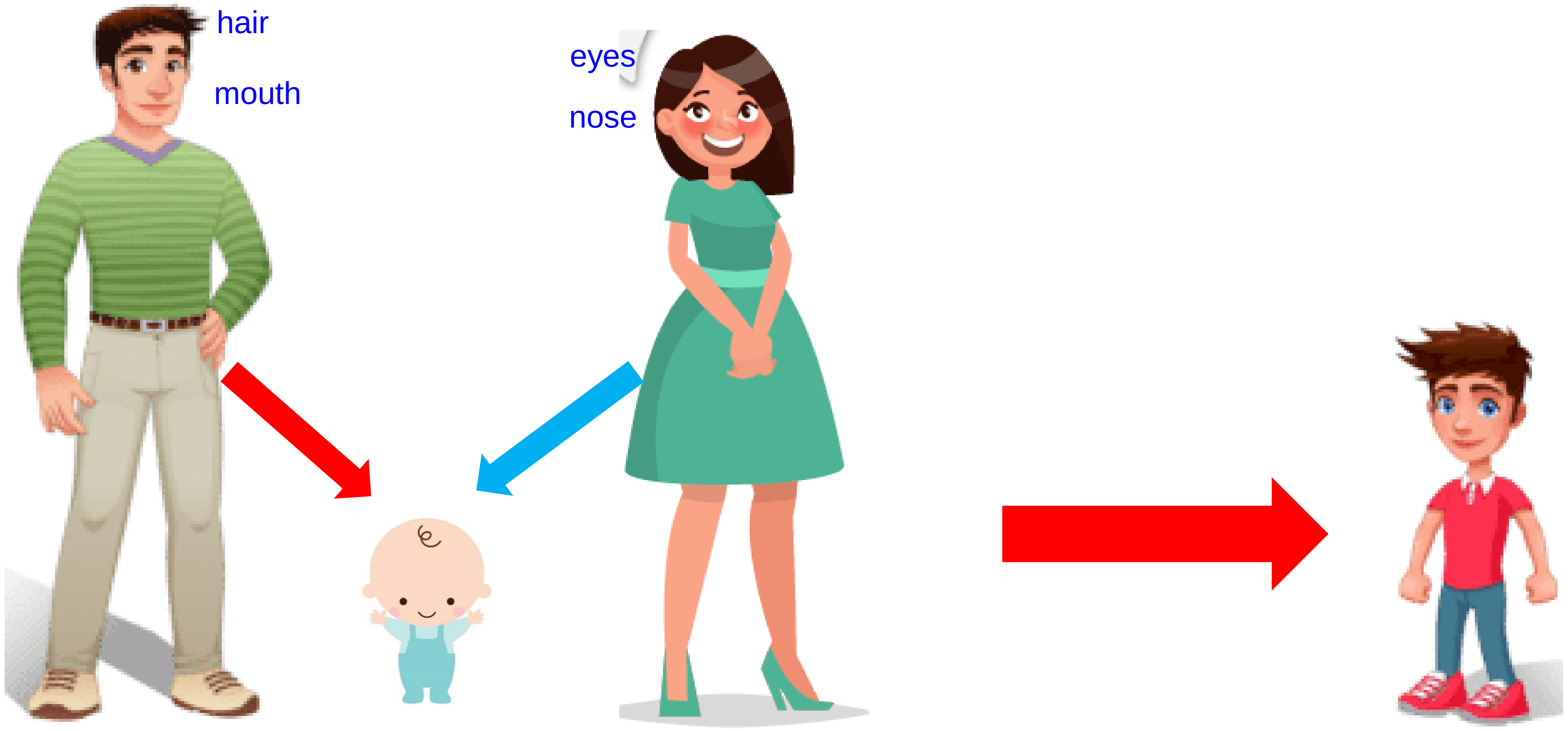
الجامعة المستنصرية / كلية العلوم
قسم علوم الحاسوب

OOP

OBJECT-ORIENTED PROGRAMMING

7

INHERITANCE


hair
mouth
eyes
nose

Inheritance is the process of acquiring properties and behaviors from one object to another object.

Allows programmers to create new classes based on an existing class.

