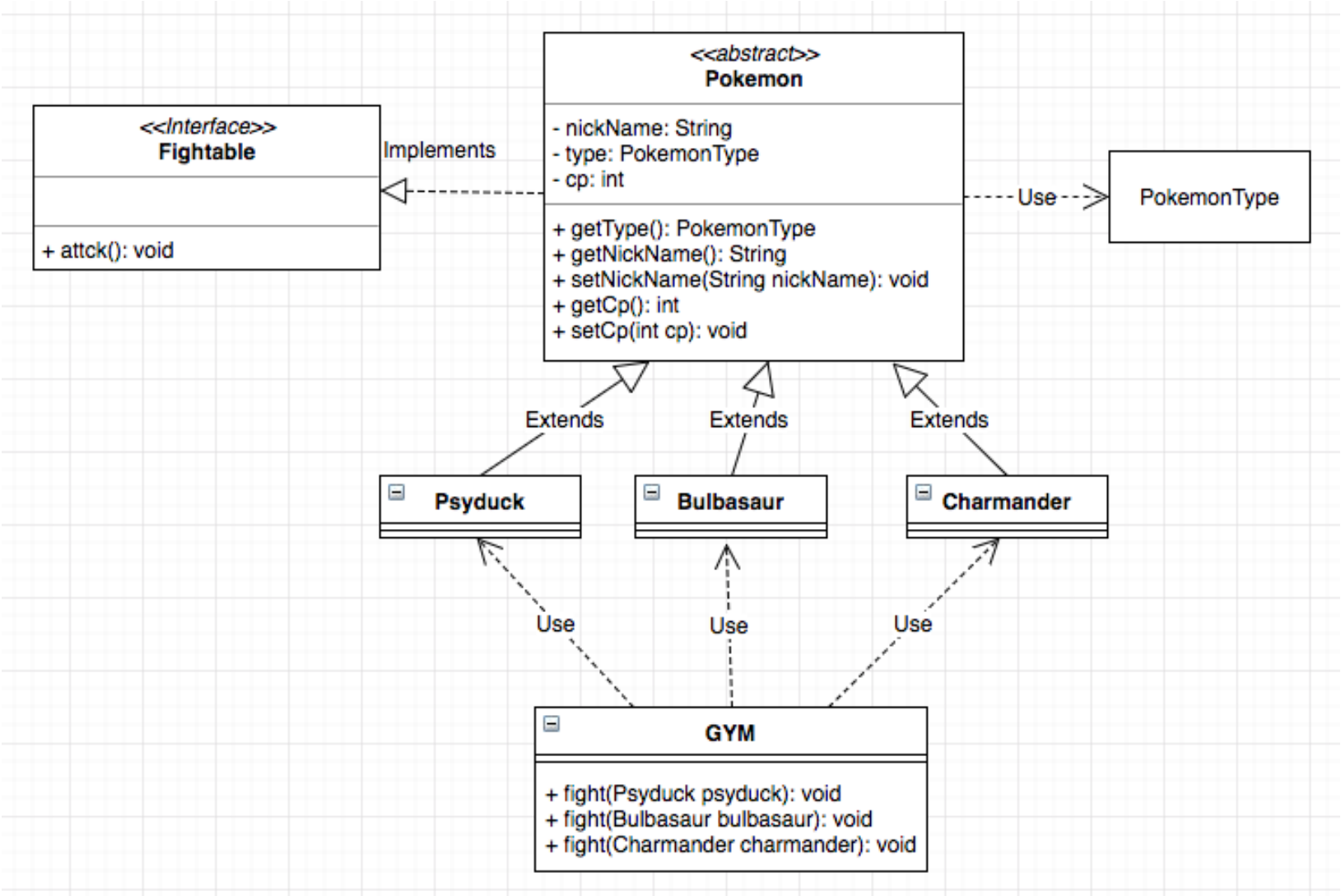
類別圖


<<Interface>>
Fightable
+ attck(): void
Implements
<<abstract>>
Pokemon
- nickName: String
- type: PokemonType
- cp: int
+ getType(): PokemonType
+ getNickName(): String
+ setNickName(String nickName): void
+ getCp(): int
+ setCp(int cp): void
Use
PokemonType
Extends
Extends
Extends
Psyduck
Bulbasaur
Charmander
Use
Use
Use
GYM
+ fight(Psyduck psyduck): void
+ fight(Bulbasaur bulbasaur): void
+ fight(Charmander charmander): void

Fightable.java

```
public interface Fightable {
  public void attack();
}
```

PokemonType.java

```
public enum PokemonType {
  FIRE, WATER, GRASS
}
```

Pokemon.java

- 此**抽象類別**擁有三個屬性：匿稱(nickName)、種類(type)與cp值
- nickName的型態為字串
- type的型態為PokemonType
- cp值的型態為整數
- type的值一旦被初始化後就不能修改
- 屬性值只能被類別內的建構子或方法使用
- type必須提供相對應的Accessor(Getter) 方法
- nickName與cp必須提供相對應的Accessor(Getter)與Mutator(Setter)方法
- 此類別內的所有方法都是對外開放使用，即任何其他類別都可以呼叫Pokemon類別所提供的方法
- 此類別只提供一個建構子其可以對三個屬性值做初始化

子類別

- Psyduck.java
  - 實作attack 方法
    - 當attack被呼叫時印出”Aqua Tail…”
- Bulbasaur.java
  - 實作attack 方法
    - 當attack被呼叫時印出” Tackle…”
- Charmander.java
  - 實作attack 方法
    - 當attack被呼叫時印出”Ember…”

Gym.java

- 此類別目前有三個方法如下

```
public void fight(Psyduck psyduck) {
 psyduck.attack();
}

public void fight(Bulbasaur bulbasaur) {
 bulbasaur.attack();
}

public void fight(Charmander charmander) {
 charmander.attack();
}
```

Main.java

```
public class Main {

  public static void main(String[] args) {
    Bulbasaur pkm1 = new Bulbasaur("I am Bulbasaur", PokemonType.GRASS, 123);
    Charmander pkm2 = new Charmander("I am Charmander", PokemonType.FIRE, 456);
    Psyduck pkm3 = new Psyduck("I am Psyduck", PokemonType.WATER, 89);

    Gym gym = new Gym();
    gym.fight(pkm1);
    gym.fight(pkm2);
    gym.fight(pkm3);
  }

}
```

- 修改類別圖，使得Gym類別可以展現「多型(Polymorphism)」特性

- 相對應的Gym與Main程式碼也要隨之修改