Atal Bihari Vajpayee

Indian Institute of Information Technology

And Management (ABV-IIITM), Gwalior

विश्वजीवनामृतं ज्ञानम्

IIITM Gwalior

Project Report

Topic: Learning OOPS concepts using OOPL

By - 2020BCS-013 - Anurag Yadav
2020BCS-015 - Aryan Dadhich
2020BCS-016 - Ashish Singh
2020BCS-017 - Ashwin Kumar Singh
2020BCS-018 - Bachute Sarvesh Vikas

INDEX

Abstract

The project “Learning OOPS with OOPL” has been made by using almost all the basic to advanced concepts of OOP. The program of this project is written by considering efficient techniques of implementing the different classes, functions and all other concepts.

On the implementation of the code, the user is asked to choose an option i.e. options are the concepts, after choosing it, again few options are given to the user like Definition, Code, Working, Types. This helps the user to study the particular concept according to his/her interest. In this way, users can get theoretical knowledge about all the concepts.

One of the most important options given to the user is for showing and implementing the entire selected concept, by choosing it, he/she is asked for some inputs and some selected operations are performed on them. Output of this is shown by using these concepts and everything about its implementation is explained at the end.In this way, the user can learn, study and test his/her knowledge about any concept of OOPL and this will help him/her to make the hands strong in this course.

Introduction

We have tried to use the concepts of OOP to let the user interact with our program to learn something new about OOP. The Program offers various choices to the users to learn from like encapsulation, abstraction, operator overloading, inheritance and polymorphism. Users can choose one of the topics and from these topics users will be given options to choose between the definition, code, example and its types. This interactive program will surely help the user to learn some new concepts or revise some other concepts. Also, after all of these ,if a user wants to test his/her knowledge of the concepts learned there is a quiz ready for the user, which will test the user for the concepts learned.

Description

Our Project for this course is not just about using the OOPL concepts and implementing them in various parts of code, but also it is about creating an interface which will be interactive with the user and also all these concepts and implementation will be used for giving outputs which are theoretical representation of them.

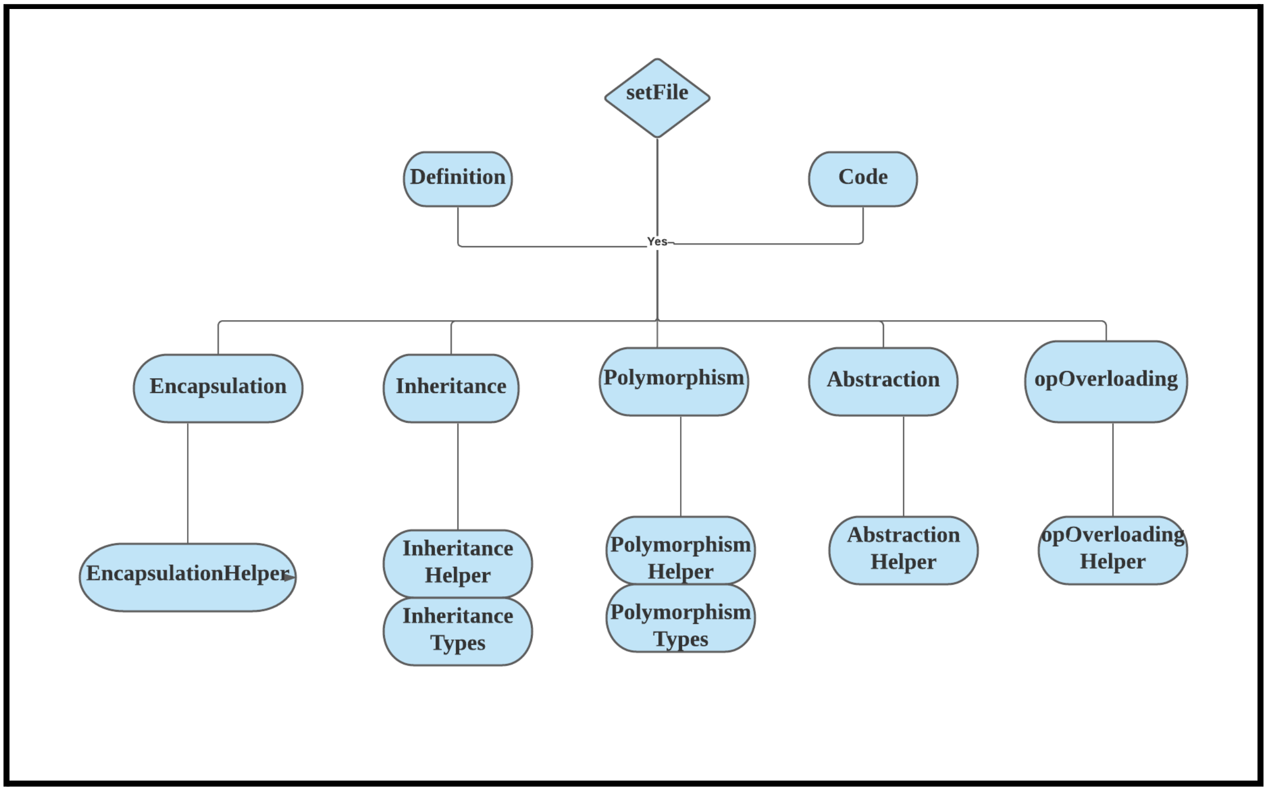
Also, the program covers almost all the basic concepts like classes,functions, encapsulation, abstraction, etc to the advanced concepts like inheritance and types, virtual functions and classes, access modifiers, etc. This helps us in complete implementation as well as in understanding the important shorthand techniques to make the code more efficient. The program is asking the user for choosing an option i.e. the concept to study. Again some options are to be chosen by the user to study that topic more specifically like its Definition, Code, Working or Types.

Along with this user can also try his/her hands on these concepts by choosing respective helper class i.e. one of the given options to give the required inputs and after some operations, output will be displayed by using that concept, this will help the user in understanding the concept even much better.

One of the options is “Quiz”. This again asks the user to choose any concept and will create a quiz. User can answer the questions and instantly he/she can know the right or wrong answer. At the end, final score will be calculated and with the help of this user can assess his/her knowledge on selected concepts.

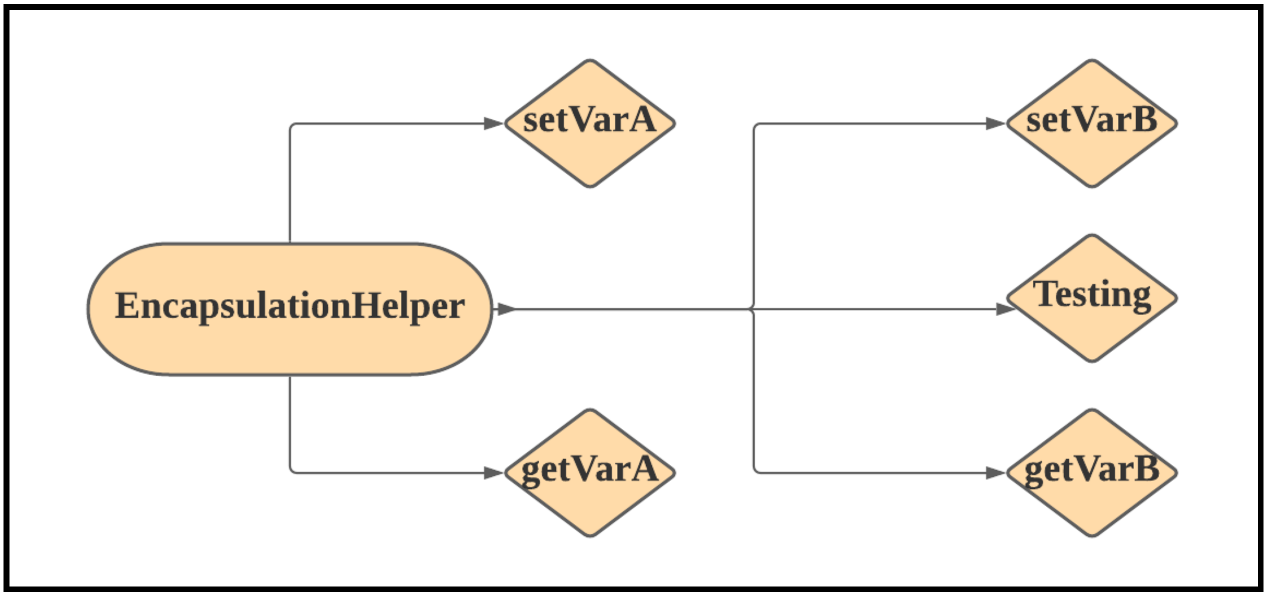
In this way, the user can learn, study and test his/her knowledge about any concept of OOPL and this will help him/her to make the hands strong in this course.

