# Arrays and Testing Continued

Lecture 7

# Announcements

- WS1 - Posted and Due Monday 9/24 at 11:59pm
- PS1 - Out by tomorrow and Due Wednesday 9/26 at 11:59pm
- Q2 - Postponed to Thursday 9/27
  - Taking with ARS? Be sure to reschedule today!

# 0. Fill in the blanks...

```
export let main = async () => {
   let a: ___A___ = ___B___();
   let b: ___C___ = ___D___(["hi"]);
};

let y = (s: ___E___): ___F___ => {
   return s.length;
};

let z = (): ___G___ => {
   return "hi";
};
```

1. Given the function definitions to the left, what does the following expression evaluate to?

```
h(f(["a", "b", "c"]))
```

```
let f = (s: string[]): number => {
    return s.length;
};

let h = (n: number): string[] => {
    let s = "h" + n;
    let a = [];
    for (let i = 0; i < n; i++) {
        a[i] = s;
    }
    return a;
};
```

# 2. What is the printed output when **`main`** runs?

```
export let main = async () => {
   print(double(3));
};

let double = (x: number): number => {
   return x * 2;
};

let test = (s: string, e: number, a: number): void => {
   if (e === a) {
      print("PASS: " + s);
   } else {
      print("FAIL: " + s);
   }
};
```

# 3. What is the printed output when **`main`** runs?

```
export let main = async () => {
   test("double(3)", 6, double(3));
};

let double = (x: number): number => {
   return x ** 2;
};

let test = (s: string, e: number, a: number): void => {
   if (e === a) {
      print("PASS: " + s);
   } else {
      print("FAIL: " + s);
   }
};
```

# 4. Suppose you're writing a **fillRange** function

- Its signature is:

```
fillRange(low: number, high: number): number[]
```

- Its purpose is to generate an array of consecutive integers increasing from low and ending with high, inclusive.

- Select the test case (input parameters and expected return value) which you believe is the best example of an **edge case**.

# Hands-on: Write Tests for **`fillRange`**

- The function generates an array of numbers from low to high, inclusive.

- One example use case:
  **`fillRange(0, 2)`** expects a return value of **`[0, 1, 2]`**

- In **`test-practice-app.ts`**:
  1. Write a test for another **use case** you can imagine.
  2. Write a test case for an **edge case**.

- Once you have three failing tests, check-in on pollev.com/compunc

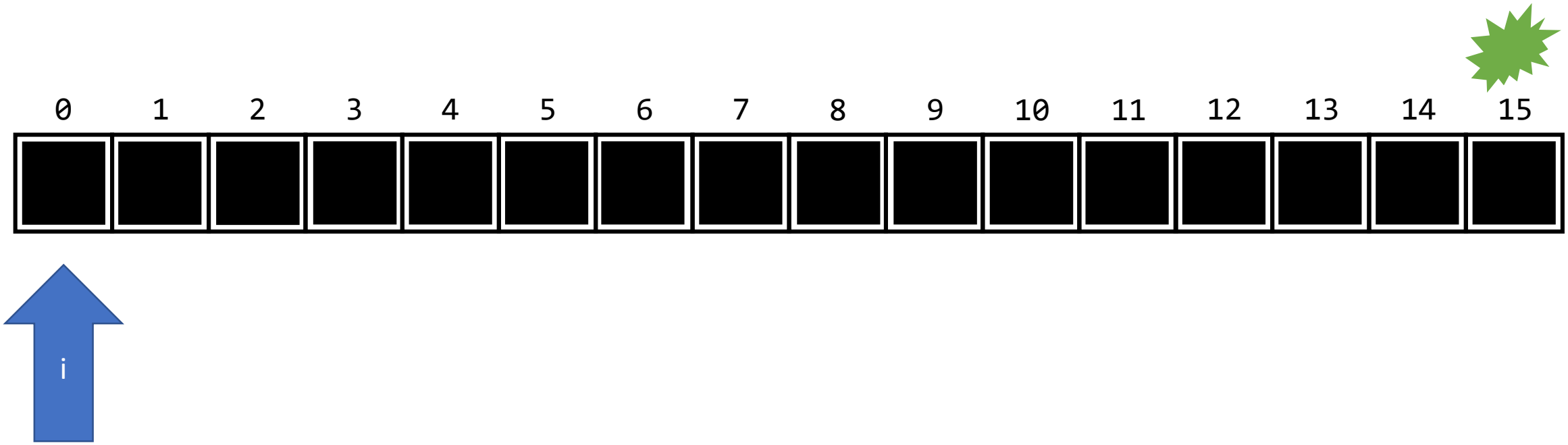
# Hands-on: Implement **`fillRange`**

1. Open **`array-functions.ts`**
2. Hint #1: Look to `fillZeros` as a starting point.
3. Hint #2: What should your loop's counting variable's initial value be?
4. Hint #3: You can append to an array named a with: **`a[a.length] = <num>`**
5. **Check-in once you have your tests passing** and a working **`fillRange`**.
6. Done? Try improving with a version that rounds down decimals and still works.

# 5. What does the following expression evaluate to: **`foo([4, 8, 16], 4)`**

```
let foo = (a: number[], n: number): number => {
   for (let i = 0; i < a.length; i++) {
      if (a[i] > n) {
         return a[i];
      }
   }
   return -1;
};
```

# The **Linear Search** Algorithm




Does the word “Yes” exist in this array of Strings?

# The **`indexOf`** Function

- Given an array of number values, and a number to search for:

1. Return the index of the number's first occurrence in the array.
2. When the number does not exist in the array, return -1.

- Use Cases:

```
indexOf([1, 2, 3], 1) should return 0
indexOf([1, 2, 3], 2) should return 1
```

- Edge Cases:

```
indexOf([1, 2, 3], 4) should return -1
indexOf([1, 2, 2], 2) should return 1
```

# Hands-on: Implementation of **`indexOf`**

- In array-functions.ts

1. Loop through every element of array **`a`**, starting from index **`0`**
2. If an element is equal to **`n`**, then return its **index**
3. Otherwise, return **`-1`**.

- Check-in when you have the test cases passing. pollev.com/compunc

# The world's 2nd worst magic trick…