Methods and attributes from the parent class are inherited by the newly-created class

New methods and attributes can be created in the new class, but don’t affect the parent class’s definition

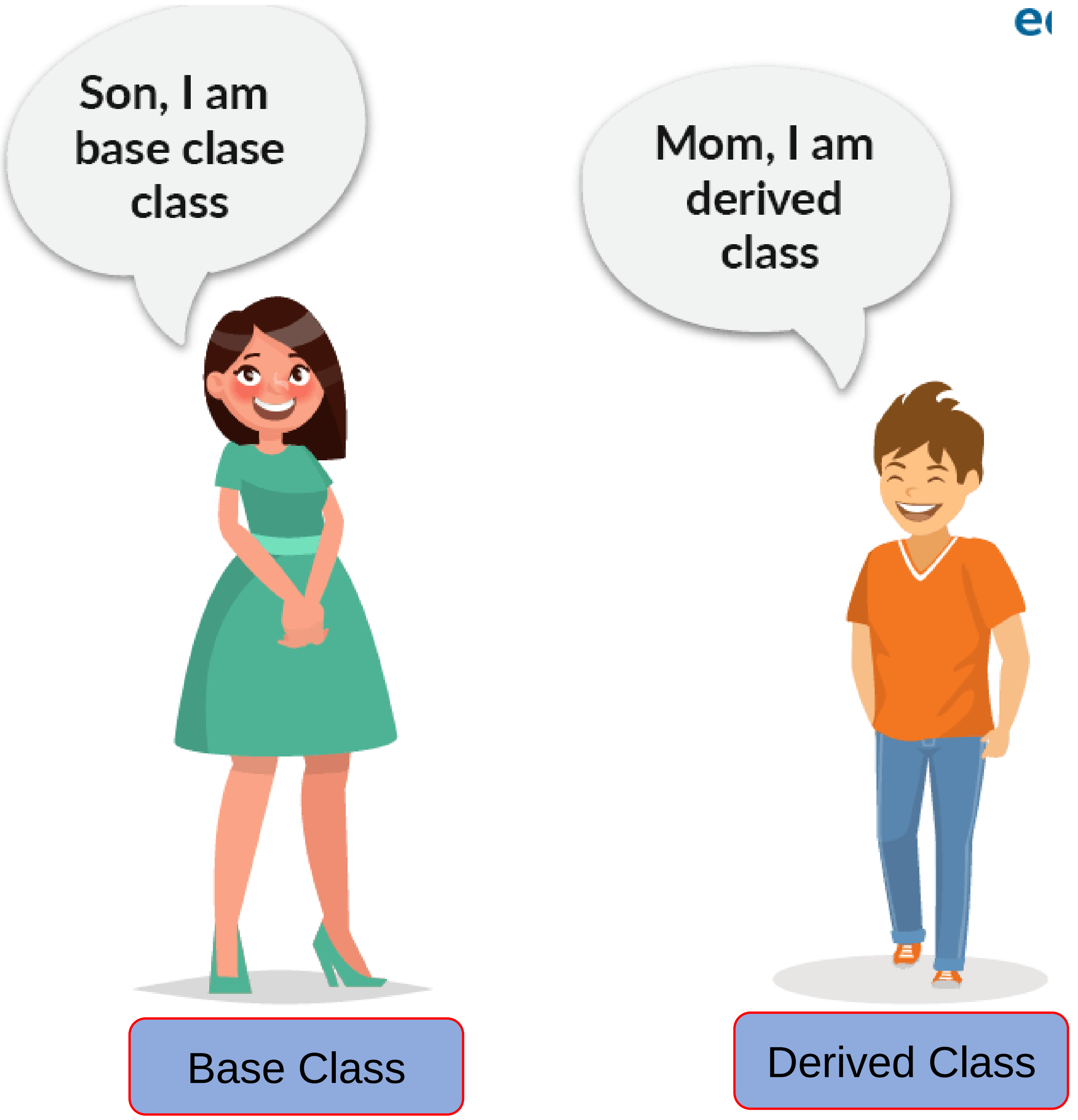
In inheritance, we derive a new class from the existing class.

The parent class is also known as base class or super class.

The child class is also known as derived class or sub class.