Classes Formed

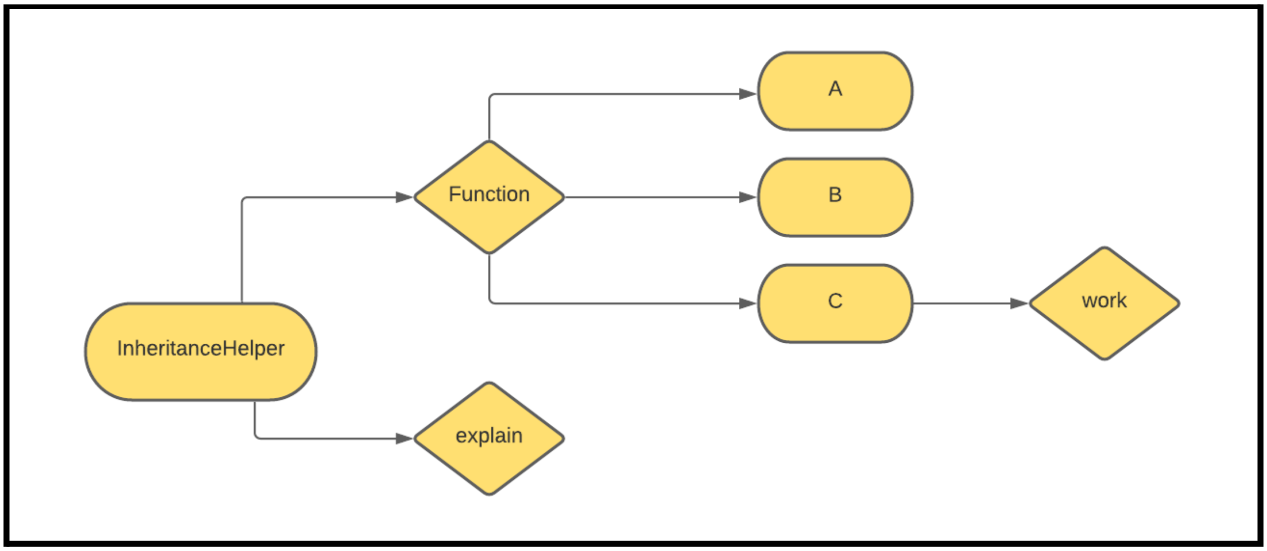


(Tree representation of all the classes and function used in the program)

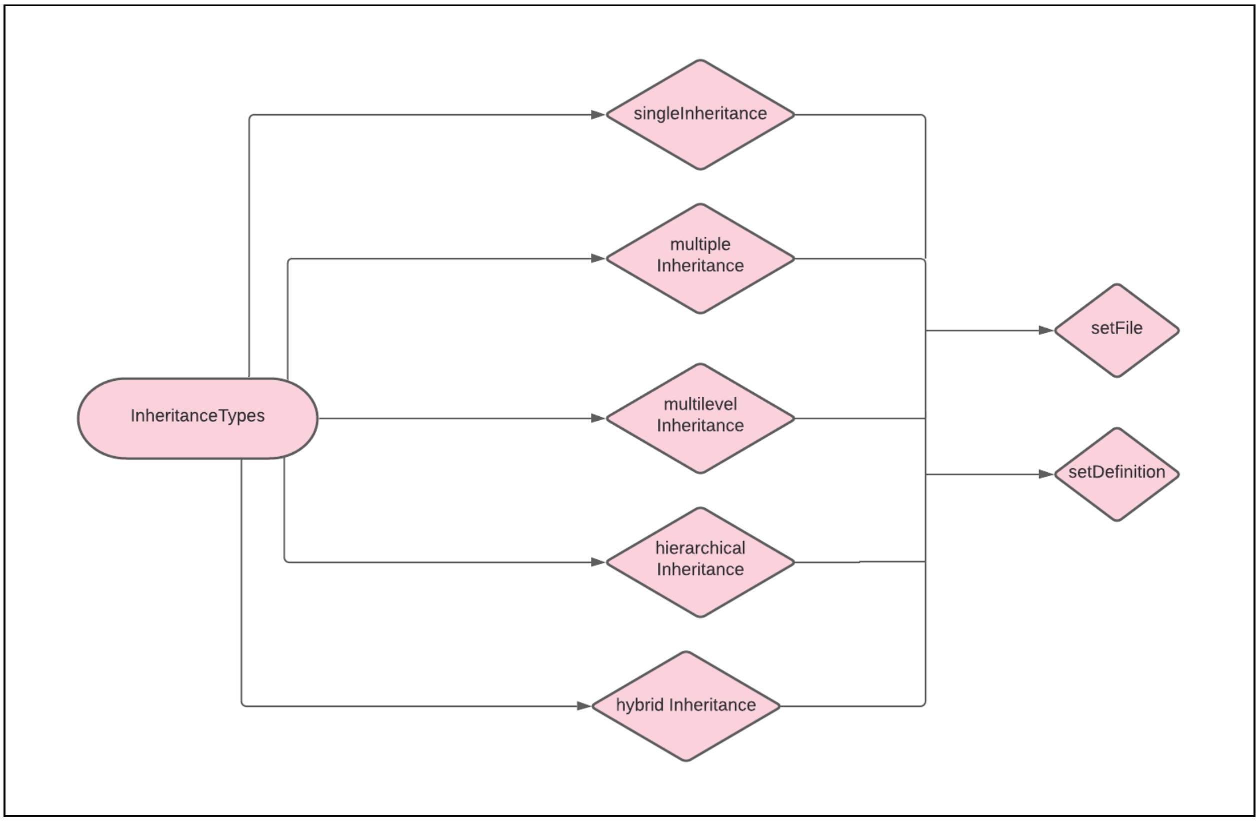
EncapsulationHelper Class:



