[: Dice] – does it really need a value?

[: Player, playerNumber]- the ArrayList<Player> players array will have an index of the arranged players. Therefore we don't need the playerNumber attribute.

[: Player, balanceAmount] - should be a double, otherwise we can't account for decimals in final balances and any operations need to be cast around, losing decimals places in the process.

[: Player, ArrayList<StartupSpace> ownedStartups] – The player doesn’t need an array of startups; the turn can check in the already existing array board, which spaces are owned by the player. (saves having to update the secondary array copy each time).

[: Player] - would be good to have a ownsStartup boolean - this way we can quickly check whether to he should be able to view properties.

[: Space, squareNumber attr] – Similar to playerNumber: the board will host the Space ArrayList and that's enough to calculate where players are. We don't need each space to have a square number since it’s assigned dynamically when populating the spaces array.

To automatically populate the board with however many spaces there are, we should have ENUMs for startups as well as actions, and these should have constructors. Otherwise distinction between Startup and Action subclasses won’t exist.

e.g.: ***BIGDAVE1***("Big Dave's WebDev - 1", "WebDev"),

***BIGDAVE2***("Big Dave's WebDev - 2", "WebDev"),

***SW1***("Software One - 1", "ServiceNow"),

***SW2***("Software One - 2", "ServiceNow"),

***SW3***("Software One - 3", "ServiceNow"),

***CLOUD1***("Cloud Experts - 1", "Cloud"),

***CLOUD2***("Cloud Experts - 2", "Cloud"),

***CLOUD3***("Cloud Experts - 3", "Cloud"),

***AI1***("FutureNow - 1", "AI"),

***AI2***("FutureNow - 2", "AI");

[: GameAdmin, boolean gameInPlay - might be better to have in TurnEngine; otherwise it needs to be public ( is it better to encapsulate the booolean?) ]

[: GameAdmin] needs a shufflePlayer() method. (This needs accounted for also before starting the game; playerNumber needs re-assigned, otherwise the player order is out of place.

[: TurnEngine, needs a movePlayer()] – Better to not bundle 2 methods into the landedOn() method. That one should exist only for checking what kind of place it landed on.

[:TurnEngine, needs a start()] - Better to contain the play to the turn engine itself:

[: TurnEngine] – beneficial to have a verifyChoice()

[: TurnEngine] - needs a checkOwnedStartups. Easiest way of implementing checking for what a player has (Good to encapsulate a turn method within the turn engine.)

[: Board, passInvest()] - This should be in TurnEngine. It will be triggered if the landed on detects player is in array 0.

[: IBalanceOperations, YAGNI] –This is a cool one for software engineering. https://en.wikipedia.org/wiki/You\_aren%27t\_gonna\_need\_it

It's only ever used by the turn engine. Would be better to host them in a [:Bank] instead.

[: IBalanceOperations] – beneficial to have a canAfford() – or preferably in a [:Bank] instead!

[: Messages] - would be nice to have a separate class with formatting of all the messages.