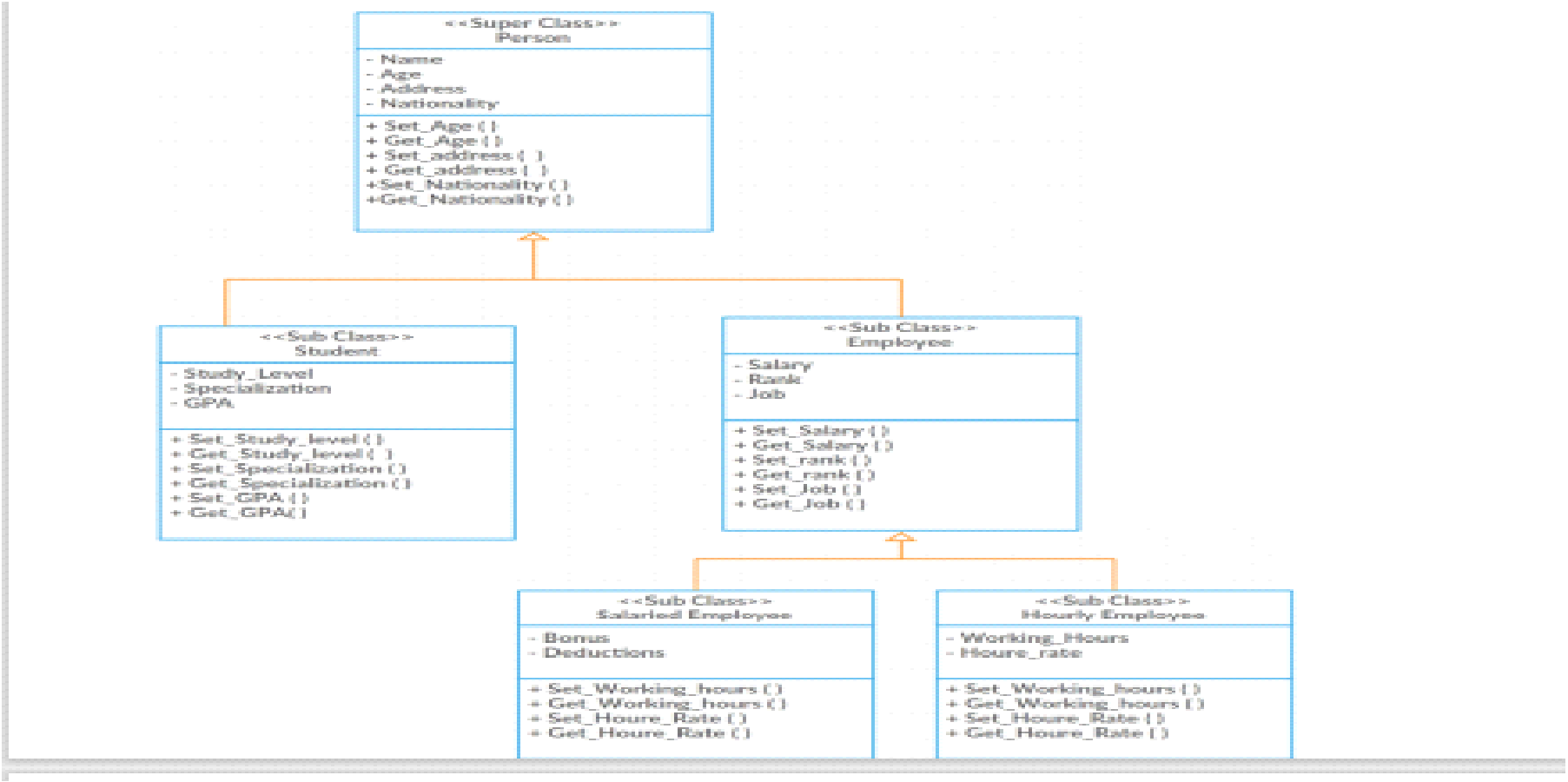
object-oriented programming

(OOP)

KIAN _ ACADEMY


<<Super Class>>
Person
- Name
- Age
- Address
- Nationality
+ Set_Age ( )
+ Get_Age ( )
+ Set_address ( )
+ Get_address ( )
+Set_Nationality ( )
+Get_Nationality ( )
<<Sub Class>>
Student
- Study_Level
- Specialization
- GPA
+ Set_Study_level ( )
+ Get_Study_level ( )
+ Set_Specialization ( )
+ Get_Specialization ( )
+ Set_GPA ( )
+ Get_GPA( )
<<Sub Class>>
Employee
- Salary
- Rank
- Job
+ Set_Salary ( )
+ Get_Salary ( )
+ Set_rank ( )
+ Get_rank ( )
+ Set_Job ( )
+ Get_Job ( )
<<Sub Class>>
Salaried Employee
- Bonus
- Deductions
+ Set_Working_hours ( )
+ Get_Working_hours ( )
+ Set_Houre_Rate ( )
+ Get_Houre_Rate ( )
<<Sub Class>>
Hourly Employee
- Working_Hours
- Houre_rate
+ Set_Working_hours ( )
+ Get_Working_hours ( )
+ Set_Houre_Rate ( )
+ Get_Houre_Rate ( )

#Modes of inheritance

#Public mode

If we derive a child class from a public parent class. Then the public member of the parent class becomes a public member for the child class and protected members of parent class becomes protected members of the child class.