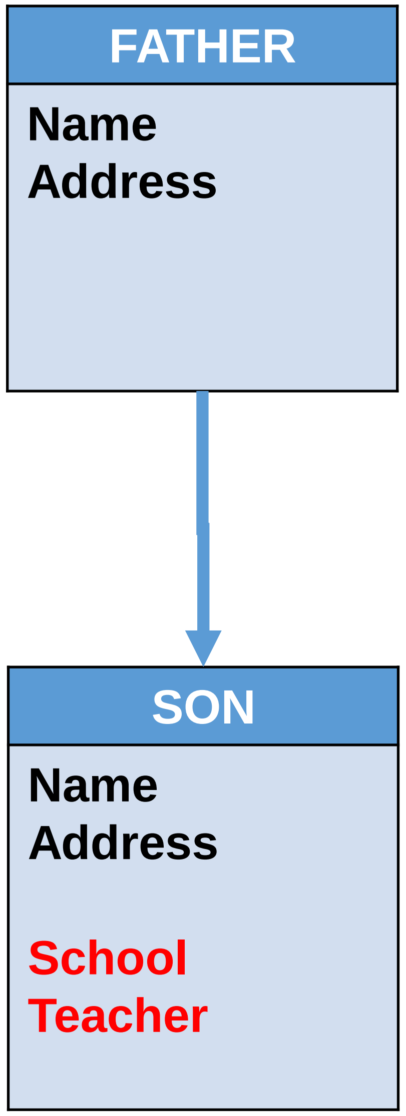
A subclass is also called a **derived class** and the class from which it is derived (parent class) is called **superclass** or **base class**.

- **Derived Class** (child) - the class that inherits from another class
- **Base Class** (parent) - the class being inherited from



To inherit from a class, use the symbol **:**

```
class son : father
```



```
using System;
namespace inheritance
{
 class Program
 {
    class father
   {
     public string name;
     public string address;
   }

   class son : father
   {
     public string school;
     public string teacher;
   }
}
```

```
static void Main(string[] args)
        {
            son  stu = new son();

            stu.name     = "Ahmad";
            stu.address = "Baghdad";
            stu.school   = "Mustansiria";
            stu.teacher  = "Mohammad";

            Console.ReadLine();
        }
 }
}
```

(in Inheritance) : The members (attributes) of the class should be **protected** so they can be accessed within that class or its subclass.

```
using System;
namespace inheritance
{
 class Program
 {
   class father
   {
     protected string name;
     protected string address;
   }

   class son : father
   {
     public string school;
     public string teacher;
   }
}
```

```
static void Main(string[] args)
        {
            son stu  = new son();

            stu.name     ="Ahmad";
            stu.address = "Baghdad";

            stu.school   = "Mustansiria";
            stu.teacher  = "Mohammad";

            Console.ReadLine();
        }
 }
}
```

```
using System;
namespace inheritance
{
    class father
    {
      protected string name;
      protected string address;
    }

    class son : father
    {
      private string school;
      private string teacher;
    }
```

```
static void Main(string[] args)
        {
            son stu  = new son();



            Console.ReadLine();
        }
 }
}
```

```
using System;
namespace inheritance
{
 class Program
 {
    class father
   {
     public string name;
     public int d1,d2,d3;
   }

   class son : father
   {
     public double av;

   }
}
```

```
static void Main(string[] args)
        {
            son stu = new son();

            stu.name     ="ALI";
            stu.d1=90; stu.d2=80; stu.d3=70;

            stu.av  = (d1+d2+d3)/3;
            Console.WriteLine("AVAREGE="+av);

            Console.ReadLine();
        }
 }
}
```

```
using System;
namespace inheritance
{
   class person
   {
     protected string name;
     protected string address;
   }

   class student : person
   {
     private string school;
     private string teacher;

    public void readinfo()
     {
      name     = "Ahmad";
      address = "Baghdad";
      school   = "Mustansiria";
      teacher = "Mohammad";
      }
     public void printinfo()
     {
      Console.WritLine("NAME       =" + name);
      Console.WritLine("ADDRESS =" + address);
      Console.WritLine("SCHOOL    =" + school);
      Console.WritLine("TEACHER =" + teacher);
    }
   }
```

```
static void Main(string[] args)
        {
            student    stu    = new student();

            stu.readinfo();
            stu.printinfo();

            Console.ReadLine();
        }
 }
}
```

C# and .NET support *single inheritance* only. That is, a class can only inherit from a single class. However, inheritance is transitive, which allows you to define an inheritance hierarchy for a set of types. In other words, type D can inherit from type C, which inherits from type B, which inherits from the base class type A. Because inheritance is transitive, the members of type A are available to type D.

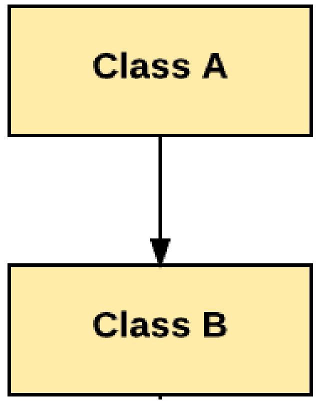
A class can be derived from more than one classes, which means it can inherit data and functions from multiple base classes.

