Object Oriented Methodology

Week – *4*, Lecture – *1*

Multiple Inheritance

SAURABH SRIVASTAVA

VISITING FACULTY

IIIT LUCKNOW

What we have covered already …

A derived class inherits properties and behaviours from its base class …

- … excluding, of course, the `private` members

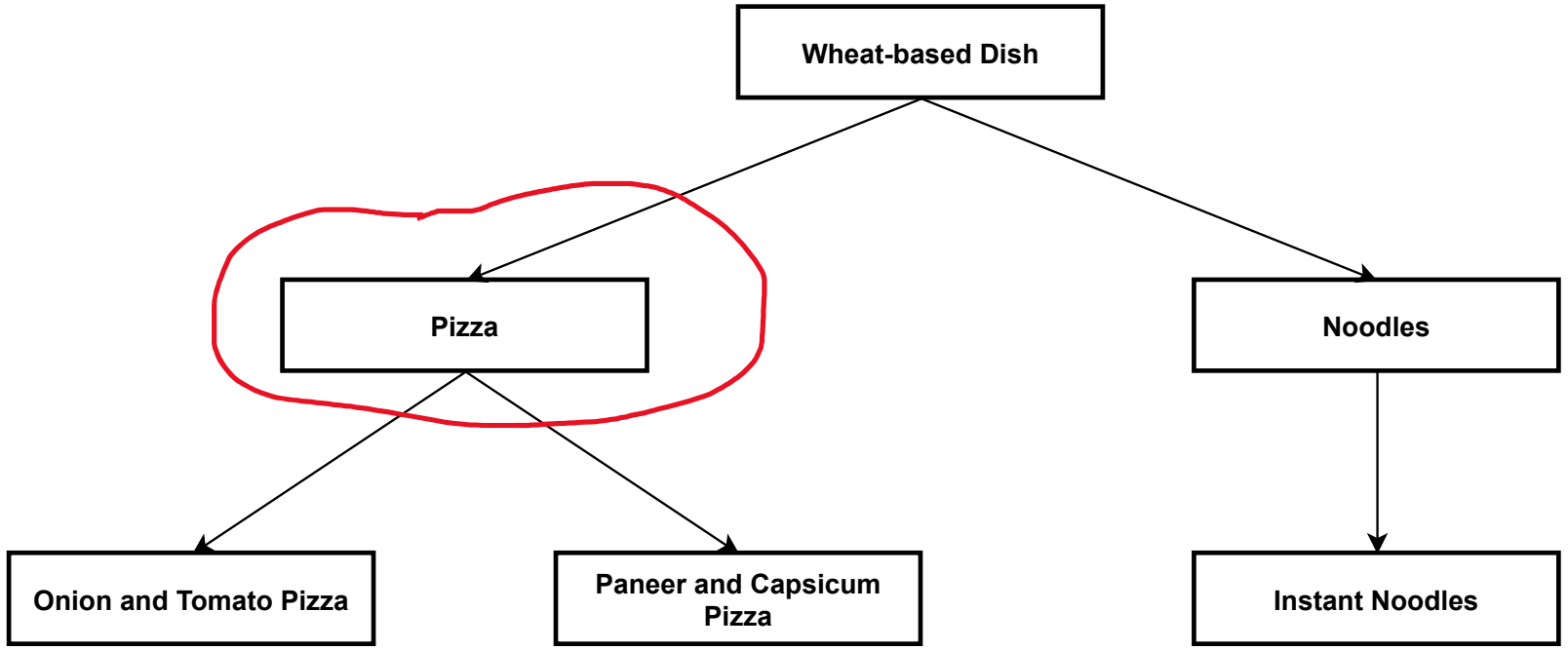
If a particular method “may have to be” reimplemented in derived classes …

- … such a function should be declared as `virtual` in the base class
- If the function’s definition is not provided at all in the base class, it is called a “pure” virtual function

