**Name:**

**Advanced Programming in Java**

**Lab Exercise 12.12.2024**

1. Write a program that simulates a slot machine. The player starts out with M tokens. The value of M is input into the program, and coins cost 25 cents each. For each play, the player can bet 1 to 4 coins. If the player enters 0 for the number of coins to bet, then the program stops playing. At the end of the game, the program displays the number of coins left and how much the player has won or lost in the dollar amount. There are three slots on the machine, and each slot will display one of three possible pieces: BELL, GRAPE, and CHERRY. When certain combinations appear on the slots, the machine will play the player. The payoff combinations are as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number | Combinations | | | Payoff |
| 1 | BELL | BELL | BELL | 10 |
| 2 | GRAPE | GRAPE | GRAPE | 7 |
| 3 | CHERRY | CHERRY | CHERRY | 5 |
| 4 | CHERRY | CHERRY | ---------- | 3 |
| 5 | CHERRY | ---------- | CHERRY | 3 |
| 6 | ---------- | CHERRY | CHERRY | 3 |
| 7 | CHERRY | ---------- | ---------- | 1 |
| 8 | ---------- | CHERRY | ---------- | 1 |
| 9 | ---------- | ---------- | CHERRY | 1 |

The symbol ---------- means any piece. If the player bets 4 coins and gets combination 5 for example, the machine pays the player 12 coins.

**Extra Problems**

1. **Exists a Number Higher?**

Write a function that returns true if there exists **at least one number** that is **larger than or equal to** n.

**Examples**

existsHigher([5, 3, 15, 22, 4], 10) ➞ true

existsHigher([1, 2, 3, 4, 5], 8) ➞ false

existsHigher([4, 3, 3, 3, 2, 2, 2], 4) ➞ true

existsHigher([], 5) ➞ false

**Notes**

Return false for an empty array [].

1. **Many Operators!**

Some basic arithmetic operators are +, -, \*, /, and %. In this challenge you will be given three parameters, num1, num2, and an operator. Use the operator on parameters num1 and num2.

**Examples**

operate(1, 2, "+") ➞ 3

// 1 + 2 = 3

operate(7, 10, "-") ➞ -3

// 7 - 10 = -3

operate(20, 10, "%") ➞ 0

// 20 % 10 = 0

**Notes**

There will be no division by zero.

1. **Capture the Rook**

Write a function that returns true if two rooks can attack each other, and false otherwise.

**Examples**

canCapture(["A8", "E8"]) ➞ true

canCapture(["A1", "B2"]) ➞ false

canCapture(["H4", "H3"]) ➞ true

canCapture(["F5", "C8"]) ➞ false

**Notes**

* Assume no blocking pieces.
* Two rooks can attack each other if they share the same row (letter) or column (number).