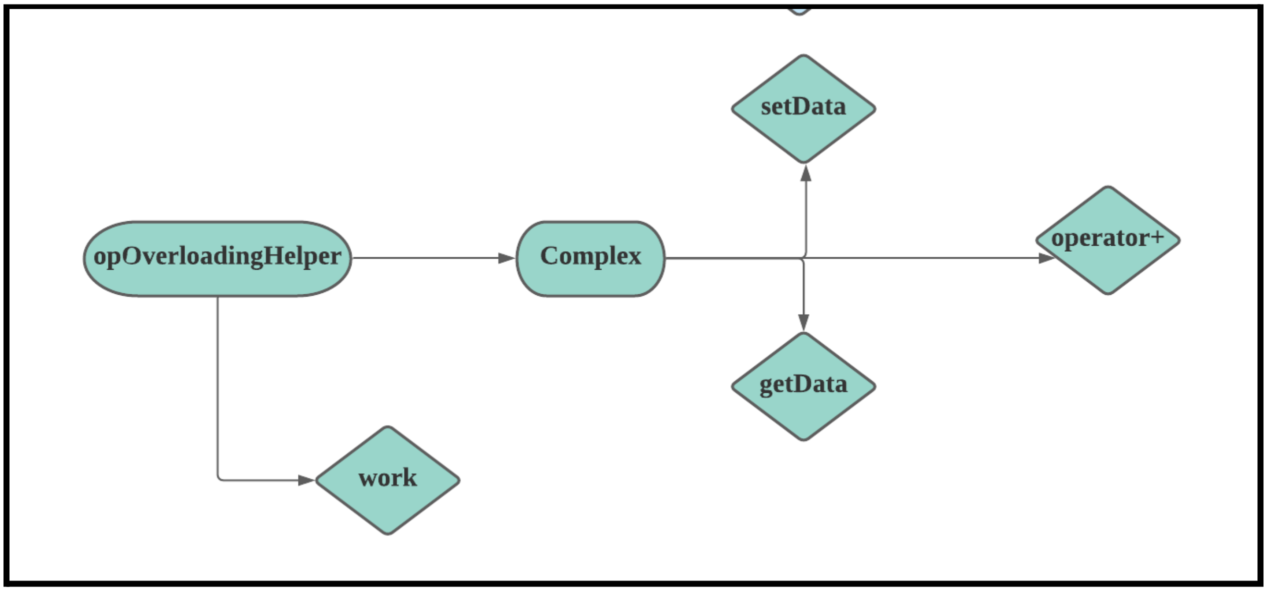
InheritanceHelper Class:



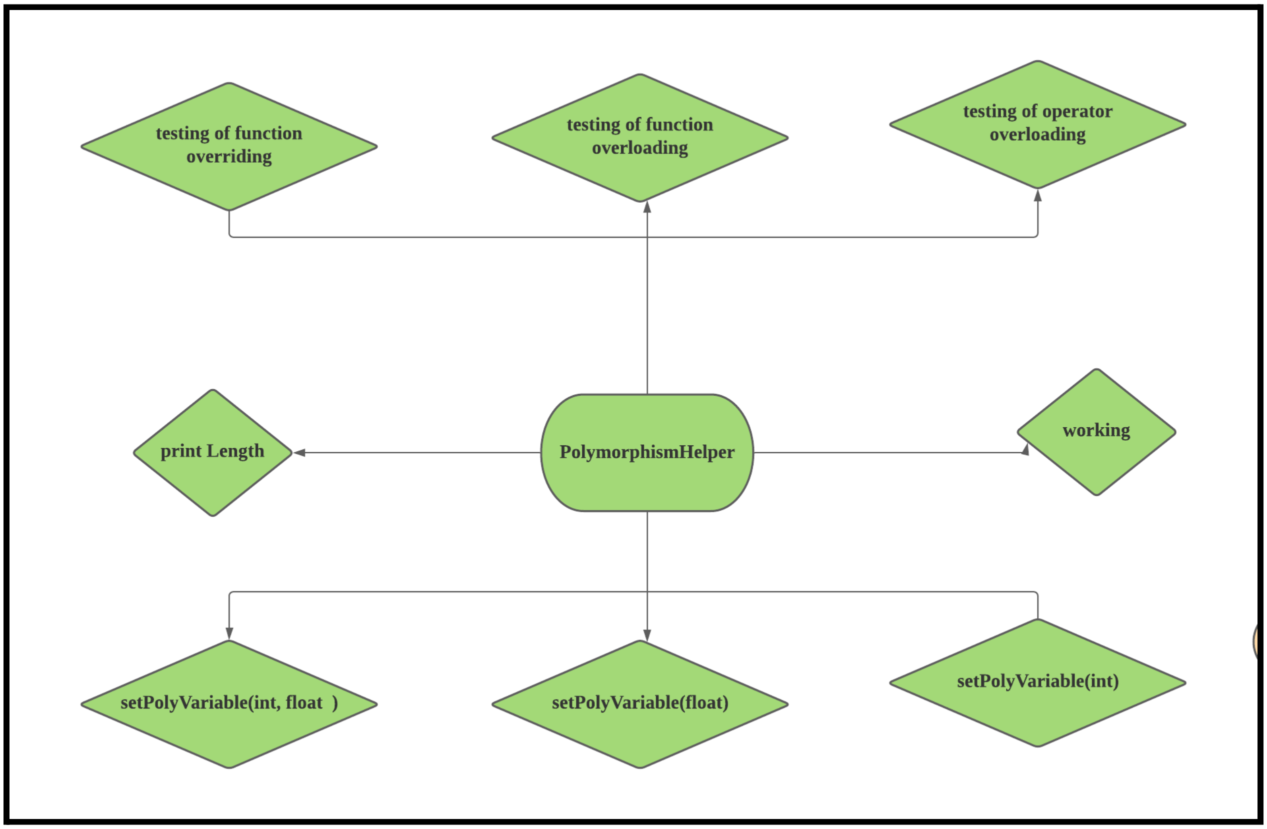
InheritanceTypes Class:



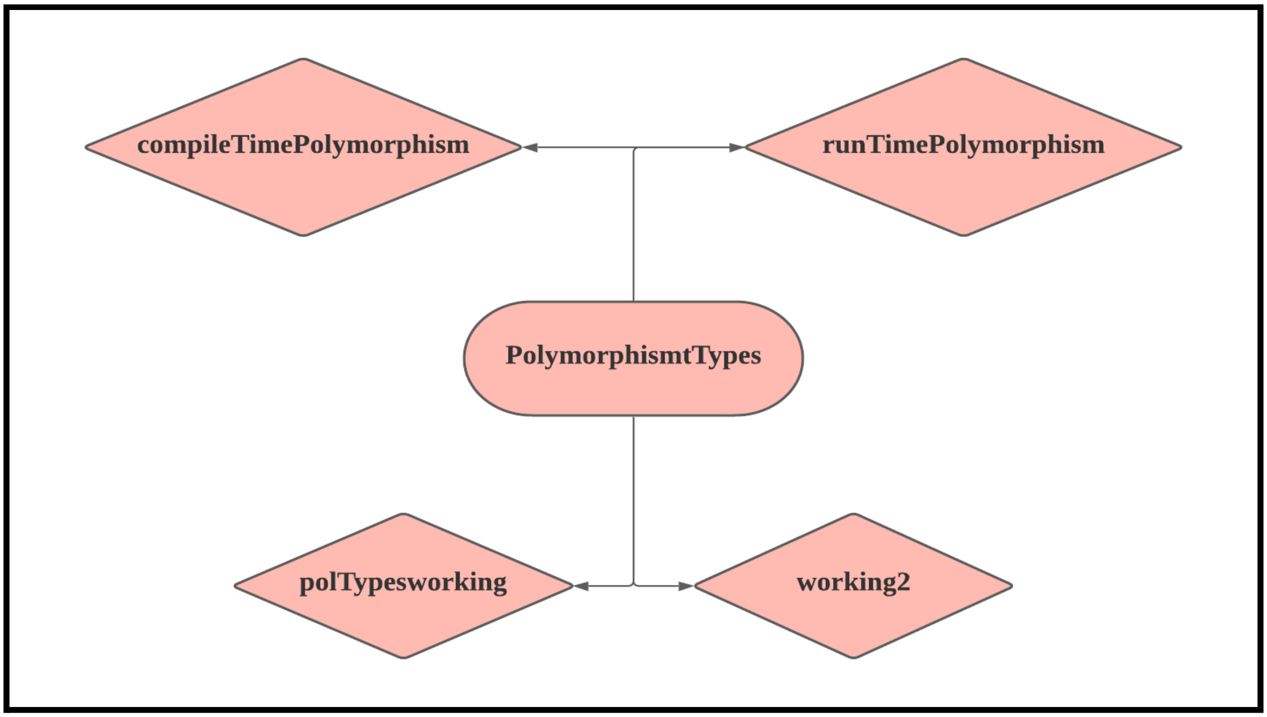
opOverloading Class:



PolymorphismHelper Class:

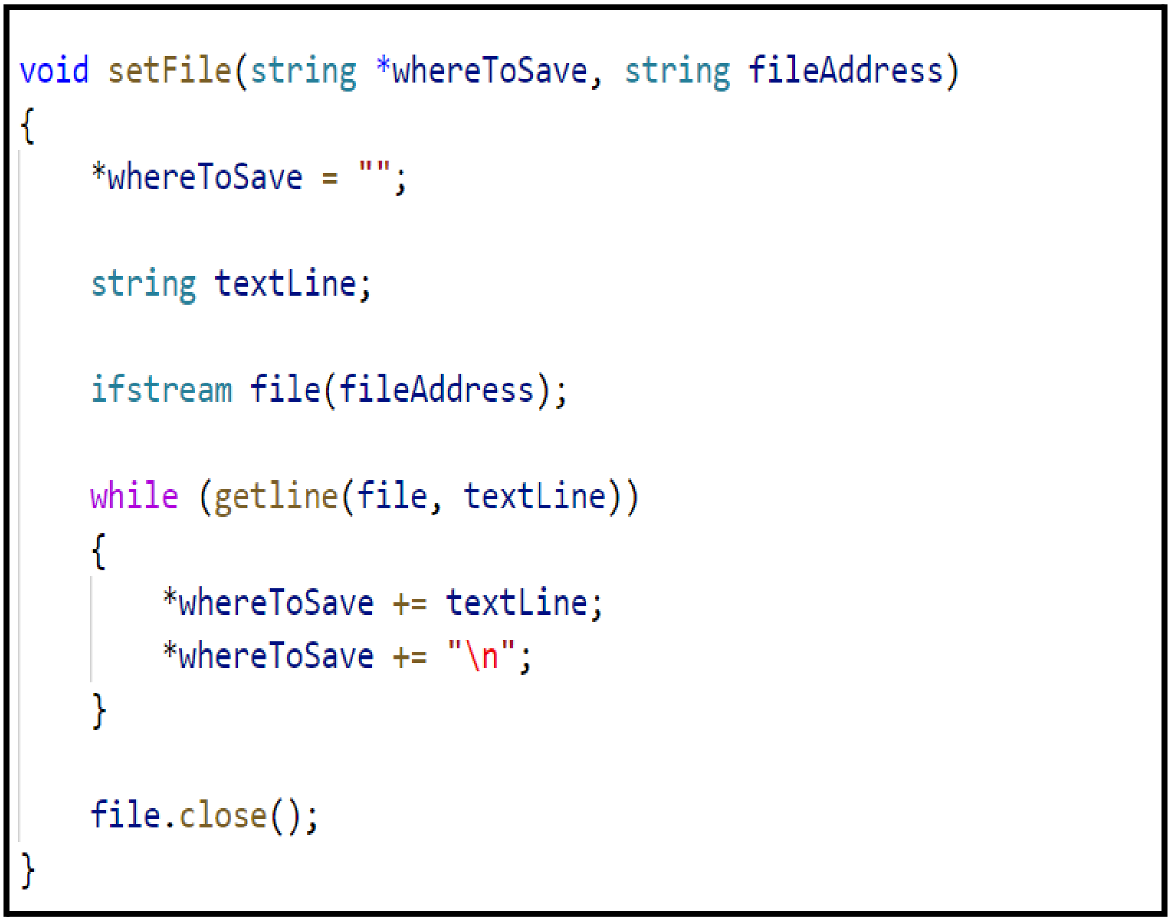

testing of function overriding
testing of function overloading
testing of operator overloading
print Length
PolymorphismHelper
working
setPolyVariable(int, float)
setPolyVariable(float)
setPolyVariable(int)

PolymorphismTypes Class:



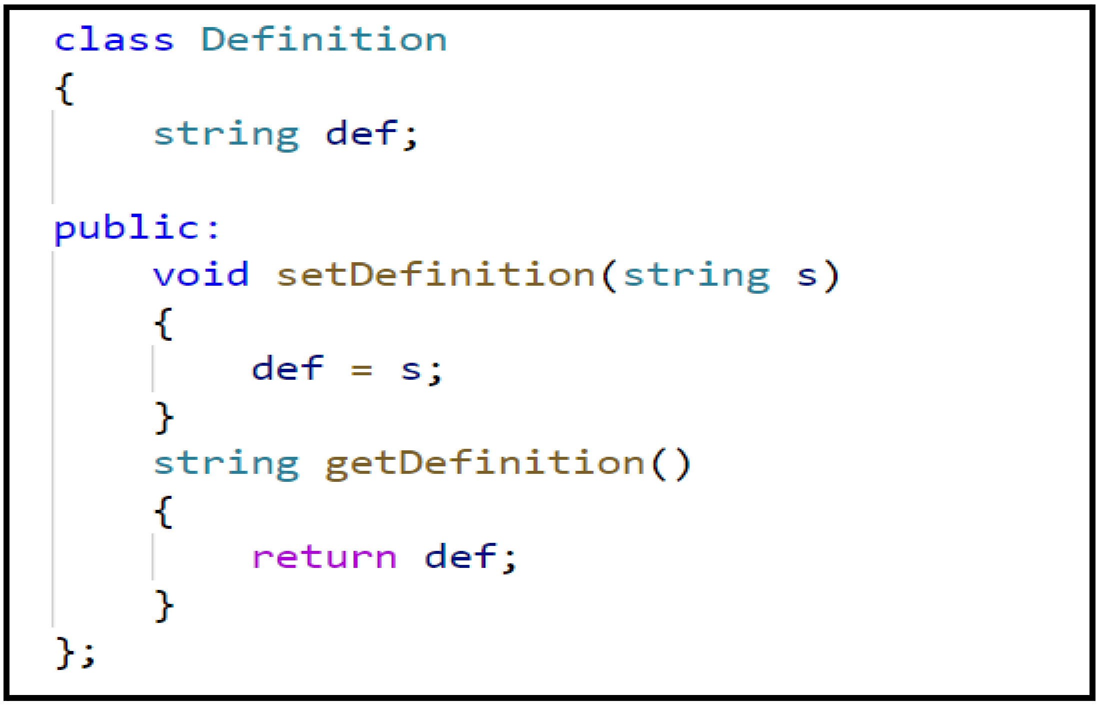
Commonly used Functions and Classes

- **Set File:** This function is used throughout the code to save the files in the corresponding variables.



