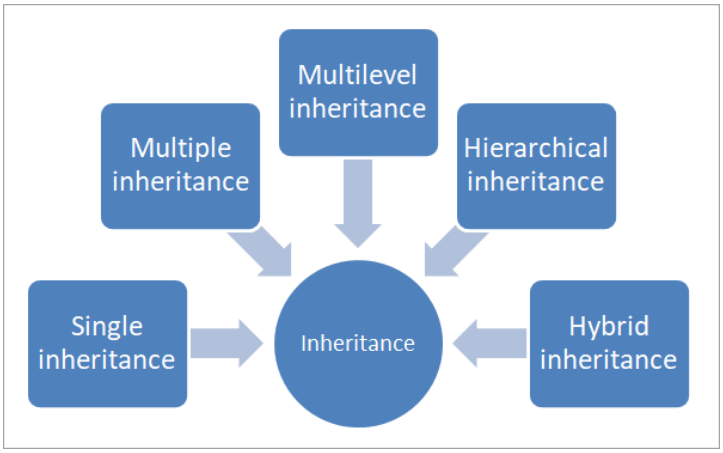
GROUP 4

Names:

1. Nafisa Abdelaziz
2. Hossam Mohamed
3. Bassant Selima
4. Ashraf Abdulkhaliq

What is Inheritance?

Inheritance is one of four pillars of Object-Oriented Programming (OOPs), it's a feature in which new classes are created from the existing classes (derived class), the derived class inherits all the properties of the base class, without changing the properties of base class and may add its own new features to class. There are mainly five types of Inheritance in C++ that you will explore in this report. They are as follows:

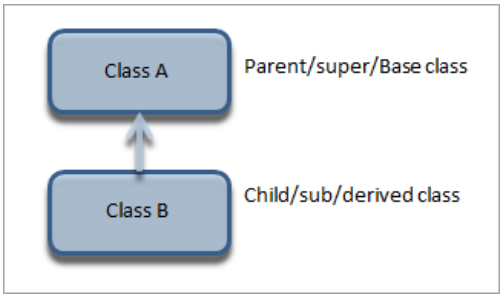


(Types of Inheritance)

Types of Inheritance:

1. Single Inheritance:

Is the most primitive among all the types of inheritance in C++. In this inheritance, a single class (child) inherits the properties from only base class (parent).

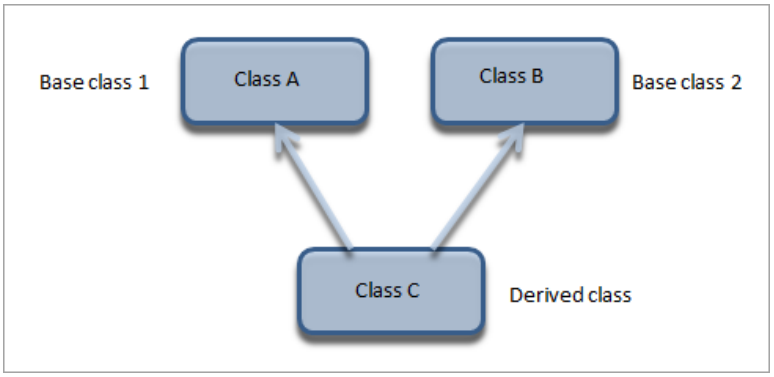


Syntax:

```
class Class_A {
        //class specific code;
 };
class Class_B : public Class_A {
        //class specific code;
 };
```

2. Multiple Inheritance:

Is a child class inheriting from more than one class, and the data members of all the base classes are accessed by the derived or child class according to the access specifiers.