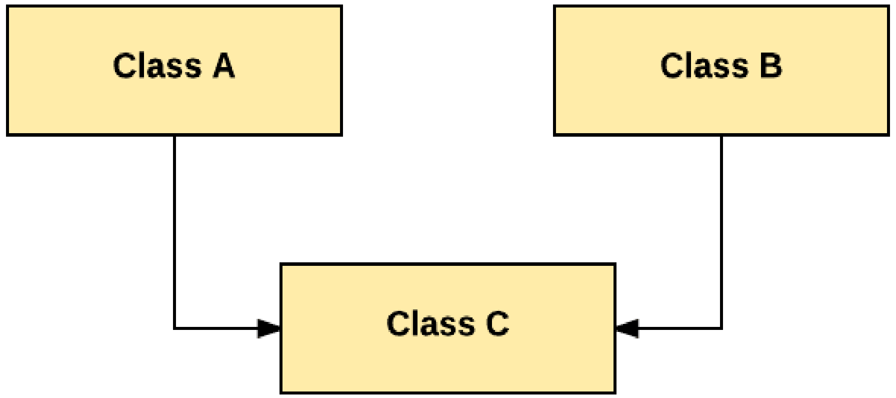
Class hierarchy

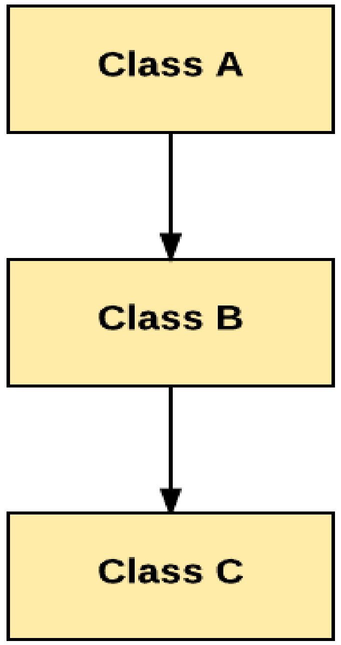
The parent-child relationship between classes can be represented in a hierarchical view often called *class tree view*.
The class tree view starts with a general class called superclass (sometimes referred to as *base class*, *parent class*, *ancestor class*, *mother class* or *father class*),

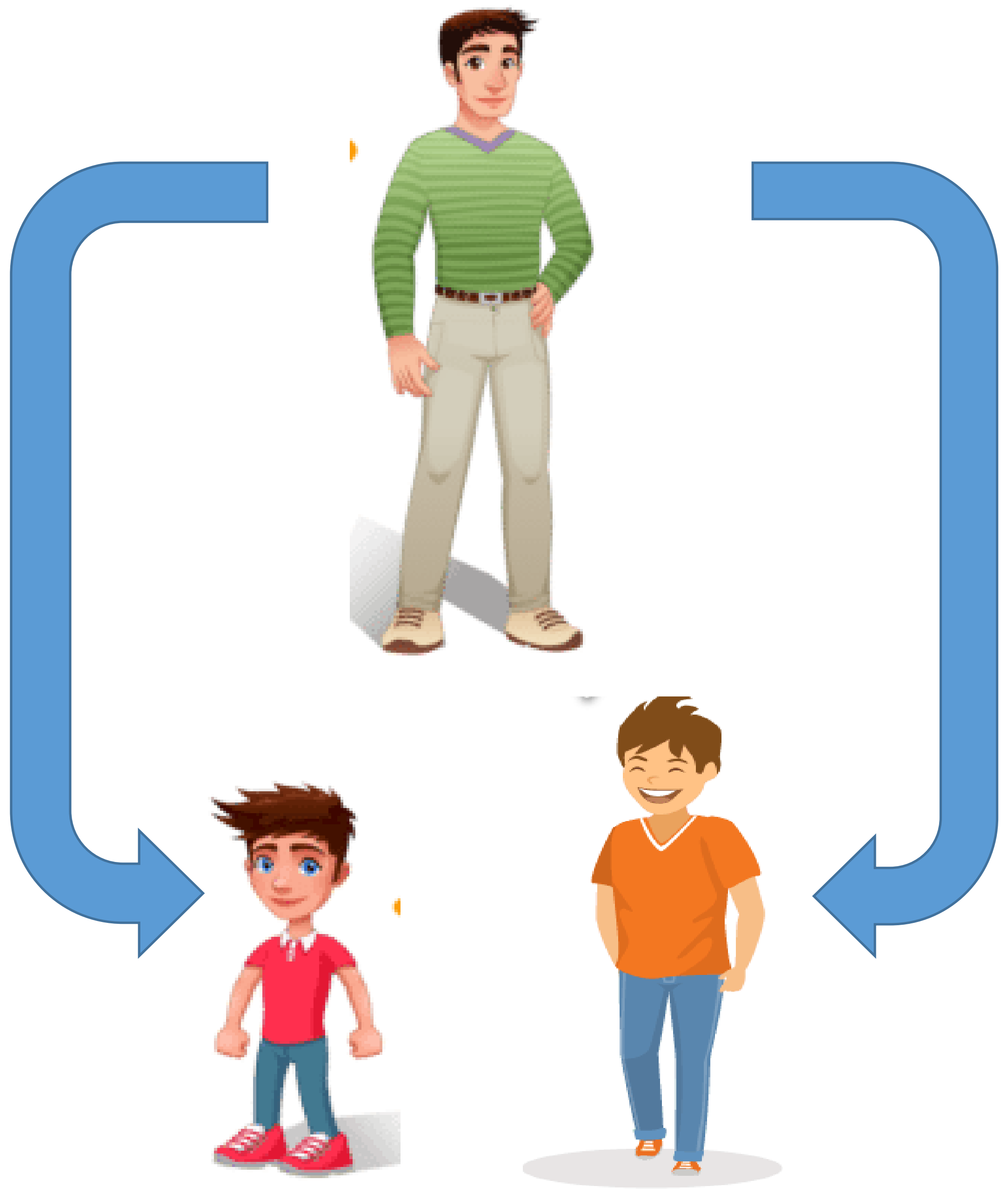
Derived classes (*child class* or *subclass*).

