**Week 12 Assignment - Due: 17th Dec 2021**

**Rules**

1. **Don’t second guess yourself about what I might’ve intended you to do. This assignment is about what decisions you make based on what you’ve learned.**
2. **If you spot an anomaly in the description, just decide how to resolve it using the principles you’ve learned. When you submit your assignment, describe the anomaly, your decision and how you reached it - even if it was arbitrary.**
3. **If you see an opportunity to enhance the functionality or expand the idea in any way, feel free to do so.**
4. **Don’t spend time worrying about what I would’ve done. Do what you feel comfortable doing and have confidence that you know enough to complete this. If that means you can’t use a particular language feature or technique to satisfy one of the requirements cuz you don’t feel confident with that, then do it another way, but when you submit your assignment, tell me how you’d \*like\* to have done it and what stopped you, so that I can offer you some help with that**
5. **Have fun with it!**

**PART A - GREENFIELD**

**Objective:**

* **Write a black jack game.**

**Requirements:**

* **Load a database of players and dealers in JSON format**
  + **A sample file is given in Kata/Kata/blackjack.json**
  + **The format is:**
    - **Dealers = {“name”: “number-of-hands”}**
    - **Players = {“name”: “spend-limit”}**
* **Create 6 decks of playing cards. You’ll need numbers 2-10 + jack queen king and ace of hearts, diamonds, clubs and spades for each deck**
* **Load the chute randomly selecting cards across all 6 decks**
* **Select a dealer at random**
* **Ask the user to identify themselves. If they enter a known name, say “hi” to them and remind them of their limit.**
* **If they’re not known, register them and ask for their spend limit**
* **Represent the concept of a blackjack table with 6 positions for players to sit**
* **Randomly decide a number of other players to play at the same table**
  + **Make it possible for the new player to be playing alone**
* **Have each player place a £5 bet**
* **Let the user’s player decide their own bet but it must be at least £5**
* **No player can continue if they exceed their spend limit**
* **Deal one card to each player in order of their position at the table**
* **Deal one card to the dealer**
* **Deal another card to each player**
* **Deal one card to the dealer**
* **Display the value of each card including the first of the dealer’s card. The other card the dealer has is face down. Represent this however you like.**
* **Iterate thru the players in order of position at the table.**
* **Cards 2-9 score their face value, 10s,jacks, queens and kings score 10. An ace can be 1 or 11 at the player’s discretion, except if they have a blackjack.**
* **If the player has 21 after the deal (blackjack)**
  + **Add 1.5 x their stake to their winnings**
  + **Exclude them from the remainder of the hand**
* **If they have an ace and the other card is 2-9, ask if they want to play the hand as 1 or 11 + the value of the other card. If they have 2 aces, ask “2 or 12” as 22 would be too many.**
* **Once the player’s score is established, ask if they want another card.**
  + **If they they say no, keep them in the hand but don’t offer any more cards.**
  + **If they say yes, take another card from the chute and add it to the player’s score.**
  + **If it’s over 21 take their stake and it to the dealer’s winnings**
  + **If it’s 21, move on to the next player**
  + **If it’s under 21, ask if they want another card and repeat this step until**
    - **The player reaches 21**
    - **The player stops accepting cards**
    - **The player exceeds 21**
* **When simulating the other players’ hands**
  + **Don’t hit 18 or above**
  + **Always hit below 10**
  + **Decide at random whether to hit or stay for scores of 11 to 17 inclusive**
* **When all players hands are complete, reveal the dealer’s face-down card**
* **Dealer must play to 16 and stand on 17**
* **Add the stakes of all players with less than the dealer’s score to the dealer’s winnings.**
* **Let the stakes stand of any player with an equal score**
* **Add 1 x the player’s stake to any player with more than the dealer**
* **Ask the player to place a bet. The player can “cash out”, “sit out” or enter a bet amount >= £5**
* **If the player sits out, simulate a hand for the other players**

**PART B - BROWNFIELD**

**In Kata/CustomTypes/Demo, you’ll find a folder called Gamependium. This folder contains my implementation of the exercise where you had to set up a board for draughts, chess and backgammon.**

**My implementation makes occasional use of a couple of techniques that I haven’t shown you yet. You don’t need to use these techniques to complete this part of the assignment but you are free to copy what I’ve done if you want to. I’ll be covering these topics in December.**

**This assignment is about how well you can get to grips with an existing code base and extend its functionality. You’ll need to think about commonality between what you’re being asked to add and the existing code. Then you’ll need find out how the existing code implemented that and decide how to implement yours.**

**Specifically, this will mean whether to:**

* **Create another child of an abstract class**
* **Inherit from a concrete class**
* **Add properties or methods etc to an existing class**
* **Add overloads to existing methods and/or constructors**
* **Create new types or extend existing ones.**

