

What is the main function of object oriented programming?

- In my opinion, the main function of OOP is that: binding data and functions that operate on them together.
- Second, solving the real world problems and, making models for them.

Explain how to create a C++ model for a real world problem.

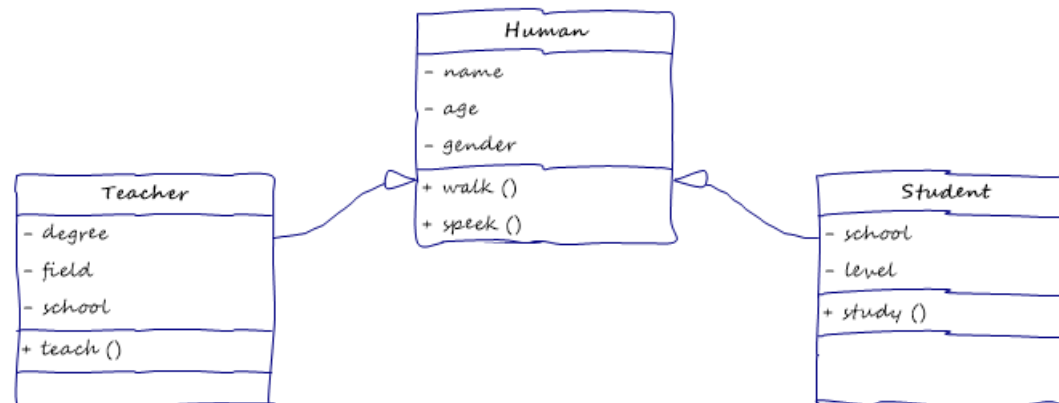
- First, we have to define out objects.
- Second, we have to define properties and functionalities of objects.
- Finally, we have to find the relationships between objects.

To explain this, let me give you an example on real world problem modeling, human model

First, we use "Generalization" approach in modeling process.

So, we can make classes in Hierarchical order

Human can be normal person, student or teacher, etc.



Which sections of a class can a member function of that class access?

- Private and Public sections

Explain how to access private data of a class using objects of the same class.

- We can't access private data from objects of the same class direct(using dot(.) operator)
- But, we can use friend function to access them
- Example:

```
Base.cpp

class Base{
private:
    int _value;
public:
    // Constructors
    Base() : _value(0){}
    Base(const int& value) : _value(value){}

    // Setters
    void value(const int& value){_value = value;}

    // Getters
    int value() const {return _value;}

    // overloading - access private data of the same class
    friend int sum(const Base& B1, const Base& B2);

    // display
    void display() const {printf("Base(value = %i)\n", _value);}

    // Destructor
    ~Base(){}
};

int sum(const Base& B1, const Base& B2){return B1._value + B2._value;}
```

```
Base.cpp

#include <iostream>
using namespace std;
#define endl "\n"

// run
int main(){
    // create objects
    Base B1(10), B2(20);
    Base B3(sum(B1, B2));

    // display
    B1.display();
    B2.display();
    B3.display();
    return 0;
}

/* output
PS GU\OOP\LAB\Assignments\#9\code> c++ main.cpp -o main.exe; .\main.exe
Base(value = 10)
Base(value = 20)
Base(value = 20)
*/
```

What is executed automatically when the control reaches the end of the class scope?

- A special method called "Destructor".

Which class members can be accessed only from inside the class?

- Private Data Members, in private section.

What is the return type of a destructor function?

- Void, it returns nothing, as I mentioned: Special Method.

Explain how to access member functions of a class.

- Using dot operator after objects of the same class.

Which feature of OOPS described the reusability of code?

- Inheritance, as the most feature, and Overloading.

What is meant by an object in C++?

- Everything we can describe it in the world is an object.
- Object in OOP is an instance of Blue-print(class)
- Object Contain, Combine methods and data.

What is the term used to indicate variables and constants of a class?

- Data members, data placed in private section.

Which class members can be accessed from outside the class?

- Public members, in public section.

What is the term used to indicate methods of a class?

- Using dot operator, like this: *classObject.methodName(params)*;

How can friend functions be invoked?

- The friend function isn't member function,
- so we can use it as normal like this: *functionName(params)*;

Does your code satisfy data-hiding? How?

