Data Structures and Object Oriented Programming

Lecture 16

Dr. Naveed Anwar Bhatti

Webpage: naveedanwarbhatti.github.io

Inheritance

Inheritance

Definition:

“Process of extending existing class into new class is known as inheritance”

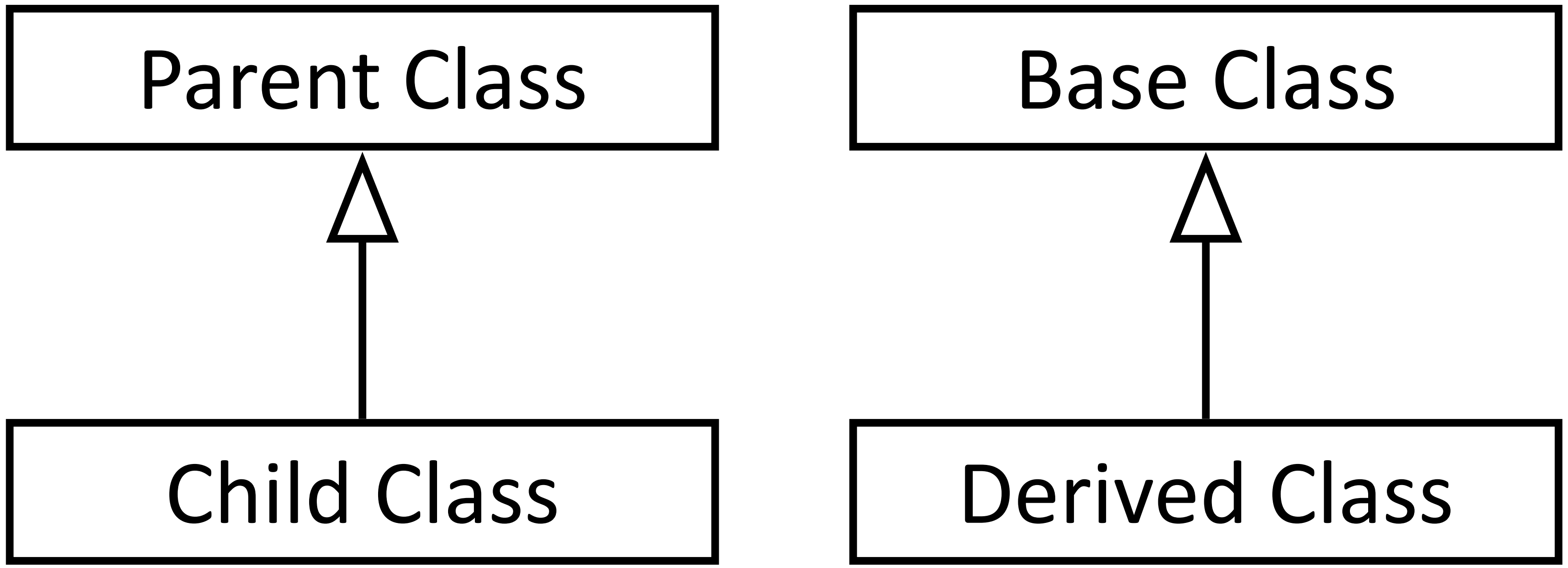
- Existing class is known as **Base Class** (or Parent Class)
- New class is known as **Derived Class** (or Child Class)

What is inherited from the base class?

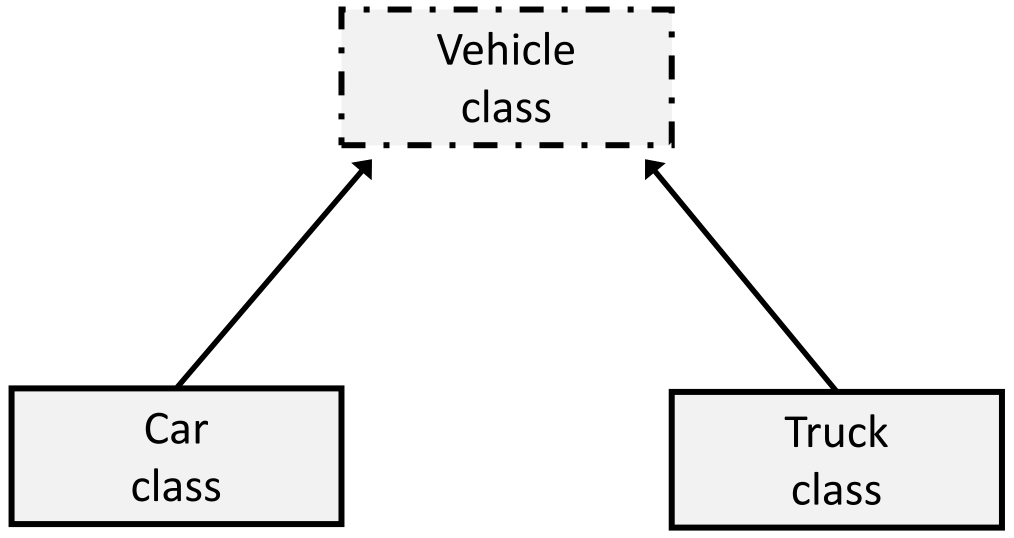
Every member of a base class except:

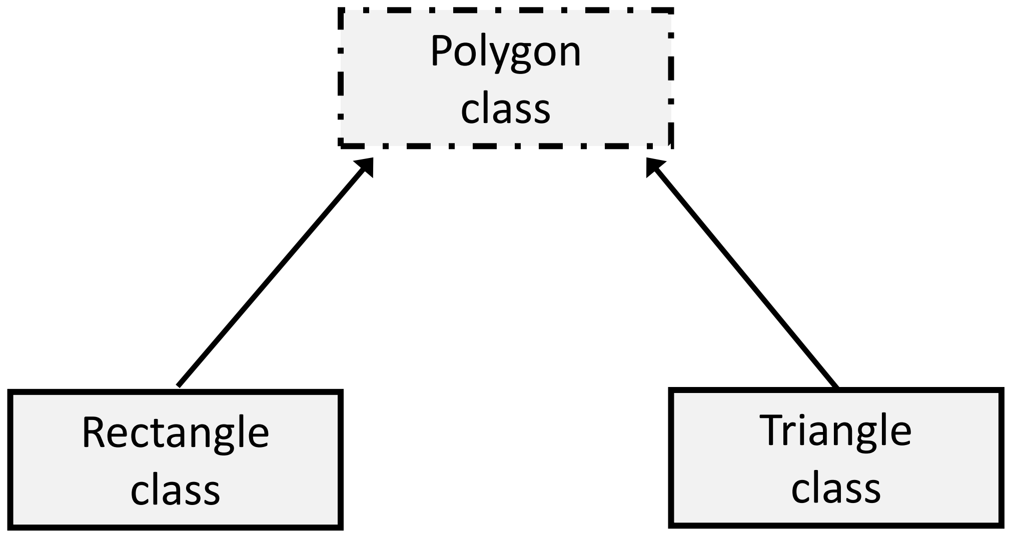
- constructors and its destructor
- assignment operator members (operator=)
- friends
- private members (**Hidden**)

UML Notation



Inheritance (example)





Inheritance (Syntax and Example)

Syntax

```
class ChildClass: public BaseClass
{
    ...
};
```

```
#include <iostream>
using namespace std;

class Parent
{
public:

    int x, y;
    void set_values(int a, int b)
    {
        x = a; y = b;
    }
};
```

```
class Child : public Parent {
  public:
    int multiply()
    {
        return x * y;
    }
};


int main()
{
    Child var1;
    var1.set_values(4, 5);
    cout << var1.multiply();
    return 0;
}
```

Access in Inheritance

- There are three different access control in inheritance
  - Public
  - Protected
  - Private
- Use keyword public, private or protected to define access level in inheritance

Public Inheritance

```
class Child: public Parent {…};
```

Member access in	
Base Class	**Derived Class**
Public	Public
Protected	Protected
Private	Hidden

Protected Inheritance

class Child: protected Parent {…};

Member access in	
Base Class	**Derived Class**
Public	Protected
Protected	Protected
Private	Hidden

Private Inheritance

```
class Child: private Parent {…};
```

Member access in	
Base Class	**Derived Class**
Public	Private
Protected	Private
Private	Hidden

Inheritance (example)

```
#include <iostream>
using namespace std;

class Polygon {
 private:
   int width, height;
 public:
   void set_values(int a, int b)
   {
      width = a; height = b;
   }
   int get_width()
   {
      return width;
   }
   int get_height()
   {
      return height;
   }

};
```

Inheritance (example)

```cpp
#include <iostream>
using namespace std;

class Polygon {
 private:
   int width, height;
 public:
   void set_values(int a, int b)
   {
      width = a; height = b;
   }
   int get_width()
   {
      return width;
   }
   int get_height()
   {
      return height;
   }

};
```

```cpp
class Rectangle : public Polygon {
  public:
    int area()
    {
       return get_width() * get_height();
    }
};
```

Inheritance (example)

```cpp
#include <iostream>
using namespace std;

class Polygon {
 private:
   int width, height;
 public:
   void set_values(int a, int b)
   {
      width = a; height = b;
   }
   int get_width()
   {
      return width;
   }
   int get_height()
   {
      return height;
   }

};
```

```cpp
class Rectangle : public Polygon {
  public:
    int area()
    {
       return get_width() * get_height();
    }
};


int main()
{
    Rectangle rec1;
    rec1.set_values(4, 5);
    cout << rec1.area();
    return 0;
}
```

