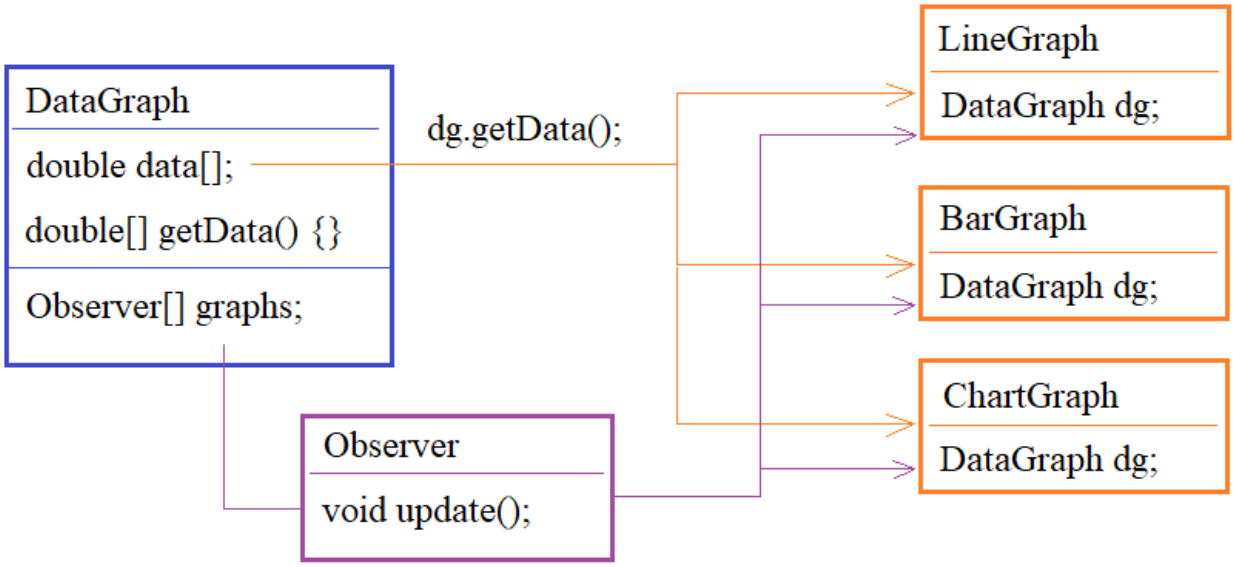
Путь кодера

Великий подвиг. Объявить класс DataGraph для хранения данных для графика в виде массива вещественных чисел размерностью N элементов (число N задать как константу, например, N=10). Записать отдельные классы (НЕ дочерние): LineGraph (точки в графике соединяются линиями), BarGraph (график в виде столбцов), ChartGraph (график в виде круговой диаграммы). При создании экземпляров этих классов они должны хранить ссылку на объект класса DataGraph. При рисовании графиков, данные следует брать через публичный метод getData() (класса DataGraph), т.е. получать ссылку на массив из N вещественных чисел. Взаимодействие между объектами классов должно выглядеть так:



Далее, объявить интерфейс Observer с методом update() и применить его к классам LineGraph, BarGraph и ChartGraph. По методу update() должно происходить обновление данных и перерисовка графика. В классе DataGraph хранить массив graphs для экземпляров классов LineGraph, BarGraph и ChartGraph. Как только происходит изменение данных в массиве data, вызывать метод update через ссылки graphs. (Изменение данных делать искусственно, например, в программе поменять данные, а затем, вызвать некий метод в DataGraph для запуска вызовов update).

Если все сделать правильно, то управление перерисовкой графиков будет выполняться через интерфейс Observer и благодаря этому классы графиков могут иметь любую структуру наследования, т.к. мы у них не задаем никаких базовых классов. В этом преимущество данной схемы реализации.

Для самых неистовых. В данной реализации класс DataGraph должен иметь только один экземпляр. Поэтому здесь целесообразно реализовать метод getInstance(), который бы возвращал ссылку на объект класса и контролировал бы единственность этого объекта. При этом нужно закрыть возможность создавать экземпляр класса напрямую через оператор new.