

# CMPSC 302 WEB DEVELOPMENT



## Checking your work

```
const validatePattern = () => {
 if(choices.length !== pattern.length) return false;
 for(var i = 0; i < pattern.length; i++)
   if(pattern[i] !== choices[i]) return false;
 return true;
                   If we've survived all of our
                   tests, return that it's a valid
                   match!
```

If the two patterns aren't the same length, then something's wrong!

| false is a "Boolean" type: it's either "yes" or "no" |
| A different kind of for statement that checks every entry in both arrays to see if

they're the same



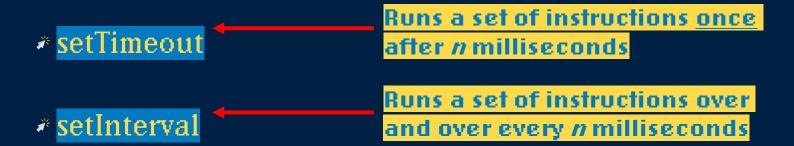
#### A It's all in the timing

- To play our game, we rely on a "give and take" approach to time:
  - Page-generated pattern
  - User-entered response
  - Delays proportional to user "level"
- \*Here, our assumptions are:
  - Every "level" should add 5 seconds to user response time
    - That's 5 seconds for every 1 additional entry in the pattern



### A tale of two procedures

We're going to negotiate between:





```
setTimeout(() ⇒ {
 console.log("l second!");
}, 1000);
setInterval(() ⇒ {
 console.log("1 second!");
}, 1000);
```



```
const oneSecond = setInterval(() => {
    console.log("1 second!");
}, 1000);
```

// I want to stop this!
clearInterval(oneSecond);



## Function junction

We've seen the "anonymous" way of making functions happen:

 () ⇒ {
 // Stuff here
 }

\*There's another way, though which is named function() {
// Stuff