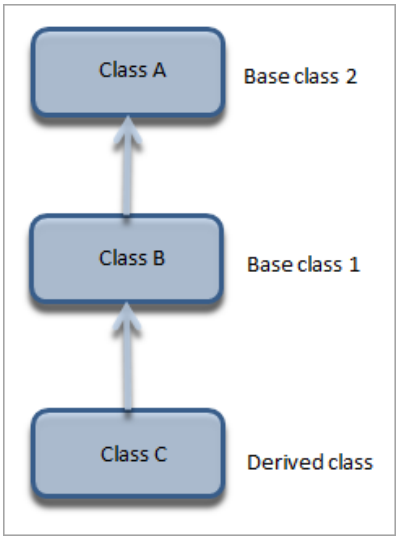


Syntax:

```
class Class_A {
                //class specific code;
 };
class Class_B {
                //class specific code;
 };
class Class_C : public Class_A, public Class_B {
                //class specific code;
 };
```

3. Multilevel Inheritance:

Is a child class is created from another child class. Inheritance is transitive so the last derived class acquires all the members of all its base classes.

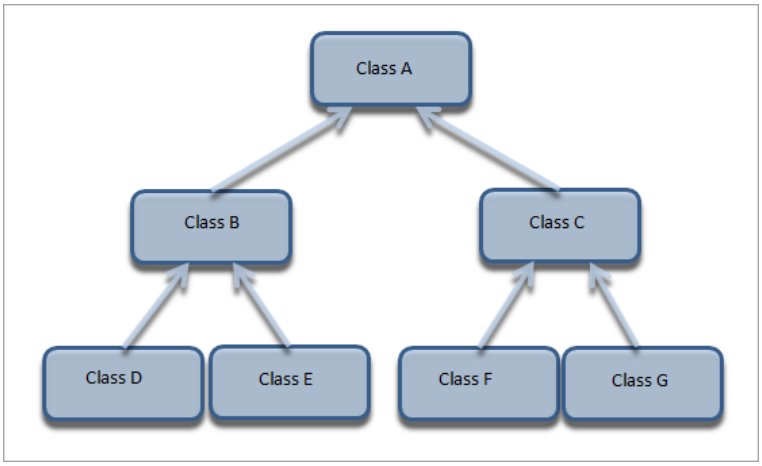


Syntax:

```
class Class_A {
                //class specific code;
 };
class Class_B public Class_A {
                //class specific code;
 };
class Class_C : public Class_B {
                //class specific code;
 };
```

4. Hierarchical Inheritance:

Is a single base class inheriting multiple derived classes. This inheritance has a tree-like structure since every class acts as a base class for one or more child classes.

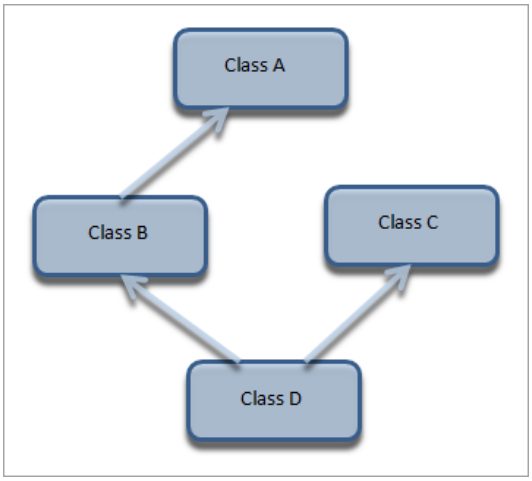


Syntax:

```
class Class_A {
        //class specific code;
 };
class Class_B public Class_A {
        //class specific code;
 };
class Class_C : public Class_A {
        //class specific code;
 };
class Class_D public Class_B {
        //class specific code;
 };
class Class_E : public Class_B {
        //class specific code;
 };
class Class_F public Class_C {
        //class specific code;
 };
class Class_G : public Class_C {
```

5. Hybrid Inheritance:

Hybrid inheritance is usually a combination of more than one type of inheritance. In the next figure, we have multiple inheritance (B, C, and D) and multilevel inheritance (A, B, and D) to get a hybrid inheritance.



Syntax:

```
class Class_A {
        //class specific code;
 };
class Class_B public Class_A {
        //class specific code;
 };
class Class_C {
        //class specific code;
 };
class Class_D public Class_C, public Class_B
{
        //class specific code;
 };
```