#Protected mode

If we derive child class from a protected base class, then the public, as well as a protected member of the parent class, becomes the protected members of the child class.

#Private mode

If we derive a child class from a private base class, then the public, as well as protected members, become private for the derived class.

Private members of a base class cannot be directly accessed in the derived class in any circumstance.

Example

```
#include <iostream>
using namespace std;
class Super
{
private:
   int x;
   void setX(int x1)
   {
      x=x1;
   }
public:
   int y;
   void setY(int y1)
   {
      y=y1;
   }
protected:
   int z;
   void setZ(int z1)
   {
      z=z1;
   }
};
class Sub: public Super
{
private:
   int a;
   void setA(int a1)
   {
      a=a1;
   }
public:
   int b;
   void setB(int b1)
   {
      b=b1;
   }
protected:
   int c;
   void setC(int c1)
   {
      c=c1;
   }
};
```

Order of Constructor Call with Inheritance in C++

- Whether derived class's default constructor is called or parameterized is called, base class's default constructor is always called inside them.

- To call base class's parameterized constructor inside derived class's parameterized constructor, we must mention it explicitly while declaring derived class's parameterized constructor

Function Overriding :

It is the redefinition of base class function in its derived class with same signature

Example

```
#include <iostream>
using namespace std;
class BaseClass
{
public:
    void disp()
    {
        cout<<"Function of Parent Class";
    }
};
class DerivedClass: public BaseClass
{
public:
    void disp()
    {
        cout<<"Function of Child Class";
    }
};
int main()
{
    DerivedClass obj;
    obj.disp();
    return 0;
}
```

Output:

Function of Child Class

Note

In function overriding , the function in parent class is called the overridden function .

and function in child class is called overriding function.

Example

```
#include <iostream>
using namespace std;
class BaseClass
{
public:
    void disp()
    {
        cout<<"Function of Parent Class";
    }
};
class DerivedClass: public BaseClass
{
public:
    void disp()
    {
        cout<<"Function of Child Class";
    }
};
int main()
{
    BaseClass obj;
    obj.disp();
    return 0;
}
```

Output:

Function of Parent Class

Note

If you want to call the Overridden function from overriding function then you can do it like this:

Object (sub class) . parent_class_name :: function_name

Example

```
#include <iostream>
using namespace std;
class BaseClass
{
public:
    void disp()
    {
        cout<<"Function of Parent Class";
    }
};
```

```
class DerivedClass: public BaseClass
{
public:
    void disp()
    {
        cout<<"Function of Child Class";
    }
};
```

```
int main()
{
    DerivedClass obj;
    obj.BaseClass::disp();
    return 0;
}
```

Output:

Function of Parent Class

Exam questions:

- **(1)** Which among the following best describes the Inheritance?

- A) Copying the code already written
- B) Using the code already written once
- C) Using already defined functions in programming language
- D) Using the data and functions into derived segment

- **(2)** Which among the following best defines single level inheritance?

- A) A class inheriting a derived class
- B) A class inheriting a base class
- C) A class inheriting a nested class
- D) A class which gets inherited by 2 classes

- **(3)** Which programming language doesn’t support multiple inheritance?

- A) C++ and Java
- B) C and C++
- C) Java and SmallTalk
- D) Java

- **(4)** Which among the following is correct for a hierarchical inheritance?

- A) Two base classes can be used to be derived into one single class
- B) Two or more classes can be derived into one class
- C) One base class can be derived into other two derived classes or more
- D) One base class can be derived into only 2 classes

- **(5)** Which access type data gets derived as private member in derived class?

- A) Private
- B) Public
- C) Protected
- D) Protected and Private

- **(6)** If a base class is inherited in protected access mode then which among the following is true?

- A) Public and Protected members of base class becomes protected members of derived class
- B) Only protected members become protected members of derived class
- C) Private, Protected and Public all members of base, become private of derived class
- D) Only private members of base, become private of derived class

- **(7)** Members which are not intended to be inherited are declared as _____

- A) Public members
- B) Protected members
- C) Private members
- D) Private or Protected members

- **(8)** While inheriting a class, if no access mode is specified, then which among the following is true? (in C++)

- A) It gets inherited publicly by default
- B) It gets inherited protected by default
- C) It gets inherited privately by default
- D) It is not possible

- **(9)** If a derived class object is created, which constructor is called first?

- A) Base class constructor
- B) Derived class constructor
- C) Depends on how we call the object
- D) Not possible

- **(10)** The private members of the base class are visible in derived class but are not accessible directly.

- A) True
- B) False

- **(11)** How can you make the private members inheritable?

- A) By making their visibility mode as public only
- B) By making their visibility mode as protected only
- C) By making their visibility mode as private in derived class
- D) It can be done both by making the visibility mode public or protected

Answer the questions :-

- 1 - d
- 2 - b
- 3 - d
- 4 - c
- 5 - a
- 6 - a
- 7 - c
- 8 - c
- 9 - a
- 10- a
- 11- d