**It’ll depend on what sub-task you’re trying to complete.**

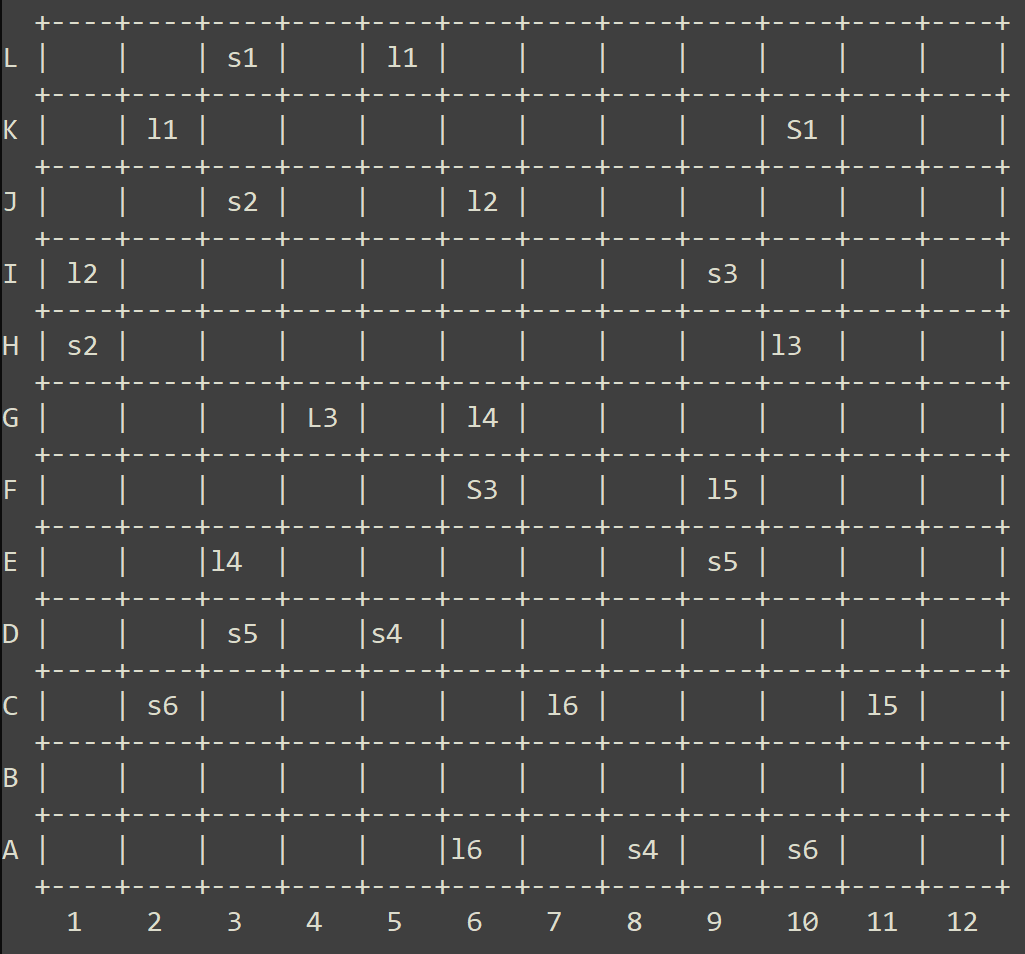
**Objective:**

* **Add Snakes and ladders to the Gamependium**

**Requirements:**

* **The board is 12 x 12**
* **You have to position the snakes and ladders randomly when the game is reset.**
* **There are 6 snakes and 6 ladders to position, varying in height and width between 3 and 9 squares for snakes and 4 and 10 squares for ladders.**
  + **E.g. The head of snake 1 might be on square F9 and its tail on B5. Hence the width of the snake is 4 and its height is 5.**
  + **Similarly the foot of a ladder might be A11 and the top come out on C10. So the ladder 2 squares wide by 3 squares tall. See snakes-n-ladders.json for an example representation but you should determine the dimensions of the snakes and ladders at random.**
  + **When generating the snakes and ladders ensure that:**
    - **There are no duplicates**
    - **The ends cannot share a square with another snake or ladder.**
    - **There must be 1 clear square to the left and to the right of a snake/ladder, except at the edge of the board when the clear square must be the next/prev square (depending on the direction of play)**
* **There can be more than 2 players**
* **the counters can be any colour but not black or white.**

**Here’s a sample board:**

****

**PART C - BONUS**

**Simulate a game of snakes and ladders.**

**How can you adapt the play simulation you wrote for blackjack into this? Consider providing the flexibility for your design to be used or extended to simulate the other games as well.**

**Don’t write anything that specifically references the other games. Just think about a level of abstraction that would mean your snakes and ladders simulator is an implementation of that. So, the other games could be simulated by providing their own implementation.**