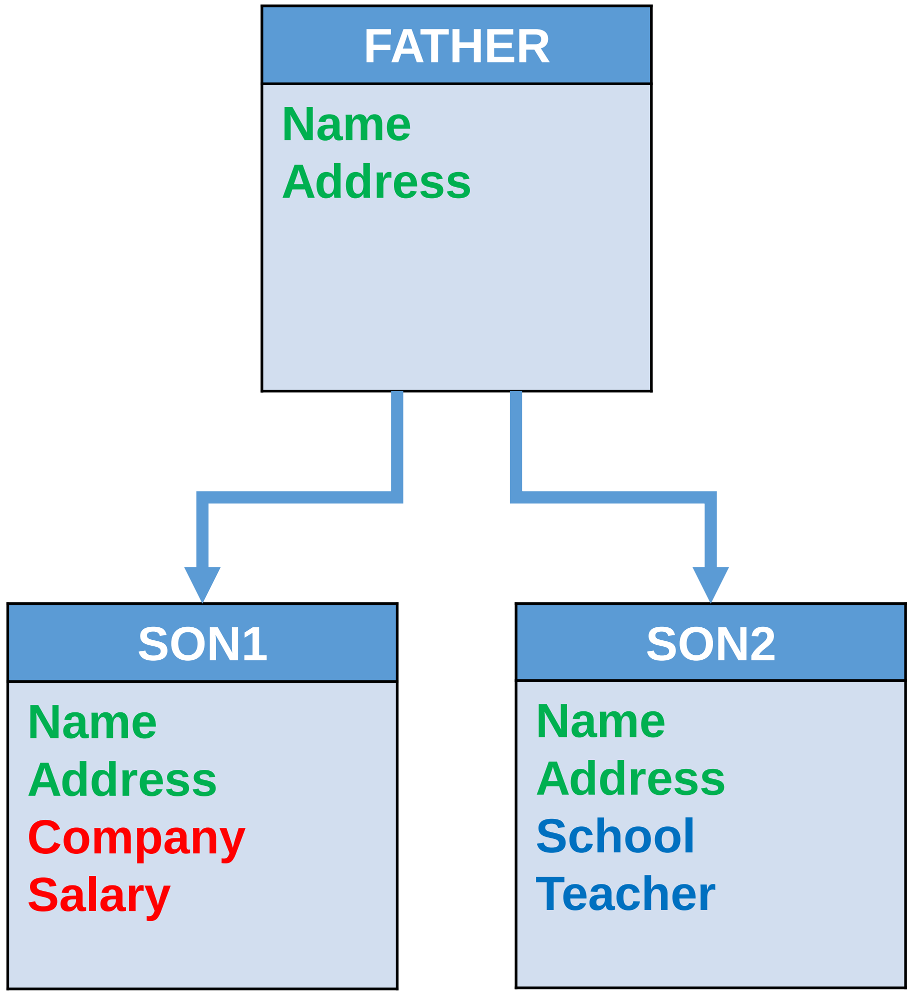
Single Inheritance	Multiple inheritance	Hierarchical Inheritance
In Single Inheritance one class extends another class (one class only).	Some object oriented languages, such as C++ allow multiple inheritance, meaning that one class can inherit attributes from two superclasses. This method can be used to group attributes and methods from several classes into one single class.	In Hierarchical Inheritance, one class is inherited by many sub classes. Class B, C, and D inherit the same class A.



