Math 482 - Programming Project (SPRING 2022)

- Minimal Obstacles in Instant Insanity

Name: Jong Won Lim

The Instant Insanity Puzzle consists of n slices that could be "pizza shaped" slices or "cube shaped" slices. The main difference between them is that the Pizza Instant Insanity has 5 faces, and the Cube Instant Insanity has 6 faces. When the slices are stacked in a tower, the top face and the bottom face of the Pizza slice are not being used, so 3 faces are used in the Pizza Instant Insanity. However, the Cube Instant Insanity uses all 6 faces, but one pair of faces is covered by the stacked slices, so identifying which pair should be covered to find the solution makes the puzzle more complex.

Part A-

There are 6 puzzles to be solved, and for each puzzle, the number generating function are different. Because of the modulus, some numbers were appearing more than 3 times in a puzzle, which creates a minimal obstacle of size 2. Therefore, the puzzles were forced to have one number(1 ~ 14) exactly 3 times, which may have a solution or not. The method that we used to fill the puzzle with correct inputs was making an array of size 42, and plugging in the number generated by the generator function, starting with $n = 1$. This is repeated until the array is filled with exactly 42 numbers. Then the inputs that we got are:

Puzzle 1: [12, 9, 7, 4, 2, 13, 11, 8, 6, 3, 1, 12, 10, 7, 5, 2, 14, 11, 9, 6, 4, 1, 13, 10, 7, 5, 2, 14, 11, 9, 6, 4, 1, 13, 10, 8, 5, 3, 14, 12, 8, 3]
Puzzle 2: [9, 4, 13, 8, 3, 12, 7, 2, 11, 6, 1, 10, 5, 14, 9, 4, 13, 8, 2, 11, 6, 1, 10, 5, 14, 9, 4, 13, 8, 3, 12, 7, 2, 11, 6, 1, 3, 12, 7, 10, 5, 14]
Puzzle 3: [11, 7, 4, 14, 10, 7, 3, 13, 10, 6, 2, 13, 9, 5, 2, 12, 8, 5, 1, 11, 8, 4, 14, 11, 7, 3, 14, 10, 6, 3, 13, 9, 6, 2, 12, 9, 5, 1, 12, 8, 1, 4]
Puzzle 4: [3, 5, 8, 10, 13, 1, 3, 6, 8, 11, 13, 2, 4, 6, 9, 11, 14, 2, 5, 7, 9, 12, 14, 3, 5, 8, 10, 12, 1, 6, 11, 13, 1, 4, 9, 14, 2, 4, 7, 12, 7, 10]
Puzzle 5: [4, 8, 12, 2, 6, 10, 14, 4, 8, 11, 1, 5, 9, 13, 3, 7, 11, 1, 4, 8, 12, 2, 6, 10, 14, 11, 1, 5, 9, 13, 3, 7, 12, 2, 6, 10, 14, 5, 9, 13, 3, 7]
Puzzle 6: [11, 8, 5, 2, 13, 10, 7, 4, 1, 12, 9, 6, 3, 14, 11, 8, 4, 1, 12, 9, 6, 3, 14, 11, 8, 5, 2, 13, 10, 7, 4, 1, 12, 5, 2, 13, 10, 7, 9, 6, 3, 14]

The first column of the tower(the first number of the three numbers that a slice has), were located at array index [$3n$ - 3], the second column on array index [$3n$ -2], and the third column on array index [$3n$ - 1]. The function Rotating is changing the array index [$3n$-3] to array index [$3n$ - 2], array index [$3n$ -2] to array index [$3n$ -1] and array index[$3n$ - 1] to array index [$3n$ - 3]. The function Flipping is changing the array index [$3n$ - 2] to array index [$3n$ -3], which produces a lot more combinations of numbers. The algorithm we used to solve the puzzle was basically locating the slices that have similar numbers at the bottom, and minimizing the possible number of cases, because with this method the puzzle could be solved manually in most cases.

Using this method, we found that there exists "No Solution" for Puzzle 1, but there exists a Flip Solution which is flipping the Slice #14 (12, 8, 3) to (8, 12, 3). The Flip Solution for Puzzle 1 looks like:

(8 12 3) : Flipped.

(5 3 14)

(13 10 8)

(6 4 1)

(14 11 9)

(7 5 2)

(1 13 10)

(9 6 4)

(2 14 11)

(10 7 5)

(3 1 12)

(11 8 6)

(4 2 13)

(12 9 7)

The Puzzle 2, had a Solution which is:

(10 5 14)

(3 12 7)

(11 6 1)

(12 7 2)

(13 8 3)

(14 9 4)

(1 10 5)

(2 11 6)

(4 13 8)

(5 14 9)

(6 1 10)

(7 2 11)

(8 3 12)

(9 4 13)

Puzzle 3, No Solution, but it had Flip Solution:

(8 4 1) Flipped

(5 1 12)

(2 12 9)

(13 9 6)

(10 6 3)

(7 3 14)

(4 14 11)

(1 11 8)

(12 8 5)

(9 5 2)

(6 2 13)

(3 13 10)

(14 10 7)

(11 7 4)

Puzzle 4, No Solution, but it had Flip Solution:	Puzzle 5, had a Solution
(10 12 7)	(3 7 13)
(7 2 4)	(5 9 14)
(14 4 9)	(10 2 6)
(13 1 11)	(12 3 7)
(6 12 1)	(13 5 9)
(5 8 10)	(1 14 11)
(12 14 3)	(6 10 2)
(9 7 5) Flipped	(8 12 4)
(11 14 2)	(7 11 1)
(4 6 9)	(9 13 3)
(2 11 13)	(11 1 5)
(8 3 6)	(14 4 8)
(1 13 10)	(2 6 10)
(3 5 8)	(4 8 12)

Puzzle 6 had a Solution:

(6 3 14)

(10 7 9)

(5 2 13)

(1 12 4)

(13 10 7)

(8 5 2)

(3 14 11)

(9 6 12)

(4 1 8)

(14 11 3)

(12 9 6)

(7 4 1)

(2 13 10)

(11 8 5)

The Puzzle 2, 5, 6 had a Solution, and Puzzle 1, 3, 4 had No Solution. However, when the restriction that the slice can't be flipped was removed, all the puzzles had a solution. This clearly shows that flipping the slice makes more possible combinations of the numbers, so that helps remove the Minimal Obstacle of the puzzle.

Part B-

The Minimal Obstacle of a Puzzle is the set of slices that is never the part of a Solution. The process that we used to check the size of the Minimal Obstacle that we constructed is finding a smaller Minimal Obstacle that could possibly exist in the Puzzle. For example, a M.O of size 6 has No Solution only when those 6 slices are stacked, which means that the stack of any of those 2 to 5 slices should have at least one solution.

M.O Size 2:	M.O Size 3:	M.O Size 4:
[1 3 2]	[3 4 1]	[4 3 2]
[1 2 3]	[2 3 1]	[2 4 1]
	[1 2 3]	[3 1 2]
		[1 3 4]

M.O Size 5:	M.O Size 6:	M.O Size 7(Extra):
[7 6 6]	[6 3 7]	[10 4 1]
[2 1 5]	[5 1 8]	[6 11 3]
[4 6 1]	[9 6 5]	[5 3 9]
[5 7 4]	[2 5 7]	[4 5 8]
[1 5 7]	[3 4 2]	[2 4 5]
	[1 2 6]	[7 6 2]
		[1 2 3]