There can be a hierarchy of classes too …

- … meaning the same class can be a derived class w.r.t. to one class, and a base class w.r.t. another class

Inheritance – Building stuff hierarchically



An example of Inheritance Hierarchy

Here, *Pizza* is a derived class of *Wheat-based Dish*, and a base class for *Onion and Tomato Pizza*

What we have covered already …

A derived class inherits properties and behaviours from its base class …

- … excluding, of course, the `private` members

If a particular method “may have to be” reimplemented in derived classes …

- … such a function should be declared as `virtual` in the base class
- If the function’s definition is not provided at all in the base class, it is called a “pure” virtual function

There can be a hierarchy of classes too …

- … meaning the same class can be a derived class w.r.t. to one class, and a base class w.r.t. another class

This hierarchy can be seen as a list of ”specialisation – generalisation” relationships …

- … e.g. *Pizza* is a type of *Wheat-based Dish* and *Pizza* can be of different types …
- … e.g. *Onion and Tomato Pizza* or *Paneer and Capsicum Pizza*

More “complex” hierarchies

What we have seen till now, are cases where a derived class has had *only one* base class

- Note that the same base class could have had multiple derived classes

This type of inheritance scenario is called *Single Inheritance*

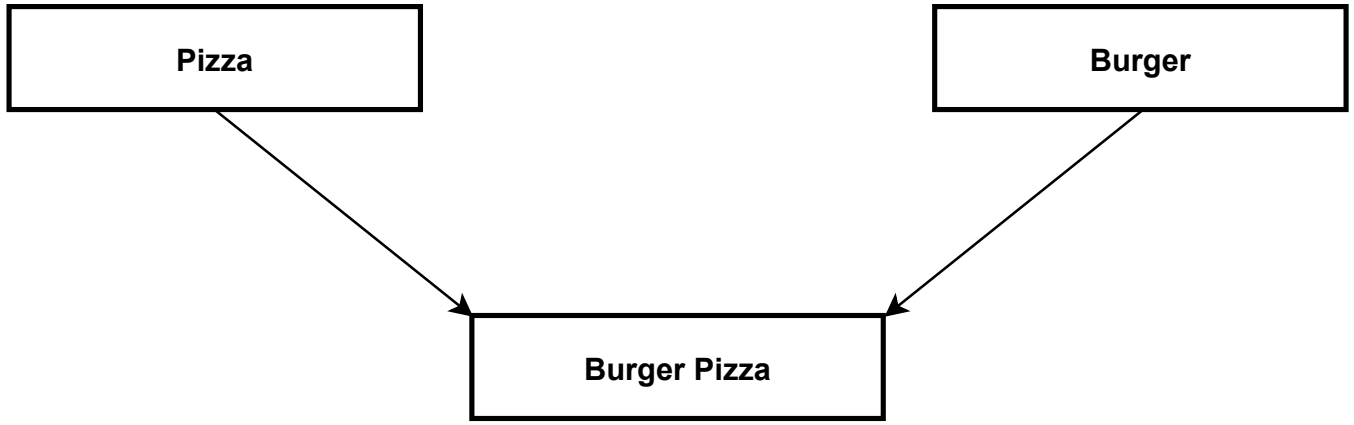
Another form of inheritance that you may see is where a derived class has multiple bases classes

- This type of inheritance scenario is called *Multiple Inheritance*

… and here's an example :P

Source: https://www.dominos.co.in/menu/burger-pizza

Multiple Inheritance – Class Hierarchy



In terms of Inheritance, *Burger Pizza* has two base classes – *Pizza* and *Burger*

More “complex” hierarchies

What we have seen till now, are cases where a derived class has had *only one* base class

- Note that the same base class could have had multiple derived classes

This type of inheritance scenario is called *Single Inheritance*

Another form of inheritance that you may see is where a derived class has multiple bases classes