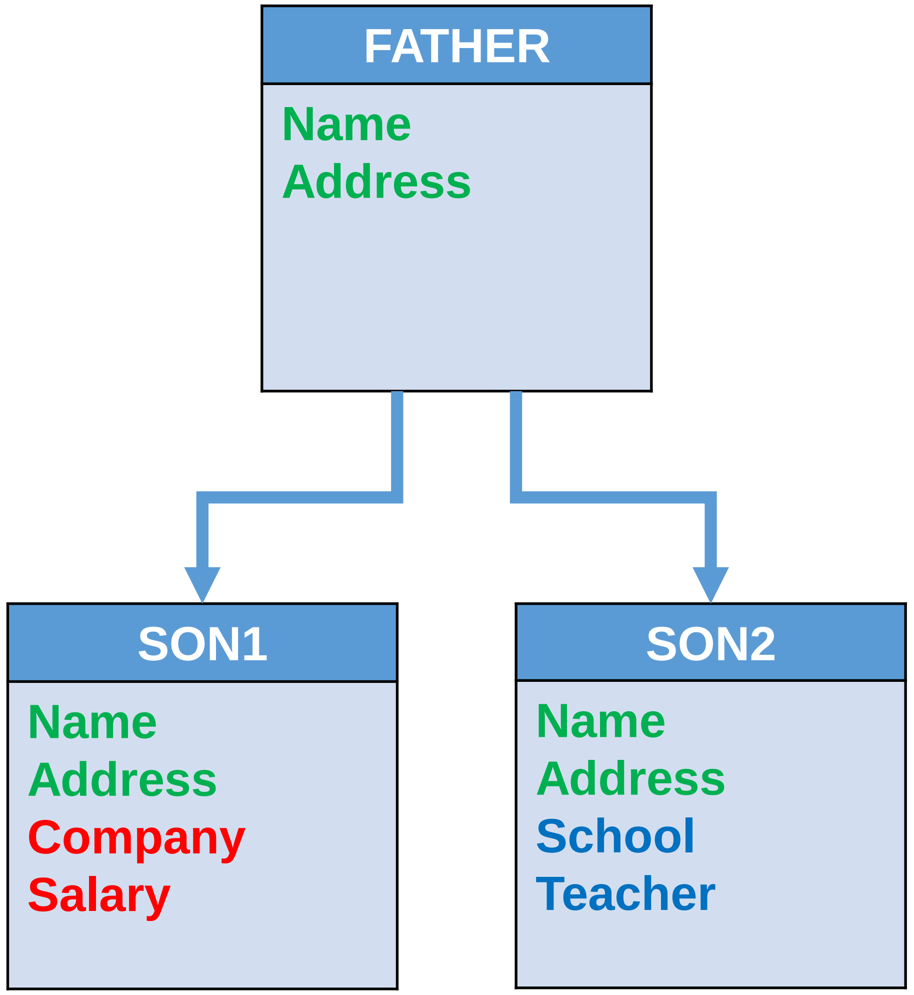






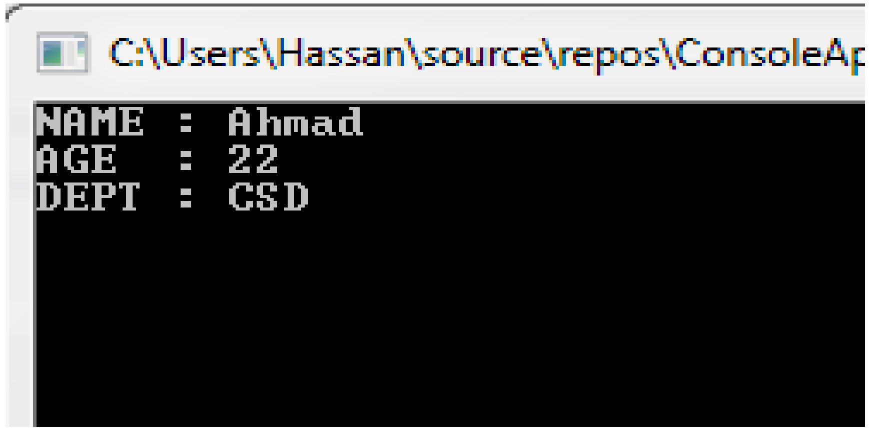

FATHER
Name
Address
SON1
Name
Address
Company
Salary
SON2
Name
Address
School
Teacher

```
using System;
namespace inheritance
{
    class father
    {
      protected string name;
      protected string address;
    }

      class son1 : father
      {
          private string company;
          private double salary;
      }

      class son2 : father
      {
          private string school;
          private string teacher;
      }
```



```
using System;
namespace inheritance
{
  class student
  {
    protected string name;
    protected int age;
    public void readinfo()
    {
       name = "Ahmad";
       age    = 22;
    }
  }

  class person : student
   {
       private string dept="CSD";
       public void printinfo()
       {
          Console.WriteLine("NAME : " + name);
          Console.WriteLine("AGE    : " + age);
          Console.WriteLine("DEPT  : " + dept);
       }
   }
}
```

```
static void Main(string[] args)
       {
           person st = new person();

           st.readinfo();

           st.printinfo();

           Console.ReadLine();
       }
   }
 }
```


C:\Users\Hassan\source\repos\ConsoleAp

```
NAME  :  Ahmad
AGE   :  22
DEPT  :  CSD
```

The sealed Keyword

If you don't want other classes to inherit from a class, use the sealed keyword:
If you try to access a sealed class, C# will generate an error
sealed keyword is used to restrict a class from being derived.
We can also use sealed keyword with methods to prevent them for being overridden.

```
class Vehicle
{
  ...
}

class Car : Vehicle
{
  ...
}
```

```
sealed class Vehicle
{
  ...
}

class Car : Vehicle
{
  ...
}
```

```
using System;
namespace inheritance
{
 class Program
 {
  sealed class student
  {
   protected string name;
   protected int age;
   public void readinfo()
   {
      name = "Ahmad";
      age    = 22;
   }
  }
  class person : student
   {
      private string dept=“CSD”;
      public void printinfo()
      {
         Console.WriteLine("NAME : " + name);
         Console.WriteLine("AGE     : " + age);
         Console.WriteLine("DEPT   : " + dept);
      }
   }
```

```
static void Main(string[] args)
        {
            person st = new person();

            st.readinfo();

            st.printinfo();

            Console.ReadLine();
        }
 }
}
```

