Hands-on Eggsercises

Lecture 26

Today:

Egg Russian Roulette

https://www.youtube.com/watch?v=vVhNc0

Overview

- One carton of 12 eggs
- 8 are hard boiled, 4 are raw
- The eggs are shuffled, randomized in carton
- Players alternate choosing an egg at random and crack it on their head
- First to select 2 raw eggs loses

Code Read

- Play Rock-Paper-Scissors with a neighbor
- Winner reads and eggsplains Egg.ts
- Not-quite-winner eggsamines and discusses Carton.ts
- Check-in on PollEv.com/comp110 when you have a sense of what both Egg.ts and Carton.ts contain

Code Walk

- Let's review a few files together, then we're ready to begin
- EggRoulette.ts - where we'll implement the game logic
- app-script.ts - the starting point of our app
- Graphics.ts - the code for drawing the carton, eggs, and scoreboard
- First, let's reveal the content of the eggs while we're working on the game

Hands-on #1) Boiling the Eggs

- At TODO #1.1, in EggRoulette.ts, define a method named **boilEggs**
  1. It has no parameters and returns nothing
  2. It should loop through all but the last 4 elements of the **carton** property's **eggs** array
  3. It should call the **boil** method on those Egg objects

- At TODO #1.2, in the constructor, call the **boiledEggs** method on **this**

- You should see 8 of the carton's Eggs become "hard boiled"

- Check-in when you think you've got it or are completely stuck.

```
boilEggs = (): void {
    for (let i = 0; i < this.carton.eggs.length - 4; i++) {
        let egg = this.carton.eggs[i];
        egg.boil();
    }
}
```

Hands-on #2) Picking an Egg

- The **pick** Method is called from the Graphics code each time you click an Egg. The parameter **index** refers to the index of the Egg clicked in Carton's Egg array.

- At TODO #2, declare a variable named **egg**, of type Egg, and assign it a reference to the Egg at **index** in the **carton** property's **eggs** array. It will refer to the Egg object that was clicked

- If **egg** is **not cracked**, then
  1. crack it (by calling its crack method)
  2. if egg is raw, increment the property ***scores*** array at index ***this.currentPlayer()*** by 1
  3. increment the turn by 1

- The first time you click on any egg, score should go up by 1 if raw 0 if not. Second time you click on any egg, because it's "cracked", nothing happens. Check-in on Poll-Ev!

```
pick = (index: number): void => {
    // TODO #2: pick method - logic for picking an Egg
    let egg: Egg = this.carton.eggs[index];
    if (!egg.isCracked) {
        egg.crack();
        if (egg.isRaw) {
            this.scores[this.currentPlayer()]++;
        }
        this.turn++;
    }
}
```

Hands-on #3) Determining a Winner

- The first player to have a score of 2 (2 raw eggs picked) loses, so the other player is the winner

- In the winner() method, implement the following logic:
  - If player 0's score (scores array index 0) is 2, then return (player) 1
  - Otherwise, if player 1's score (scores array index 1) is 2, then return (player) 0
  - Otherwise, return -1

- Now, when a player picks 2 raw eggs, the winner will show. Check-in when your conditional logic is working.

```
winner = (): number => {
    // TODO #3: Logic for determining the winner
    if (this.scores[0] === 2) {
        return 1;
    } else if (this.scores[1] === 2) {
        return 0;
    } else {
        return -1;
    }
}
```

Hands-on #4) Shuffling the Eggs

1. At TODO 4.1, define a method named **shuffleEggs**, with no parameters, that returns nothing

2. It should loop through every **Egg** in the **carton**'s array of **eggs**, using **i** as the index counter variable

3. In the loop, choose a random index and assign it to a local variable **random**
   - Generate a random index:
     `Math.floor(this.carton.eggs.length * Math.random())`

4. It should swap the Egg at index **i** with the Egg at index **random**
   - What's the key to swapping two values in memory? Hint: it's a variable.

