphism.

See the example below:

```
public class Samsumg : Mobile
{
    public void GetWIFIConnection()
    {
        Console.WriteLine("WIFI connected");
    }

    //This is one mwthod which shows camera functionality
    public void CameraClick()
         {
             Console.WriteLine("Camera clicked");
         }

         //This is one overloaded method which shows camera functionality as well but with its camera's different mode(panaroma)
         public void CameraClick(string CameraMode)
         {
             Console.WriteLine("Camera clicked in " + CameraMode + " Mode");
         }
     }
```

Compile time polymorphism the compiler knows which overloaded method it is going to call.

Compiler checks the type and number of parameters passed to the method and decides which method to call and it will give an error if there are no methods that matches the method signature of the method that is called at compile time.

Another point where in SendMessage was intended to send message to single person at a time but suppose Nokia had given provision for sending message to a group at once. i.e. - Overriding the functionality to send message to a group. This type is called Dynamic polymorphism or Runtime polymorphism.

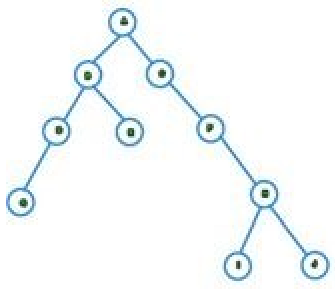
For overriding you need to set the method, which can be overridden to virtual & its new implementation should be decorated with override keyword.

```
public class Nokia : Mobile
{
      public void GetBlueToothConnection()
      {
          Console.WriteLine("Bluetooth connected");
      }

      //New implementation for this method which was available in Mobile Class
      //This is runtime polymorphism
            public override void SendMessage()
            {
                Console.WriteLine("Message Sent to a group");
            }
       }
```

By runtime polymorphism, we can point to any derived class from the object of the base class at runtime that shows the ability of runtime binding.

Inheritance



Inheritance is the ability to extend the functionality from base entity in new entity belonging to same group. This will help us to reuse the functionality which is already defined before and extend into a new entity.

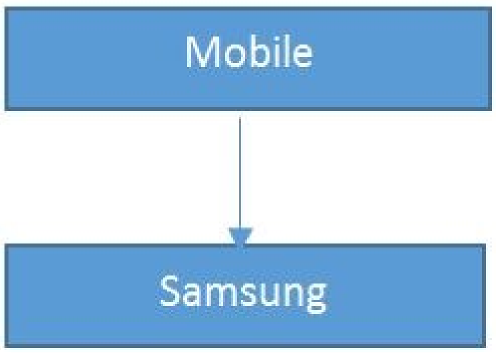
Considering the example, the above figure 1.1 itself shows what is inheritance. Basic Mobile functionality is to send a message, dial and receive a call. So the brands of mobile is using this basic functionality by extending the mobile class functionality and adding their own new features to their respective brand.

There are mainly 4 types of inheritance,

1. Single level inheritance
2. Multi-level inheritance
3. Hierarchical inheritance
4. Hybrid inheritance
5. Multiple inheritance

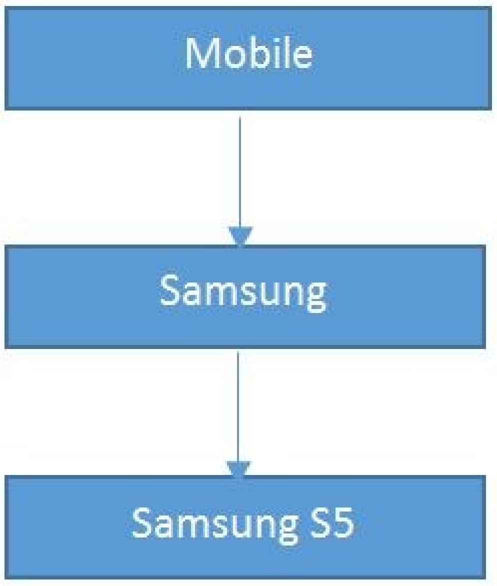
Single level inheritance

In Single level inheritance, there is single base class & a single derived class i.e. - A base mobile features is extended by Samsung brand.



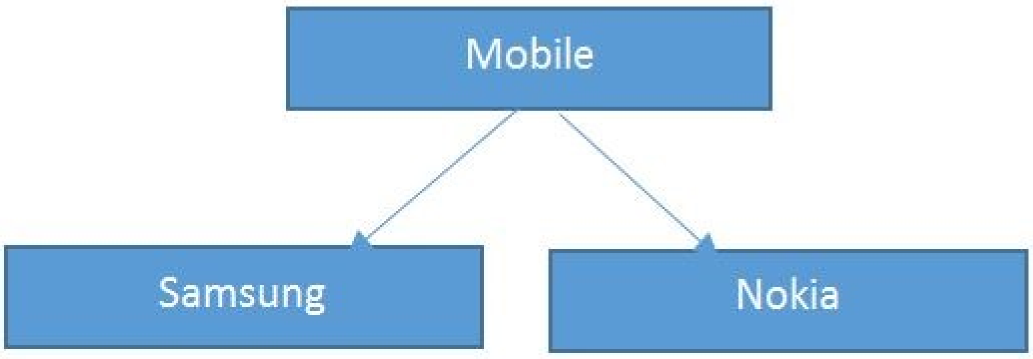
Multilevel inheritance

In Multilevel inheritance, there is more than one single level of derivation. i.e. - After base features are extended by Samsung brand. Now Samsung brand has manufactured its new model with new added features or advanced OS like Android OS, v4.4.2 (kitkat). From generalization, getting into more specification.



