The Command Pattern encapsulates a request as an object, thereby letting you parameterize other objects with different requests, queue or log requests, and support undoable operations.

* The Command Pattern decouples an object making a request from one that knows how to perform it
* A command object is at the center of this decoupling and encapsulates a receiver with an action (or set of actions)
* An invoker makes a request of a Command object by calling its execute() method, which invokes those actions on the receiver.
* Invokers can be parameterized with Commands, even dynamically at runtime.
* Commands may support undo by implementing an undo method that restores the object to its previous state before the execute() method was last called.
* Macro commands are a simple extension of Command that allow multiple commands to be invoked. Likewise, Macro commands can easily support undo().
* In practice, it is not uncommon for “smart” command objects to implement the request themselves rather than delegating to a receiver
* Commands may also be used to implement logging, queuing and transactional systems.

In this, one of the example is using lambda expressions