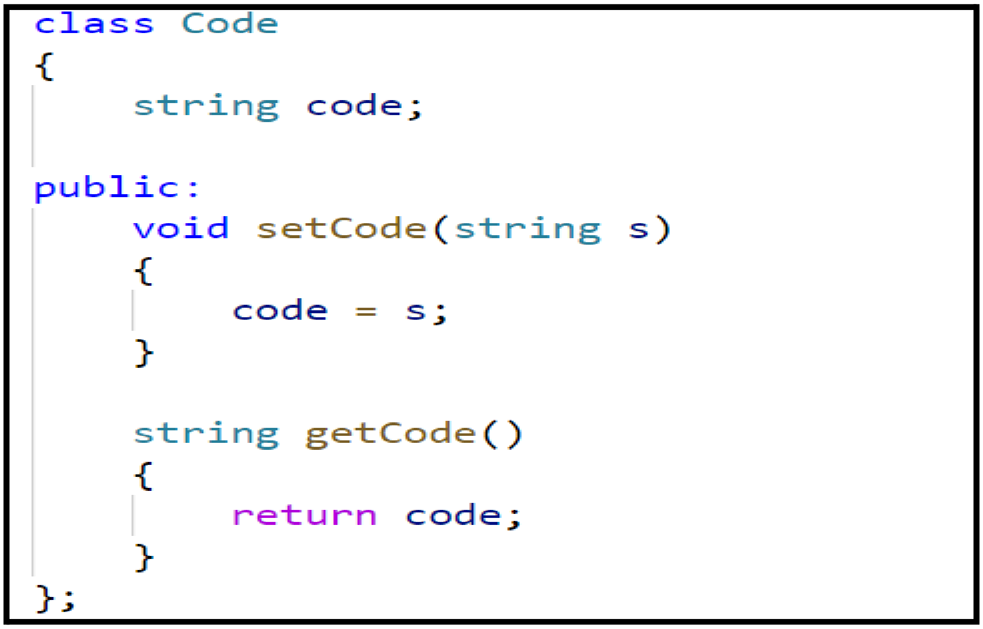
```
void setFile(string *whereToSave, string fileAddress)
{
    *whereToSave = "";

    string textLine;

    ifstream file(fileAddress);

    while (getline(file, textLine))
    {
        *whereToSave += textLine;
        *whereToSave += "\n";
    }

    file.close();
}
```

- **Definition Class:** This function is used to set definition to all the Classes (polymorphism, encapsulation, inheritance, abstraction, and others) and it returns the definition in the form of string.



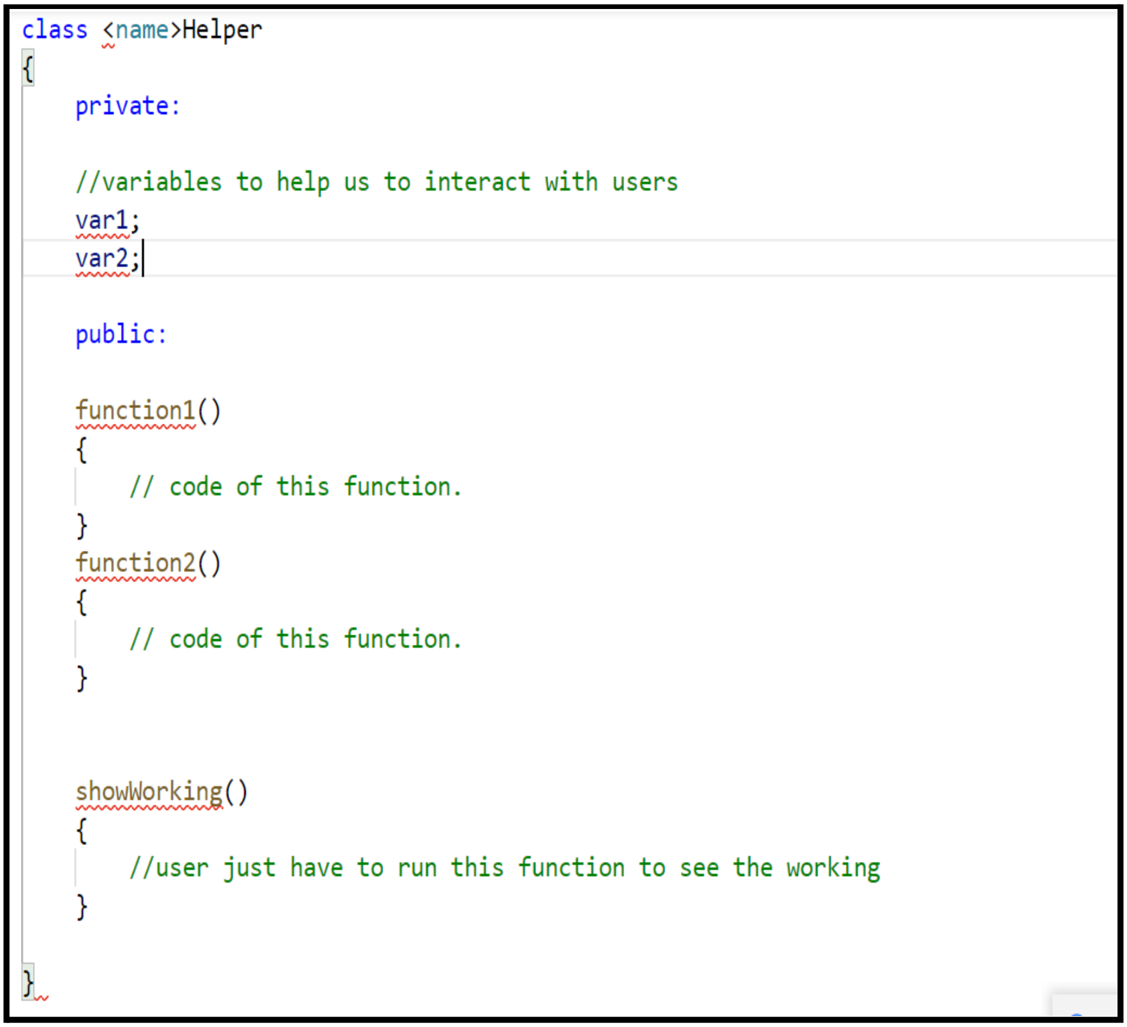
```
class Definition
{
    string def;

public:
    void setDefinition(string s)
    {
        def = s;
    }
    string getDefinition()
    {
        return def;
    }
};
```

- **Code Class:** This function is used to set Code for all the Classes (polymorphism, encapsulation, inheritance, abstraction and others) and it returns the code in the form of string.



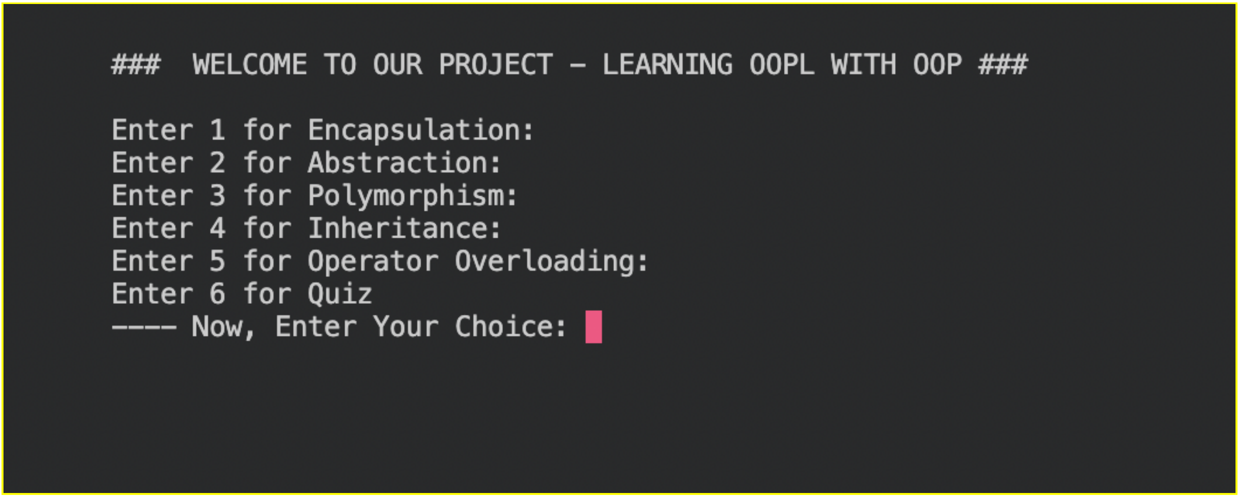
```
class Code
{
    string code;

public:
    void setCode(string s)
    {
        code = s;
    }

    string getCode()
    {
        return code;
    }
};
```

- **Helper Class:** The <PillarName>Helper is the class for interacting with the user to take the input as well as to show the working of the particular Class.