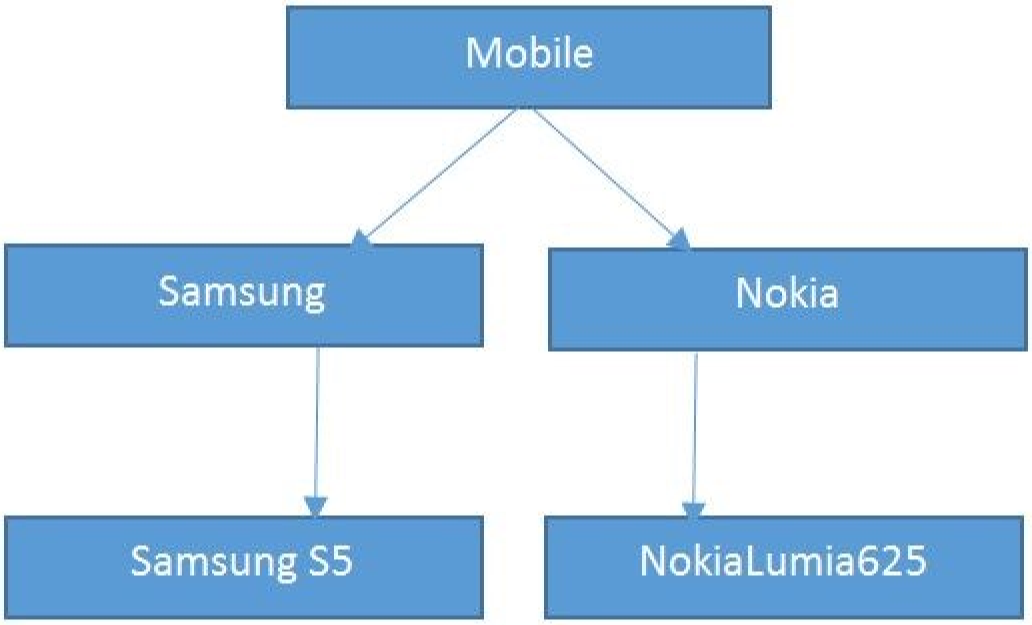
Hierarchal inheritance

In this type of inheritance, multiple derived class would be extended from base class, it's similar to single level inheritance but this time along with Samsung, Nokia is also taking part in inheritance.



Hybrid inheritance

Single, Multilevel, & hierarchal inheritance all together construct a hybrid inheritance.



```
public class Mobile
{
        //Properties
        //Methods
}

public class Samsumg : Mobile
{
        //Properties
              //Methods
     }

     public class Nokia : Mobile
     {
         //Properties
        //Methods
     }
```

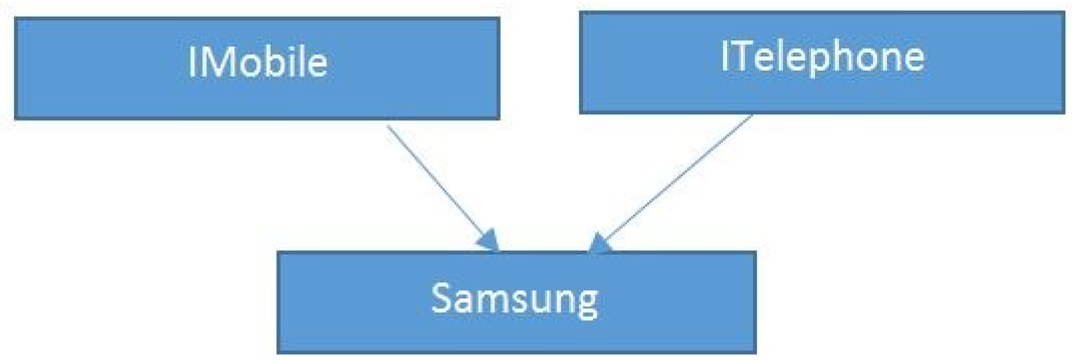
Interface

Multiple inheritance where derived class will extend from multiple base classes.
Samsung will use the function of multiple Phone (Mobile & Telephone). This would create a confusion for complier to understand which function to be called when any event in mobile is triggered like Dial () where Dial is available in both the Phone i.e. - (Mobile & Telephone). To avoid this confusion C# came with the concept of interface which is different from multiple inheritance actually.

