**GO Paradigm**

The Go solution effectively demonstrates several functional programming features:

**First-Class Functions:**

Functions like ` `generateAllSymbols`, `shuffleSymbols`, `fillColumns`, `deductBalance`, `isAllSymbolsSame`, `calculateWinnings`, and `updatePlayerBalance` are treated as first-class functions. They are either passed as arguments, returned as values, or assigned to variables.

**Closures:**

Closures are used in the `getSlotMachineSpin` methods of `NormalMachine` and `JackPot`, capturing the `symbol\_count` and `symbol\_count\_large` variables.

**Immutability:**

The code demonstrates a form of immutability by creating a new instance of the `Player` struct when updating the balance in the `deductBalance` and `updatePlayerBalance` functions.

**Higher-Order Functions:**

Functions like `getSlotMachineSpin` and `checkWinnings` take other functions as parameters, making them higher-order functions.

**Pure Functions:**

Functions such as `generateAllSymbols`, `shuffleSymbols`, `isAllSymbolsSame`, `calculateWinnings`, and `updatePlayerBalance` exhibit characteristics of pure functions by relying only on their input parameters and producing consistent results without side effects.

**Declarative Style:**

The code emphasises a declarative style, focusing on what needs to be achieved rather than detailing step-by-step instructions. Functions are designed to express desired outcomes.

The Go code effectively employs functional programming concepts, showcasing a solid grasp of FP principles. Through first-class functions, closures, and immutability, the code emphasises modularity and readability. Higher-order functions, such as `getSlotMachineSpin` and `checkWinnings`, enhance flexibility by accepting and returning functions. The use of pure functions ensures predictable outcomes and simplifies testing. The declarative style of the code prioritises expressing desired outcomes over explicit steps, contributing to readability and maintainability. Overall, the Go code demonstrates a pragmatic application of functional programming in the development of a slot machine game, aligning with the principles of functional programming.