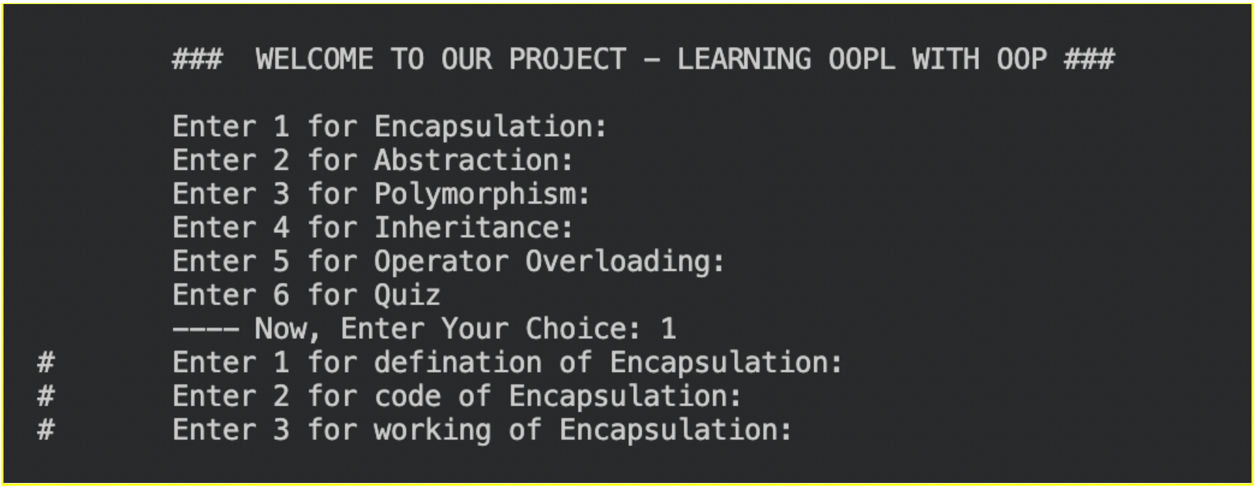


```
class <name>Helper
{
    private:

    //variables to help us to interact with users
    var1;
    var2;

    public:

    function1()
    {
        // code of this function.
    }
    function2()
    {
        // code of this function.
    }



    showWorking()
    {
        //user just have to run this function to see the working
    }


}
```

Output of the Program

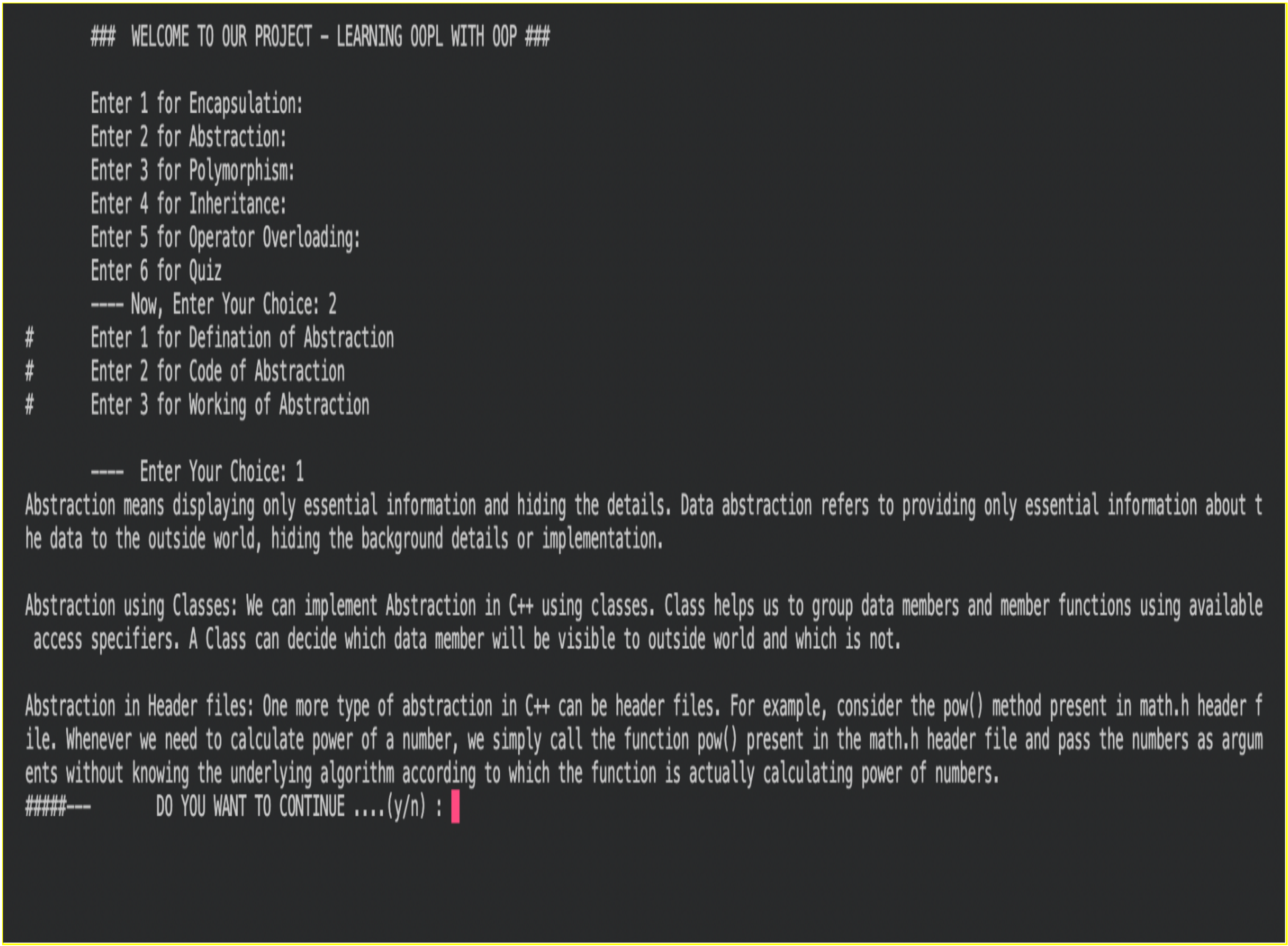


```
###  WELCOME TO OUR PROJECT - LEARNING OOPL WITH OOP ###

Enter 1 for Encapsulation:
Enter 2 for Abstraction:
Enter 3 for Polymorphism:
Enter 4 for Inheritance:
Enter 5 for Operator Overloading:
Enter 6 for Quiz
---- Now, Enter Your Choice: 
```



