COEN-244
Tutorial #6

February 23rd, 2023

POLYMORPHISM

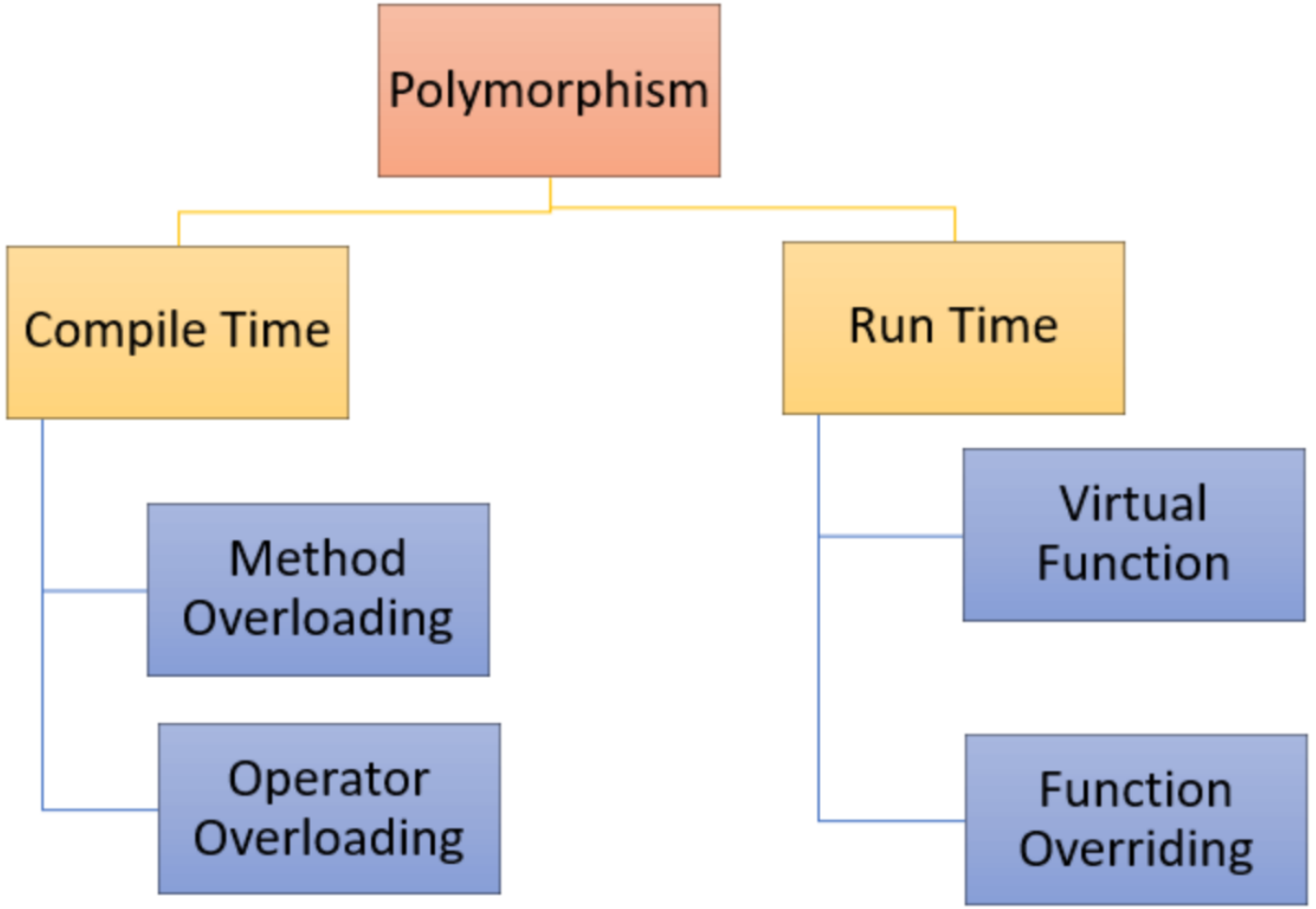
Polymorphism simply means more than one form.

- The same entity (function or operator) behaves differently in different scenarios

Different ways to do POLYMORPHISM:

- Static and dynamic binding + virtual functions
- Function overloading
- Function overriding
- Virtual destructors
- Pure virtual functions, Pure abstract classes

Where can we see Polymorphism?



Source: https://www.guru99.com/cpp-polymorphism.html

Polymorphism Exercise

- You have a class called `Mammal` which contains the data member and methods as below:

```
class Mammal
{
    public:
            Mammal(void);
            ~Mammal(void);
            virtual void Move() const;
            virtual void Speak() const;
    protected:
            int Age;
};
```

TASK: Develop additional classes for Cat, Horse, and Dog overriding the move and speak methods. These class should be inherited. *Tests to be performed using the test code.*

Source Code: https://github.com/TheBarzani/COEN244_W2023/tree/main/tut6

Type Conversion

- In **COEN243** we saw that a data type could be converted from one type to another. E.G., assigning a `bool` to `int`
- In class hierarchies, we can similarly perform conversions.
- For Example: This can be done by assigning objects of a derived class to a pointer of its parent class. (E.G., Potatoes and fries)

Two ways:

- **Implicit:** Done by using the assignment operator.
- **Explicit:** Needs extra effort/code from the programmer.

