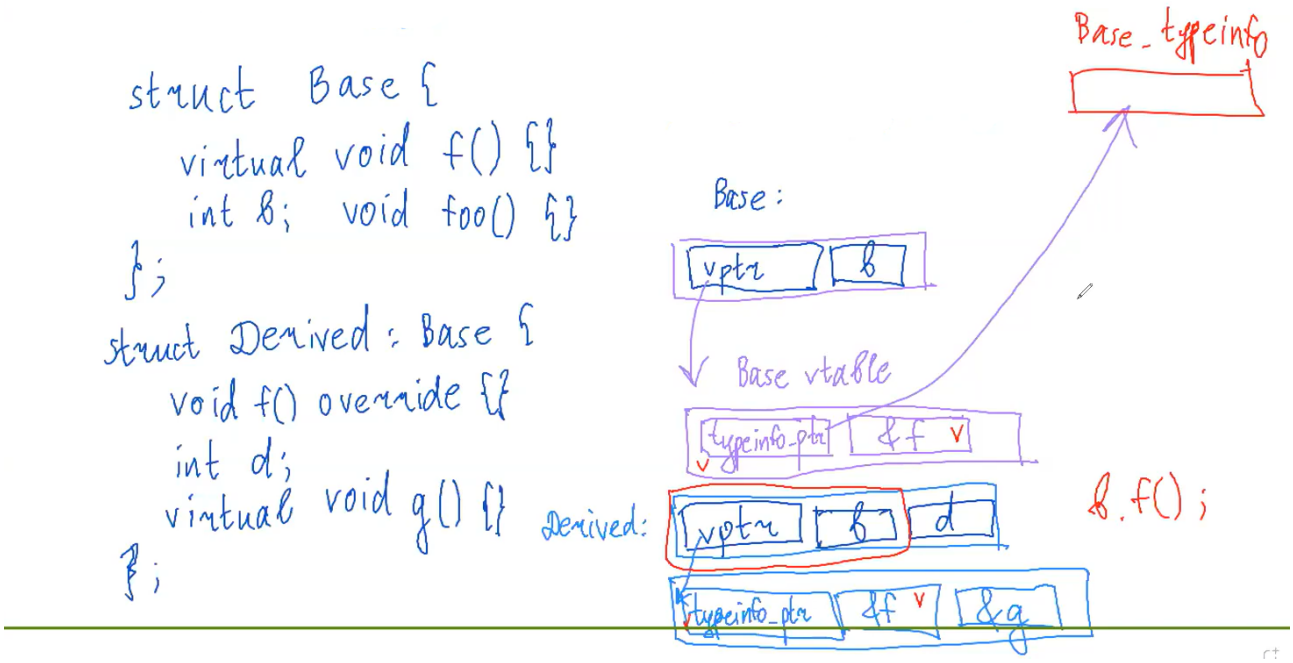
5.2. Понятие полиморфных объектов. Таблица виртуальных функций для полиморфного объекта, ее содержимое. Размещение в памяти объектов, у которых есть виртуальные функции. Объяснение, как за счет vtable происходит выбор нужной версии функции, а также за счет чего работают dynamic_cast и typeid. Разница между статическим и динамическим выбором версии функции с точки зрения реализации на низком уровне.

Рассмотрим полиморфный объект (у класса которого есть хотя бы один виртуальный метод) в памяти. Помимо его полей там будет храниться некоторый указатель - указатель на vtable. В этой таблице лежит указатель на место, где лежит typeinfo (нужно для typeid), а так же указатели на виртуальные функции.

Рассмотрим пример:



Как происходит выбор виртуальной f? В compile-time компилятор записывает инструкции: насколько шагов надо сдвинутся, чтобы получить указатель на f, и просто выполняет это в run-time.

Так как мы храним указатель на typeinfo работа typeid становится тривиальной. dynamic_cast теперь тоже может понять настоящий тип того что там лежит и сделать то что от него требуется (в примере dynamic_cast Base (красный прямоугольничек) к Derived (весь синий прямоугольничек)).

Заметим, что функция foo нигде не хранится рядом с объектами. Это объясняется тем, что про нее все известно в compile-time и нам не нужно ничего хранить в run-time для ее вызова.

Этим объясняется также разница статическим и динамическим выбором версии функции: статический - происходит в compile-time, динамический - в run-time через прыжки по нескольким указателям (что естественно медленнее чем в статическом случае).