- This type of inheritance scenario is called *Multiple Inheritance*

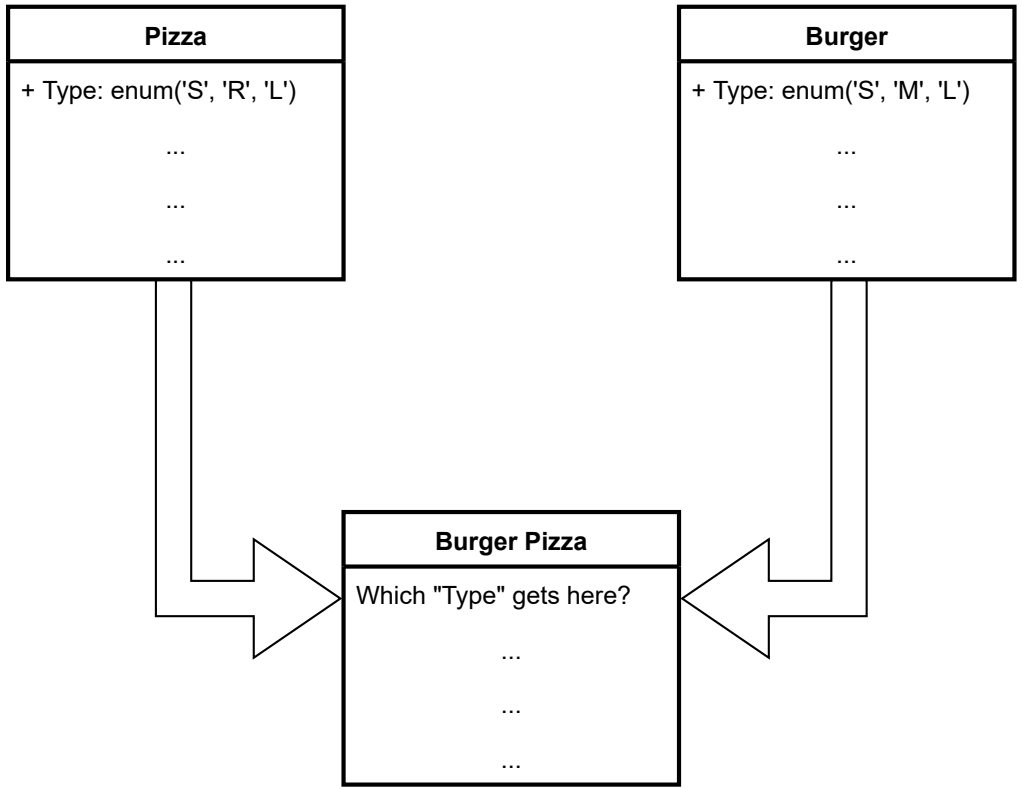
Multiple Inheritance is a “controversial” topic (to say the least)

- There are some (including me) who opine that it complicates the design
- For example, the ”specialisation – generalisation” relationships become rather complex

As an example, assume that a property or behaviour is common in both base classes …

- … which of these two versions are inherited – one, both or none?

A typical problem with Multiple Inheritance



Common members in base classes can create problems with Multiple Inheritance

More “complex” hierarchies

What we have seen till now, are cases where a derived class has had *only one* base class

- Note that the same base class could have had multiple derived classes

This type of inheritance scenario is called *Single Inheritance*

Another form of inheritance that you may see is where a derived class has multiple bases classes

- This type of inheritance scenario is called *Multiple Inheritance*

Multiple Inheritance is a “controversial” topic (to say the least)

- There are some (including me) who opine that it complicates the design
- For example, the ”specialisation – generalisation” relationships become rather complex

As an example, assume that a property or behaviour is common in both base classes …

- … which of these two versions are inherited – one, both or none?
- The solution to the problem is left to the implementation – different languages may deal with it differently
- But in one particular case, this problem becomes even more interesting …