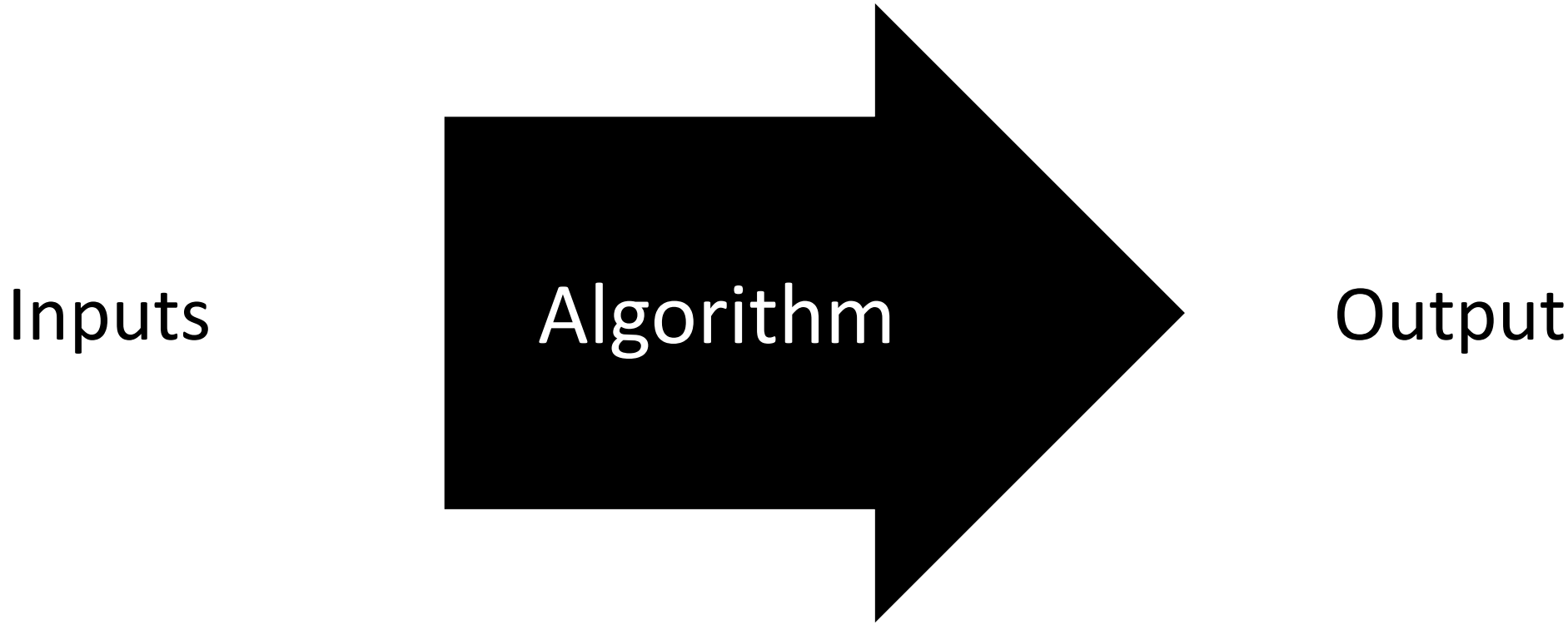
5. **Call this function in the constructor at TODO 4.2**. Save & refresh a few times. You should see your raw eggs repositioned at random.

- Back in **app-script.ts**, change **revealEggState** to be false and try playing your neighbor. Check-in when complete!