If we take an interface it is similar to a class but without implementation & only declaration of properties, methods, delegates & events. Interface actually enforces the class to have a standard contract to provide all implementation to the interface members. Then what's is the use of interface when they do not have any implementation? Answer is, they are helpful for having readymade contracts, only we need to implement functionality over this contract.

I mean to say, Dial would remain Dial in case of Mobile or Telephone. It won't be fair if we give different name when its task is to Call the person.

Interface is defined with the keyword 'interface' .All properties & methods with in the interface should be implemented if it is been used. That's the rule of interface.



```
interface IMobile
{
   Void Dial();
}

interface ITelephone
{
   void Dial();

      }

      public class Mobile : IMobile, ITelephone

      {
          public void Dial()
          {
                  Console.WriteLine("Dial a number");
          }
      }
```

Effect Analysis of New Teaching Method

In order to evaluate the teaching effect of the case based teaching method in the flipped classroom, we distributed questionnaires to the students majoring in software engineering and computer science and technology in the class. A total of 60 students participated in the questionnaire and the results of the questionnaire are shown in the figure2. Most of the students are satisfied with this new teaching method. 85% of the students are satisfied with the new teaching method. The main reason why students are satisfied with this teaching method is that they can participate in the classroom by themselves, which not only gives full play to students' initiative, but also arouses their thirst for knowledge. Thus the students' ability to solve problems has been improved.

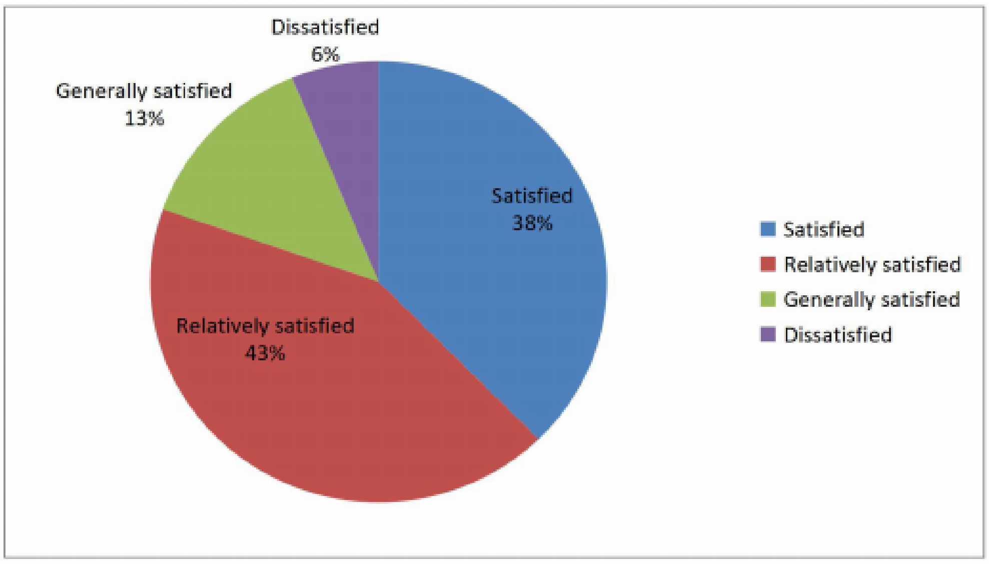


Figure 2. Students' satisfaction with teaching methods

Conclusion

After the ideas of the flipped classroom and case teaching are applied to C++ teaching practice, it can effectively improve students' practical ability in C++. Flipped classroom video allows students to learn and review at any time.

The case teaching and the teacher's targeted explanation in class enable students to quickly grasp the key points of knowledge. At the same time, because the case comes from a real project, students will have the ability to develop real projects after mastering the case.

It is proved that after the two teaching methods are integrated together, students' ability to master knowledge has been greatly improved. In the next step, we want to allow students to participate more in the design of the project, and strive to design more practical and interesting projects.

References

[1] Wang Quanrui. Practical Research on the Progressive Flipping Teaching Mode of C++ Programming[J]. Journal of Henan Colledge of Science and Technology,2020,40(08):59-64.

[2] Zhang Fan, Meng Yu, Chang Shuhui. Research on online-offline computer professional curriculum blended teaching under the background of industry-education integration——Taking "Object-oriented System Analysis and Design" as an example[J]. Industry and Technology Forum, 2020,19(14):150-151.

[3] Pouria Pourmand,Binaya Pudasaini,Mohsen Shahandashti. Assessing the Benefits of Flipped Classroom in Enhancing Construction Students' Technical Communication Skills[J]. Journal of Civil Engineering Education,2021,147(1).

[4] Mustafa Cevikbas, Gabriele Kaiser. Flipped classroom as a reform-oriented approach to teaching mathematics[J]. ZDM,2020,52.