Joseph Gareri

[jgareri@my.athens.edu](mailto:jgareri@my.athens.edu)

0099183

March 16, 2024

CS 418

Assignment 06

Problem 1 and 2

1. By definition of Dependency Inversion Principle:
2. High-level modules should not depend on low-level modules. Both should depend on abstractions.
3. Abstractions should not depend on details. Details should depend on abstractions.

Thus, this Clojure code violates the DIP because the function “tic-tac-toe.core” is dependent on the lower-level functions. Shown below:

(ns tic-tac-toe.core

(:require [tic-tac-toe.board :refer :all]

[tic-tac-toe.ai :refer [ai.move]]

[tic-tac-toe.game :refer [player-1-marker player2-marker]]

[tic-tac-toe.human :refer [user-move]]

[tic-tac-toe.display :refer [display-iteration]]))

1. By definition of Liskov Substitution Principle:
2. If a class S is a subclass of class T, then an object of class T should be replaceable by an object of class S (S is-a T) without altering the desirable properties of the program.

Therefore, the “prepend” does not violate the LSP because by using a lower type bound (‘U >: T’), which implies that this method is valid for any types ‘U’ that are a superclass of ‘T’.

1. def prepend[U >: T](element: U): List[T] = new Cons(element, this)

https://stackify.com/dependency-inversion-principle/

https://medium.com/@Sabrina-Carpenter/liskov-substitution-principle-in-python-full-guide-ead6491b414