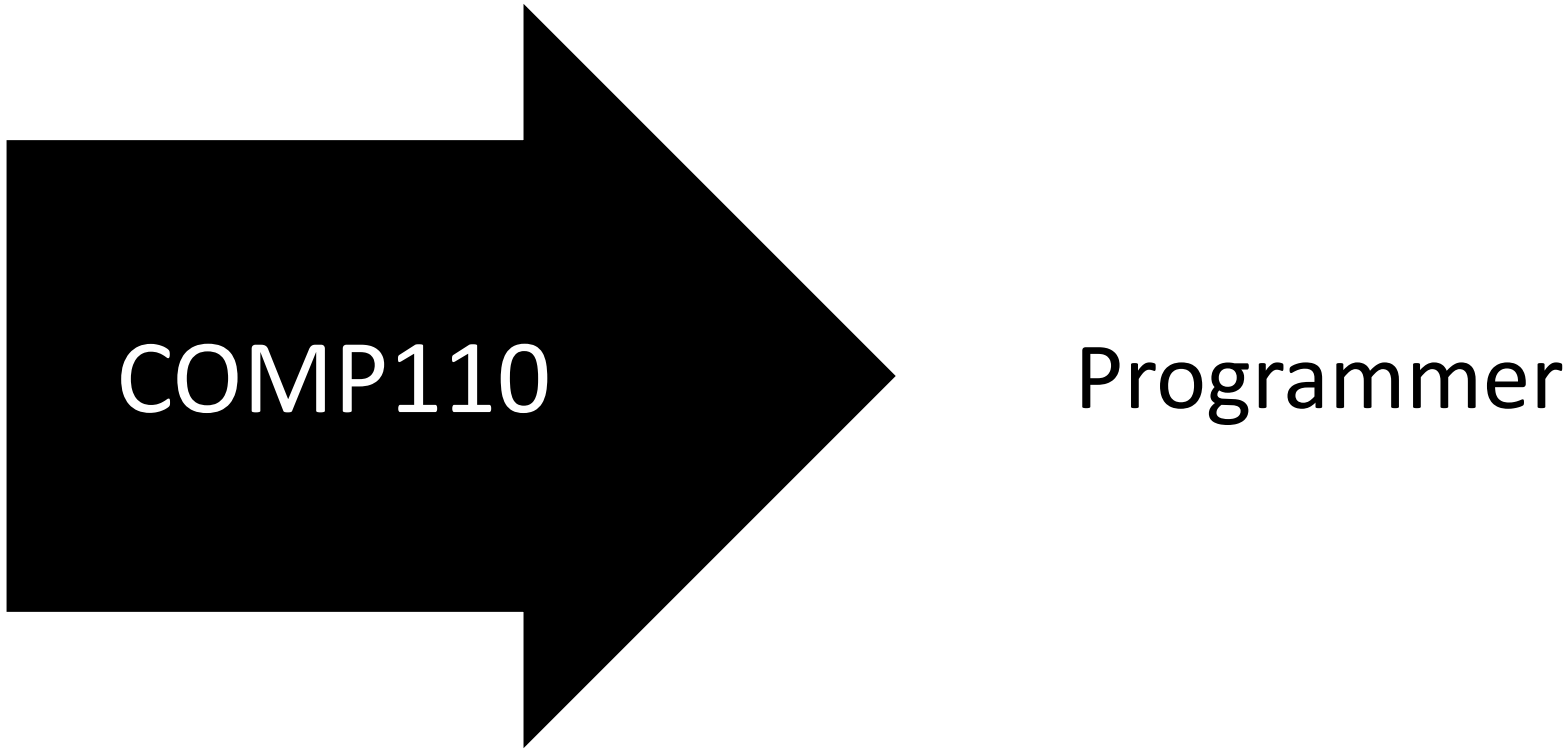
```
shuffleEggs = (): void => {
    let size: number = this.carton.eggs.length;
    for (let i: number = 0; i < size; i++) {
        let random: number = Math.floor(size * Math.random());
        let temp: Egg = this.carton.eggs[i];
        this.carton.eggs[i] = this.carton.eggs[random];
        this.carton.eggs[random] = temp;
    }
}
```

Let's play.

The Fundamental Pattern



The Fundamental Pattern




STRAIGHT
OUTTA
COMP 110