The Diamond Problem

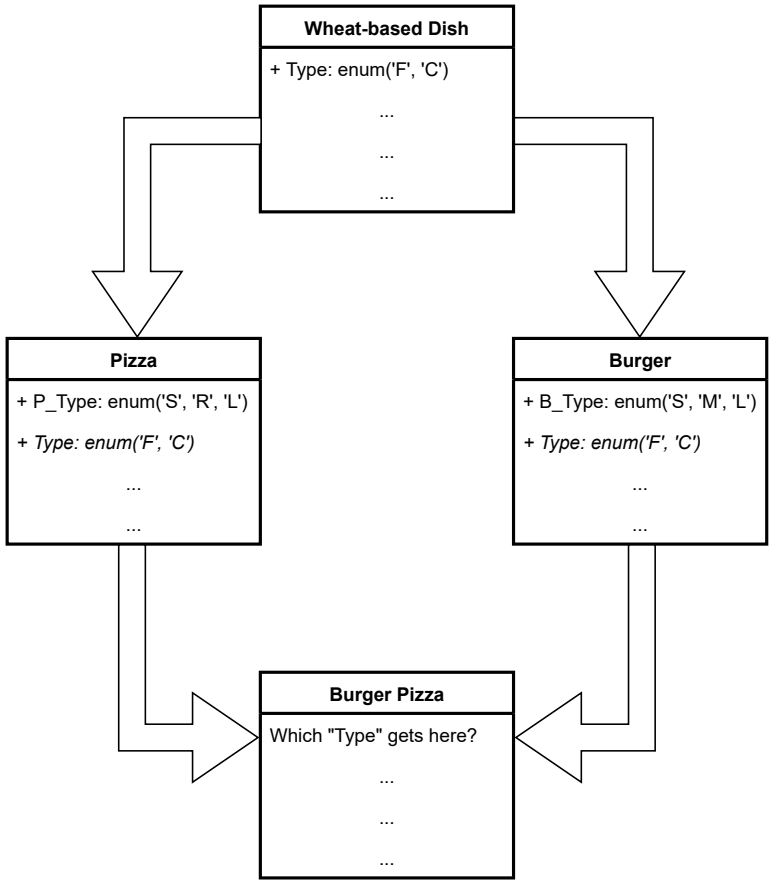
What if we change the names of the members in the example to `P_Type` and `B_Type`?

- Now, there is no confusion – The *Burger Pizza* class can inherit both of them without any issues

But in one peculiar situation, even this renaming cannot save the day

- Assume that *Pizza* and *Burger* have a common base class, e.g. *Wheat-based Dish*
- Now, all "inheritable" members of *Wheat-based Dish* become a part of *Pizza* and *Burger* as well
- Provided that these items remain inheritable, the same problem that we saw before will have to be solved
- This situation is termed as the *The Diamond Problem*

Problem with Diamond-shaped Inheritance



When parent classes for a class have the same grandparent, members from the grandparent pose a problem

The Diamond Problem

What if we change the names of the members in the example to `P_Type` and `B_Type`?

- Now, there is no confusion – The *Burger Pizza* class can inherit both of them without any issues

But in one peculiar situation, even this renaming cannot save the day

- Assume that *Pizza* and *Burger* have a common base class, e.g. *Wheat-based Dish*
- Now, all "inheritable" members of *Wheat-based Dish* become a part of *Pizza* and *Burger* as well
- Provided that these items remain inheritable, the same problem that we saw before will have to be solved
- This situation is termed as the *The Diamond Problem*

The Diamond Problem is one of the strong reasons to avoid multiple inheritance in design

- The solutions, as discussed before, are language specific

In C++, the implementation maintains a single copy of such fields …

- … and for methods, you can access specific versions using the scope resolution operator

Homework !!

Read more about the Diamond Problem, and how it could possibly be resolved

- This section on the Wikipedia page of Multiple Inheritance is more than enough to get an idea https://en.wikipedia.org/wiki/Multiple_inheritance#The_diamond_problem