Inheritance (example)

```
#include <iostream>
using namespace std;

class Polygon {
 private:
   int width, height;
 public:
   void set_values(int a, int b)
   {
      width = a; height = b;
   }
   int get_width()
   {
      return width;
   }
   int get_height()
   {
      return height;
   }

};
```

```
class Rectangle : protected Polygon {
  public:
    int area()
    {
       return get_width() * get_height();
    }
};


int main()
{
    Rectangle rec1;
    rec1.set_values(4, 5);
    cout << rec1.area();
    return 0;
}
```

Error

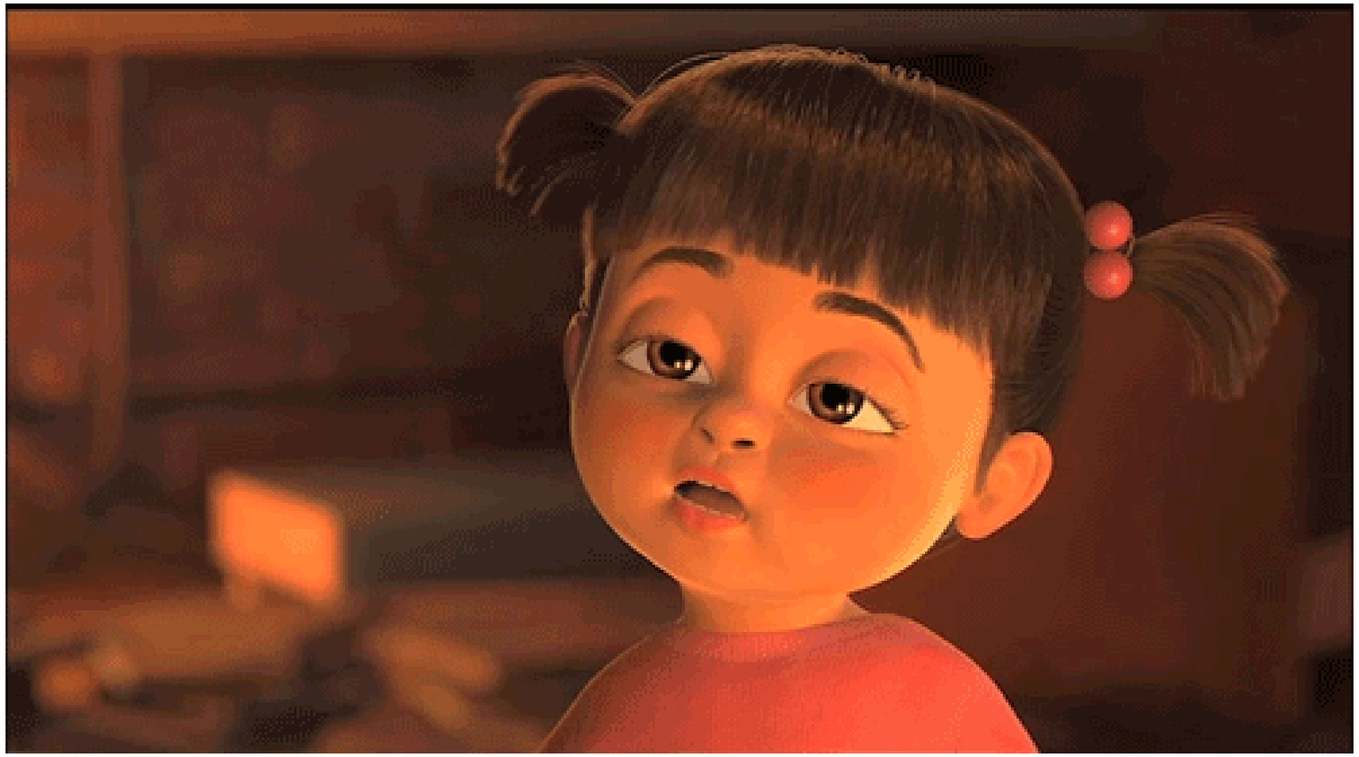
Inheritance (example)

```cpp
#include <iostream>
using namespace std;

class Polygon {
 private:
   int width, height;
 public:
   void set_values(int a, int b)
   {
      width = a; height = b;
   }
   int get_width()
   {
      return width;
   }
   int get_height()
   {
      return height;
   }

};
```

```cpp
class Rectangle : private Polygon {
  public:
    int area()
    {
       return get_width() * get_height();
    }
};


int main()
{
    Rectangle rec1;
    rec1.set_values(4, 5);
    cout << rec1.area();
    return 0;
}
```

Error

Thanks a lot



If you are taking a Nap, **wake up**.......Lecture Over