Implicit Type Conversion

The objects created by the derived class are considered as a specialization of the generalized Base class and exhibit the Is-A property.

- A Derived object contains more data than the base object like `int` and `float`

Three ways:

- Base Class Objects Assignment
- Base Class Pointer Assignment
- Reference Variable of Base Class

Explicit Type Conversion

In explicit type conversion, the programmer explicitly specifies the type conversions.

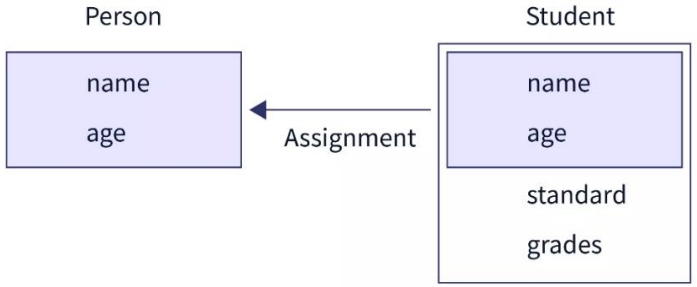
- We have to check if the explicit conversion is possible or not!

Two Types:

- **Upcasting:** Upgrading a base class pointer to derived one.
- **Downcast:** Downgrading a derived class pointer to an upgraded base class pointer

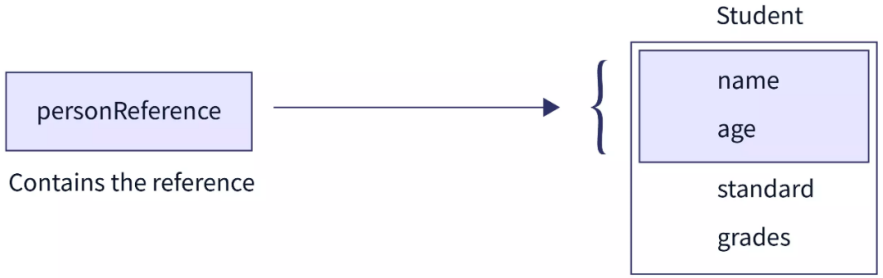
Type Conversion Comparison

Implicit Type Conversions



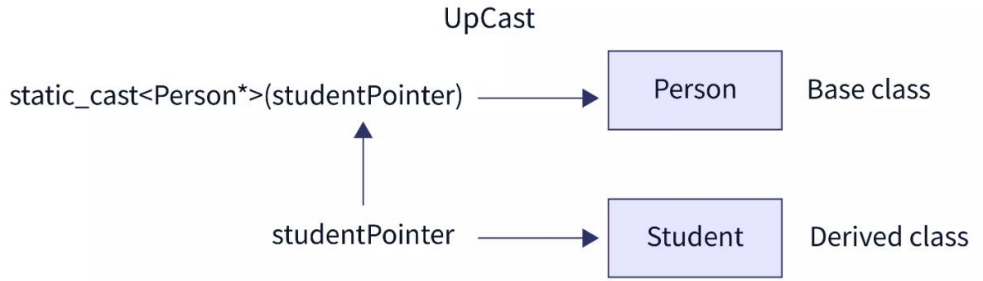
Base Class Objects Assignment

Base Class Pointer Assignment

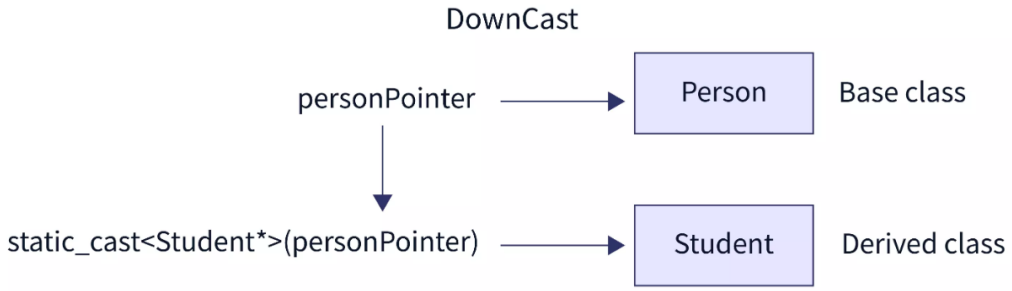


Reference Variable of Base Class

Explicit Type Conversions



Upcasting Type Conversion



Downcasting Type Conversion

THANK YOU