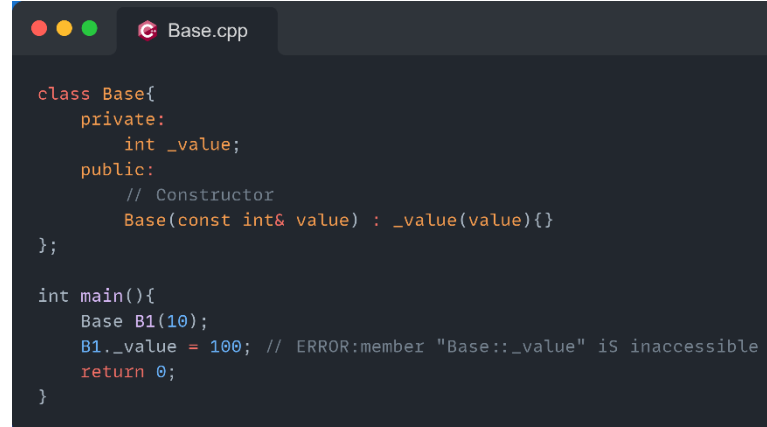
- Anyway, we say that the code satisfy data-hiding when it's data are private:
- As, it can't be accessed from outside the class except using interface (member function).
- Example: Code doesn't satisfy data-hiding:



```
class Base{
public:
    // data
    int _value;

    // Constructor
    Base(const int& value) : _value(value){}
};

int main(){
    Base B1(10);
    B1._value = 100; // No-Data-Hiding So, This is bad-practice.
    return 0;
}
```



```
class Base{
private:
    int _value;
public:
    // Constructor
    Base(const int& value) : _value(value){}
};

int main(){
    Base B1(10);
    B1._value = 100; // ERROR:member "Base::_value" is inaccessible
    return 0;
}
```

Is there encapsulation in your code? How?

- We say: there is encapsulation in code, when the data and it's operations are in the same container (combined together).

Explain how to make your code reusable.

- First, we have code design it in using OOP approach.
- Second, divide it into objects.
- Every object should contain its functionality.
- Every function should do a specific task.
- Don't repeat your-self, use inheritance, overloading and polymorphism to apply this.

What is the main function of a constructor?

- Initializing new object of a class and its attributes.

What are the different types of constructors?

- Default constructor, Parametrized Constructor and Copy constructor.

What is the main function of a destructor?

- Destroying current object. When the life time of an object ended it free the memory used by it.

Which constructor type that has an empty parameters list?

- Default constructor.

What is the main function of a friend function?

- Access private data of and object from outside the class.

Explain how to access class data members using a friend function.

- I can't explain by text, but I can explain by example:

```

class Base{
private:
    int _value;
public:
    // Constructors
    Base() : _value(0){}
    Base(const int& value) : _value(value){}

    // overloading - access private data of the same class
    friend int sum(const Base& B1, const Base& B2);

    // display
    void display() const {printf("Base(value = %i)\n", _value);}
};

// Accessing private data: B1._value, B2._value
int sum(const Base& B1, const Base& B2){return B1._value + B2._value;}
  
```

```

int main(){
    Base B1(10), B2(20);
    Base B3(sum(B1, B2));

    // display
    B1.display();
    B2.display();
    B3.display();
    return 0;
}

/* output
PS G:\OOP\LAB\Assignments\B9\code> c++ Base.cpp -o main.exe; .\main.exe
Base(value = 10)
Base(value = 20)
Base(value = 20)
*/
  
```

What is meant by const. member function?

- This function is constant, as it can't modify member data.

Show how to define const. member functions.

- Syntax: *returnType* *functionName*(params) *const* {yourcode here};

What is the main function of a static data member?

- We can define static-data-members as a shared data between all objects of the same class.
- I mean, when a class created; static data initialized.
- Static member data can be accessed from any object of the same class. So, it shared.

Which member function can access const. objects?

- Const. member functions.

What is the type of polymorphism of operator overloading?

- Compile time polymorphism.

What is meant by binding in C++?

- It's an important term in C++ specially OOP;
- It combines data and functions together. In one block, as I see.

What are the different names of static binding?

- Compile-time binding or early binding.

What is the main function of operator overloading?

- It used to specialize the functionality of operator to the new objects.
- The language can't define operator for user-defined classes.

What are the different names of dynamic binding?

