1. Single responsibility principle
   - PlaceOrderController not only have to process the order but also have to check wheather the input string is validate (if a phone number string is really in phone number format; if a Name is really validate Name format), the fuctions is declared in the controller and may be duplicated if other controllers have to check the same format of some input string ➢ we can split all the check to a separate class
   - PlaceRushOrderCOntroller face the same problem as PlaceOrderController, have to process the RushOrder and have to the the input format.
2. Open/closed principle
   - I don't write the whole code but only write the PlaceRushOrderController, and while coding I don't fix the source code ➢ This principle is not violated.

3. Liskov substitution principle
   - BaseController and BaseScreenHandler isn't been used in the code, so it is hard to say, but in overall the children classes donn't overide the fuction in the father class, so it can still do all the what the father class can do ➢ This principle is not violated.

4. Interface segregation principle
   - There is only one interface in the project is InterbankInterface, and the interface has only 2 fuctions required, so it's not to big ➢ This principle is not violated.

5. Dependency inversion principle
   - Only 1 interface, all controllers based on BaseController and all ScreenHanlder based on BaseScreenHandler ➢ This principle is not violated