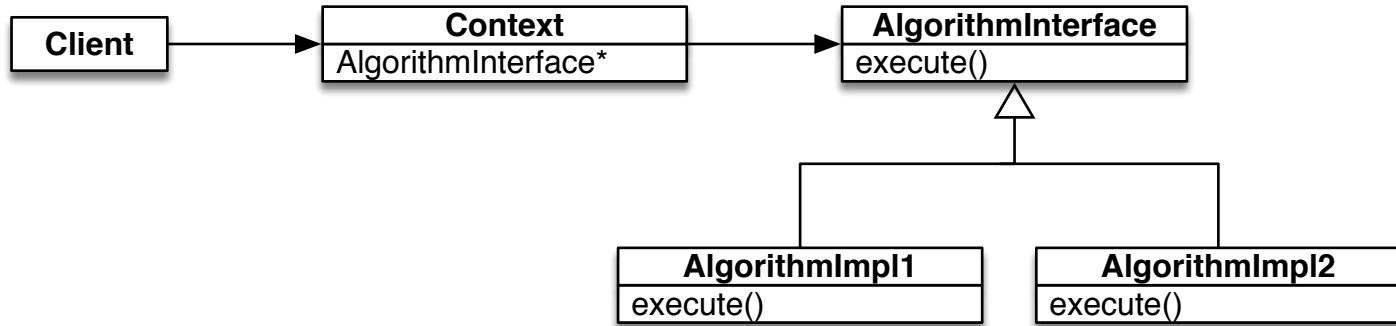## Strategy

Intent: multiple mechanisms may be interchangeable in performing some tasks. Strategy lets them involve independently (from the client).

```
┌──────────┐      ┌───────────────────────┐      ┌───────────────────────┐
│  Client  │─────▶│       Context         │─────▶│   AlgorithmInterface  │
└──────────┘      │  AlgorithmInterface*  │      │  execute()            │
                  └───────────────────────┘      └───────────────────────┘
                                                            △
                                         ┌──────────────────┴──────────────────┐
                                 ┌───────────────────┐            ┌───────────────────┐
                                 │   AlgorithmImpl1  │            │   AlgorithmImpl2  │
                                 │  execute()        │            │  execute()        │
                                 └───────────────────┘            └───────────────────┘
```

Pretty straightforward concept. Most common use case for polymorphism and what Liskov substitution is about.
Note that the interchangeable algorithms should have the same interface.
Client may need to be aware of which concrete algorithm it wants to use, when creating a context. Alternatively a context can have a default AlgorithmImpl.
If strategy can be determined at compile time, then we can use templates instead of inheritance, saving some runtime cost.