**Exercici Snakes and Ladders**

**1.** Expert: Podem veure com cada classe te informació d'allò que necessita, Board te tota la informació del taulell i crea les diferents caselles, SquareRole conté la informació de quin tipus de casella es tracta i quina funció tenen, etc.

Creator: La classe Game crea un objecte Board que conté tota la informació del taulell, la classe Board crea objectes Square que seran les caselles, podem veure que cada classe crea allò que s'espera que han de crear.

Low coupling: Al acomplirse els dos darrers patrons és molt fàcil que aquest també ho faci, totes les classes respeten les tasques que han de fer, no fan coses que no haurien de fer, no hi ha lligams entre classes innecessaris.

Polymorphism: Podem veure com SquareRole fa de superclasse sobre els diferents tipus de caselles, conté les funcions i atributs comuns i aquella subclasse que ho necessiti sobreescribeis aquelles funcions que necessita complementar amb funcionalitats extres.

**2.** Per afegir un rebot quan el jugador sobrepasa el nombre de caselles del taulell, hem de dirigir-nos a la funció ***moveAndLand*** de la classe ***SquareRole*** i modificar el que succeeix quan es compleix que la futura posició del jugador després de la tirada, serà més enllà de l’última casella del taulell. En aquest cas, si no haguéssim aplicat els principis de la programació orientada a objectes, possiblement hauríem d’haver modificat varies funcions de varies classes i d’un problema trivial tindríem un més tediós.

**3.** S’ha creat una classe ***SnakeLadderRoles*** que hereta de ***SquareRole***. Aquesta classe conté un atribut privat ***role*** per mostrar per pantalla un text o un altre, depenent de si hem caigut a una casella d’escala o de serp. La metodologia de les dues classes, tant de serp com d’escala, és la mateixa, així que només necessitàvem l’atribut ***role*** pel motiu anterior.