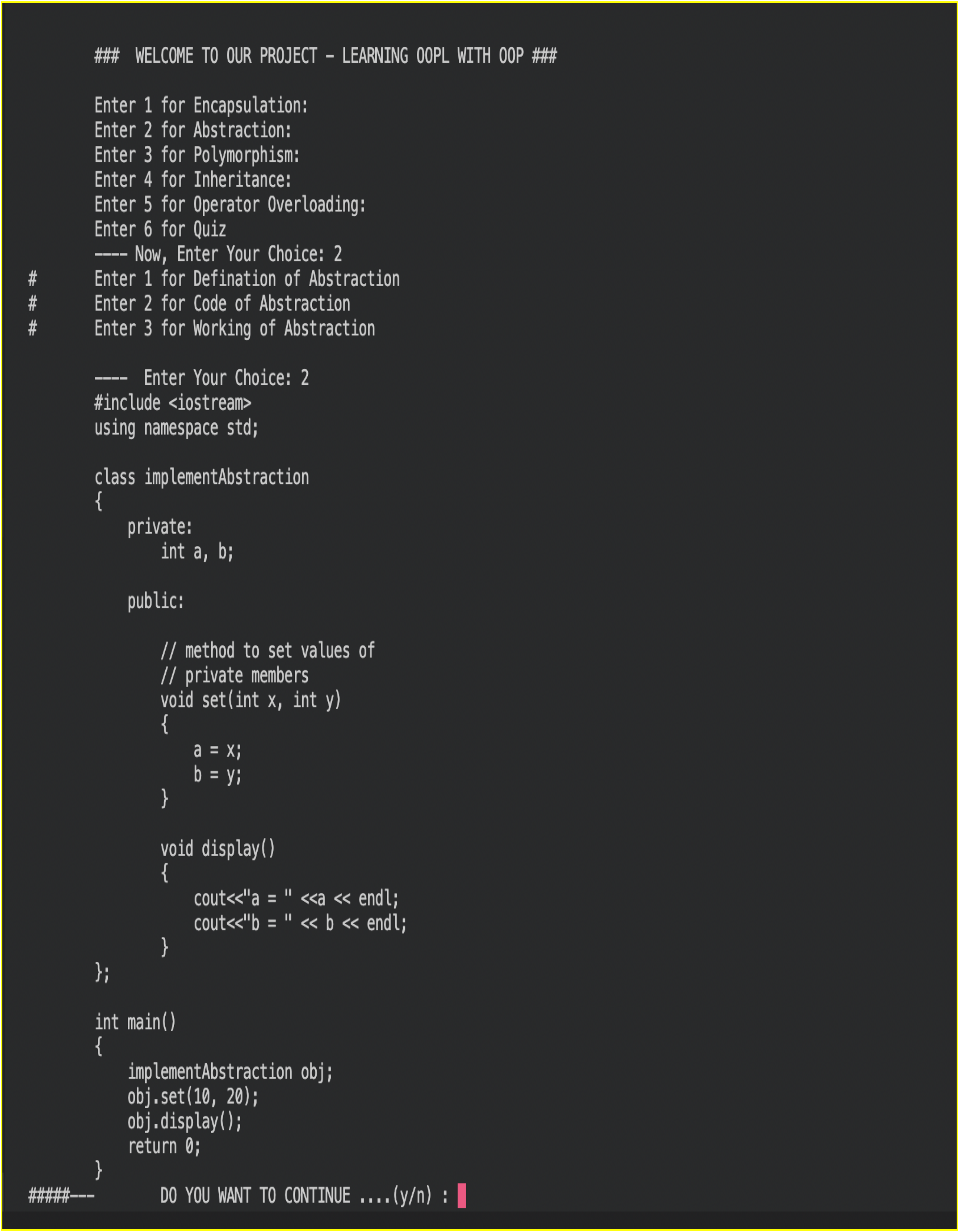
```
        ###  WELCOME TO OUR PROJECT - LEARNING OOPL WITH OOP ###

        Enter 1 for Encapsulation:
        Enter 2 for Abstraction:
        Enter 3 for Polymorphism:
        Enter 4 for Inheritance:
        Enter 5 for Operator Overloading:
        Enter 6 for Quiz
        ---- Now, Enter Your Choice: 1
#       Enter 1 for defination of Encapsulation:
#       Enter 2 for code of Encapsulation:
#       Enter 3 for working of Encapsulation:
```

(Users are asked to choose options.....)

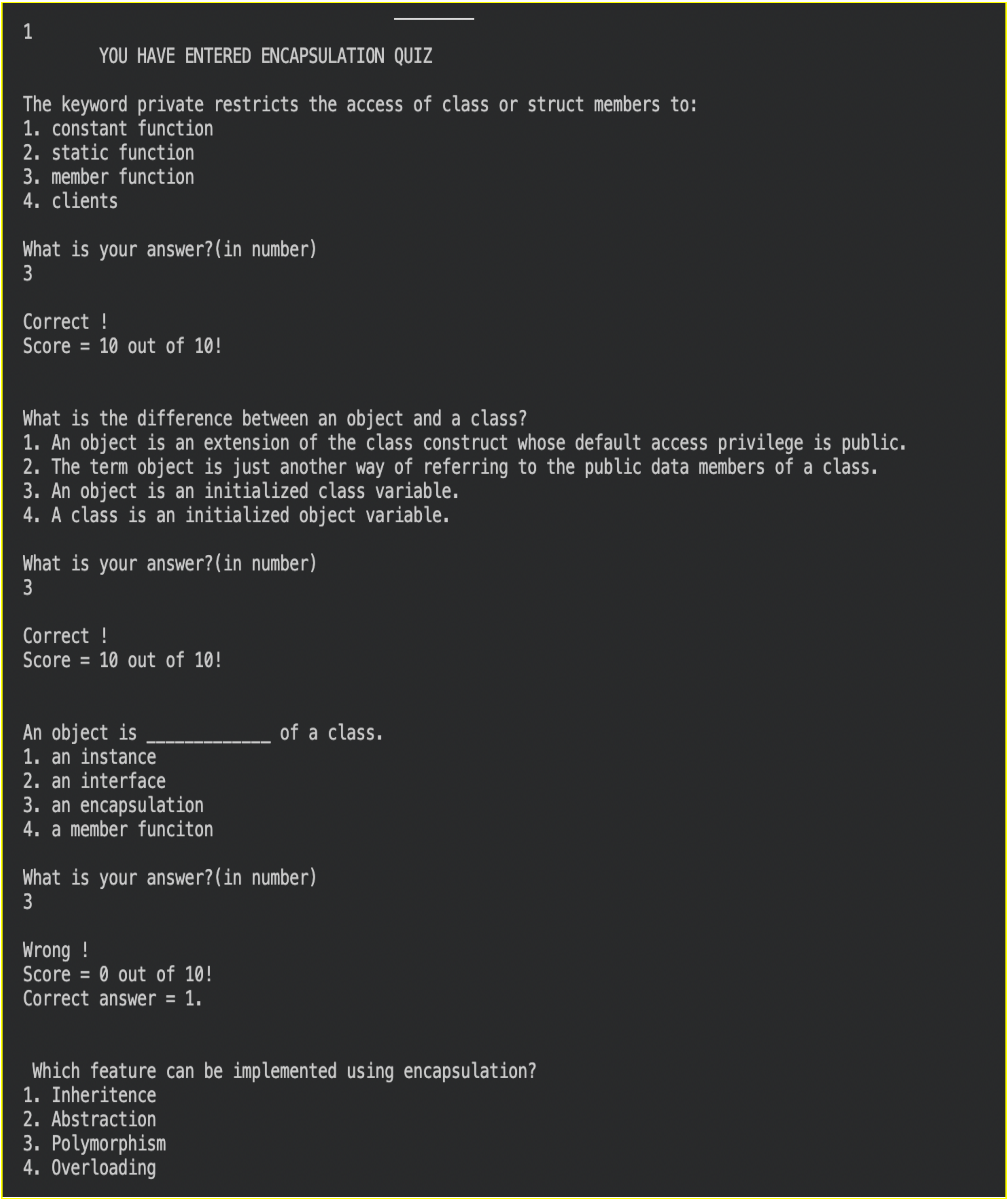


```
       ##  WELCOME TO OUR PROJECT - LEARNING OOPL WITH OOP ###

       Enter 1 for Encapsulation:
       Enter 2 for Abstraction:
       Enter 3 for Polymorphism:
       Enter 4 for Inheritance:
       Enter 5 for Operator Overloading:
       Enter 6 for Quiz
       ---- Now, Enter Your Choice: 2
#      Enter 1 for Defination of Abstraction
#      Enter 2 for Code of Abstraction
#      Enter 3 for Working of Abstraction

       ----  Enter Your Choice: 1
Abstraction means displaying only essential information and hiding the details. Data abstraction refers to providing only essential information about t
he data to the outside world, hiding the background details or implementation.

Abstraction using Classes: We can implement Abstraction in C++ using classes. Class helps us to group data members and member functions using available
 access specifiers. A Class can decide which data member will be visible to outside world and which is not.

Abstraction in Header files: One more type of abstraction in C++ can be header files. For example, consider the pow() method present in math.h header f
ile. Whenever we need to calculate power of a number, we simply call the function pow() present in the math.h header file and pass the numbers as argum
ents without knowing the underlying algorithm according to which the function is actually calculating power of numbers.
#####---       DO YOU WANT TO CONTINUE ....(y/n) :
```