ERROR:
cannot derive from sealed type

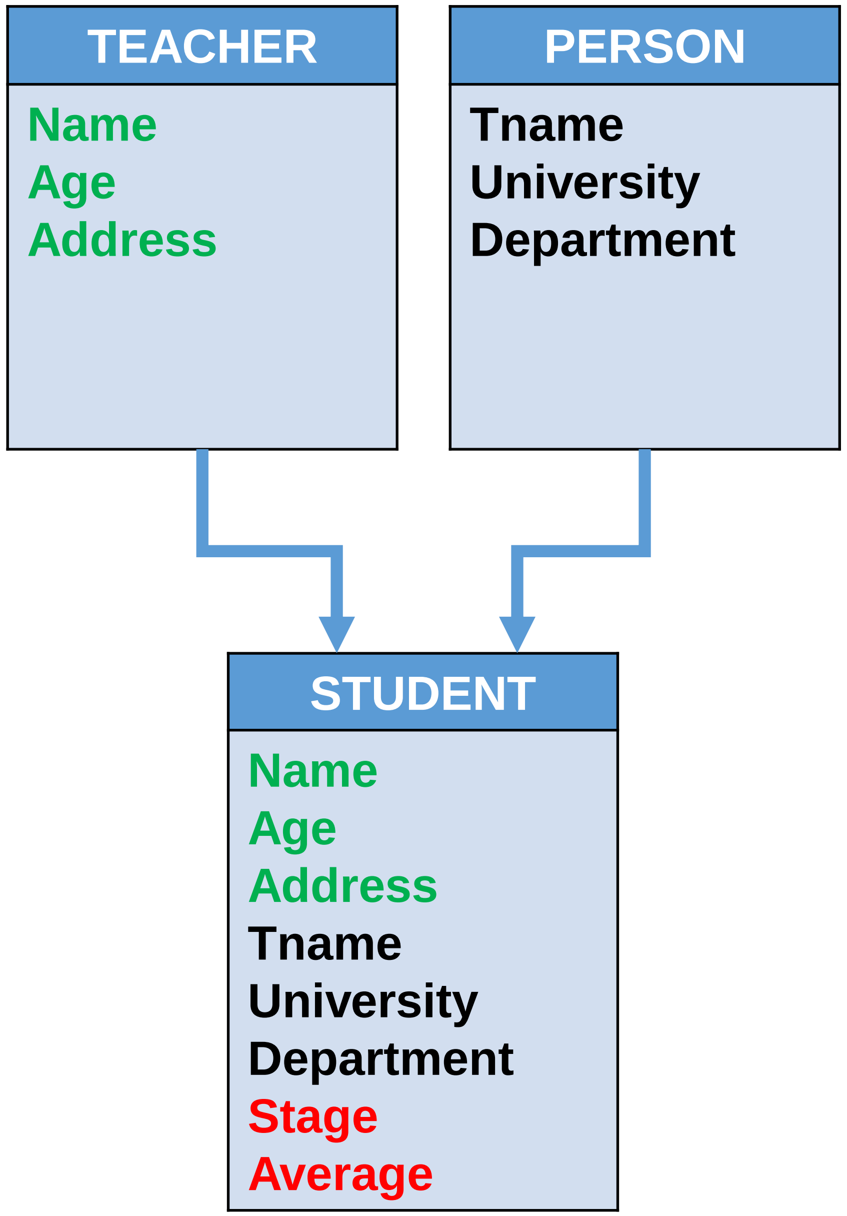
Why And When To Use "Inheritance"?

It is useful for code reusability:
reuse fields and methods of an existing class when you create a new class.

```
using System;
namespace inheritance
{
 class Program
 {
  class student
 {
  protected string name;
  protected int age;
  public void readinfo()
  {
     name = console.readline();
     age    = console.readline();
  }
 public void printinfo()
  {
    Console.WriteLine("NAME : " + name);
    Console.WriteLine("AGE    : " + age);
    Console.WriteLine("DEPT  : " + dept);
   }
 }
 class person : student
  {
     private string dept="CSD";
  }
```

```
static void Main(string[] args)
        {
            person st = new person();

            st.readinfo();

            st.printinfo();

            Console.ReadLine();
        }
 }
}
```

Write a class only to inherent

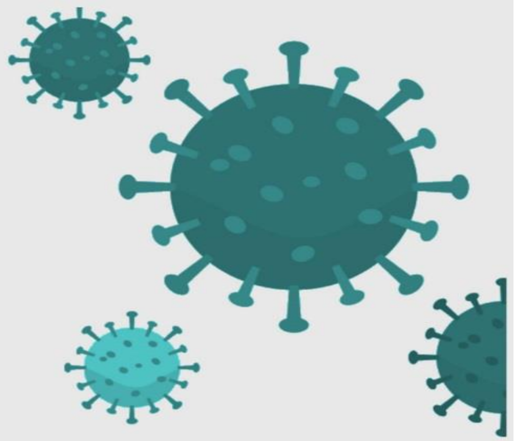


H.W.

Write complete program to find the largest number in array with 10 elements

7
3
10
40
12
1
5
2
9
6

Largest = 40
Location = 4



طرق الوقايا من فايروس كورونا

احرص دائما على تطهير يديك بعد
ملامسة الأسطح في الأماكن العامة

عند السعال والعطس قم بوضع منديل
والتخلص منه عند الإنتهاء في سلة المهملات

غسل اليدين بالماء والصابون لمدة
لا تقل عن 20 ثانية بشكل متكرر

قم بتطهير وتنظيف الأسطح التي تلامسها بشكل مُتكرر

ارتدِ قناعا واقيا كإجراء وقائي
في المستشفيات والأماكن المغلقة

(Association) : attributes and methods as you need

```
using System;
namespace inheritance
{
    class student
    {
        private string name;
        private string school;
        private string teacher;
    }

    class employee : student
    {
        private string name;
        private string company;
        private double salary;
    }
```

```
private string name;
private string school;
private string teacher;

private string name;
private string company;
private double salary;
```