- **Runtime polymorphism** or **late binding**.

Does operator overloading change associativity?

- Operator overloading does not change the associativity of operators.

What is the difference between the notations of prefix and postfix of unary operators?

- Prefix: It applies function then return value.
- Postfix: It Returns value then apply function.
- Example:
- `int value = 10;`
- `cout << ++value << endl;`      this will increment value then return it's value, output = 11; and value = 11;
- `cout << value++ << endl;`      this will return value then apply increment, output = 11; but value = 12;

Does operator overloading change normal precedence?

- Operator overloading does not change the normal precedence of operators.

What are the only operators that can be overloaded?

- `+, -, *, /, %, +=, -=, *=, /=, %=, ++ (pre), -- (pre), (post) ++, (post) --, !, ~, ==, !=, >, <, >=, <=, =`

What is the difference between unary and binary operators?

- Unary operators have only one operand, but binary operators have two operands.



Explain how overloaded binary operators work.

- By example:

```

class Base{
private:
    int _value;
public:
    // Constructors
    Base() : _value(0){}
    Base(const int& value) : _value(value){}

    // overloading (+) operator
    int operator+(const Base& secondOperand) const {
        // current object is the firstOperand
        return this->_value + secondOperand._value;
    }

    // display
    void display() const {printf("Base(value = %i)\n", _value);}
};

```

```

#include <iostream>
using namespace std;
#define endl "\n"

int main(){
    Base B1(10), B2(20);
    Base B3(B1+B2);

    // display
    B1.display(); B2.display(); B3.display();
    return 0;
}

/* output
PS GU\OOP\LAB\Assignments\#9\code> c++ main.cpp -o main.exe; .\main.exe
Base(value = 10)
Base(value = 20)
Base(value = 30)
*/

```

What is the difference between function overriding and function overloading?

- In my point-of-view,
- Overriding, make new function have the same name and signature (parameters, return-type) but, different implementation.
- Overloading, create new function with the same name and different signature.

What is the main function of UML class diagram?

- Help developers and Non-specialists to understand the model.

What does UML class diagram contain?

- Classes, The main thing in the class diagram that contain other related things.
- Attributes, Private Data for every class in the diagram.
- Methods, Public functions give functionality to the class.
- Access Specifiers, Private, Public and Protected specifiers.
- Relationships, Inheritance and relationships between base class and derived class.

What are the different types of inheritance?

- Single inheritance.
- Multilevel inheritance.
- Multiple inheritance.
- Combination of two type. I mean Hybrid inheritance.
- Also, الشكل الهرمي, can be multilevel.

Explain the main function of inheritance.

- Make relationship between two classes.
- Also, help developers to reuse code. (Don't repeat yourself)
- Because it get the properties and methods of the base class to derived class.

What are the different modes of inheritance?

- Private, Public and Protected.

Show how to define the class inheritance mode.

- Code: `class ClassName: accessSpecifier BaseClass{your code here};`

What is meant by class visibility?

- Who can see (access) class, this handled with access specifiers: private, public and protected.

Show how to define the class visibility.

- *DrivenClassName Visibility BaseClassName{your code here};*

What is the type of polymorphism that virtual functions achieve?

- When we refer to a derived class object using a pointer or reference to the base class, we can call a virtual function for that object and execute the derived class's version of the function.

What is the difference between virtual function and pure virtual function?

- Pure virtual function has a declaration in base class without any implementation.
- Virtual function implemented in base class.

Show how to define virtual functions and pure virtual functions.

- Virtual : *virtual returnType functionName(params){your code her};*
- Pure : *virtual returnType functionName(params) = 0;*

What is the main function of virtual functions?

- Virtual functions allow dynamic dispatch based on the object's type.

What is meant by abstract class?

- An abstract class contains at least one pure virtual function.

What is the type of polymorphism that member functions achieve?

- The static polymorphism.

Show how virtual functions can be accessed.

- Code: *drievdClassPtr* → *functionName(params)*;

What is the main function of abstract classes?

- The main function of abstract classes is to serve as blueprints for other classes.

Which class that can't be used to declare objects of its own?

- Abstract class, as it contain at least pure virtual function.

What is the difference between member functions and virtual functions?

- Virtual functions are runtime polymorphism.
- Member functions are compile time polymorphism.

Thanks!