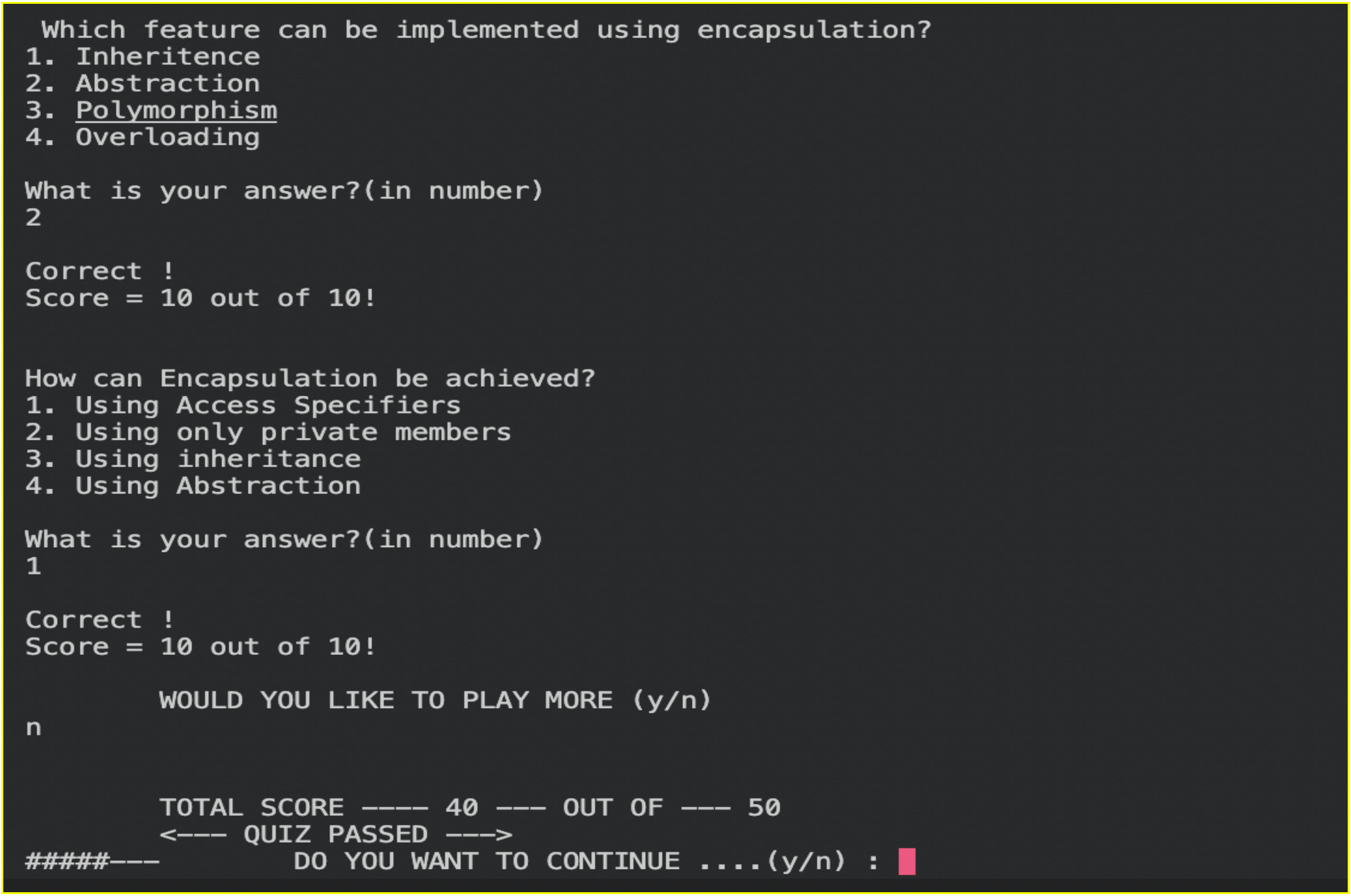
(After choosing a particular topic and subtopic output is shown...)



```
        ###  WELCOME TO OUR PROJECT - LEARNING OOPL WITH OOP ###

        Enter 1 for Encapsulation:
        Enter 2 for Abstraction:
        Enter 3 for Polymorphism:
        Enter 4 for Inheritance:
        Enter 5 for Operator Overloading:
        Enter 6 for Quiz
        ---- Now, Enter Your Choice: 2
#       Enter 1 for Defination of Abstraction
#       Enter 2 for Code of Abstraction
#       Enter 3 for Working of Abstraction

        ----  Enter Your Choice: 2
        #include <iostream>
        using namespace std;

        class implementAbstraction
        {
            private:
                int a, b;

            public:

                // method to set values of
                // private members
                void set(int x, int y)
                {
                    a = x;
                    b = y;
                }

                void display()
                {
                    cout<<"a = " <<a << endl;
                    cout<<"b = " << b << endl;
                }
        };

        int main()
        {
            implementAbstraction obj;
            obj.set(10, 20);
            obj.display();
            return 0;
        }
#####---        DO YOU WANT TO CONTINUE ....(y/n) :
```



```
1
        YOU HAVE ENTERED ENCAPSULATION QUIZ

The keyword private restricts the access of class or struct members to:
1. constant function
2. static function
3. member function
4. clients

What is your answer?(in number)
3

Correct !
Score = 10 out of 10!


What is the difference between an object and a class?
1. An object is an extension of the class construct whose default access privilege is public.
2. The term object is just another way of referring to the public data members of a class.
3. An object is an initialized class variable.
4. A class is an initialized object variable.

What is your answer?(in number)
3

Correct !
Score = 10 out of 10!


An object is _____________ of a class.
1. an instance
2. an interface
3. an encapsulation
4. a member funciton

What is your answer?(in number)
3

Wrong !
Score = 0 out of 10!
Correct answer = 1.


 Which feature can be implemented using encapsulation?
1. Inheritence
2. Abstraction
3. Polymorphism
4. Overloading
```



```
 Which feature can be implemented using encapsulation?
1. Inheritence
2. Abstraction
3. Polymorphism
4. Overloading

What is your answer?(in number)
2

Correct !
Score = 10 out of 10!


How can Encapsulation be achieved?
1. Using Access Specifiers
2. Using only private members
3. Using inheritance
4. Using Abstraction

What is your answer?(in number)
1

Correct !
Score = 10 out of 10!

        WOULD YOU LIKE TO PLAY MORE (y/n)
n


        TOTAL SCORE ---- 40 --- OUT OF --- 50
        <--- QUIZ PASSED --->
#####---        DO YOU WANT TO CONTINUE ....(y/n) :
```

Conclusion

- All the OOPL concepts are implemented using efficient techniques.
- Theoretical as well as practical knowledge and implementation of concepts can be understood.
- Codes for different classes and functions are written separately for better understanding.
- Various classes and functions are used for reducing the complexity and number of lines of code.