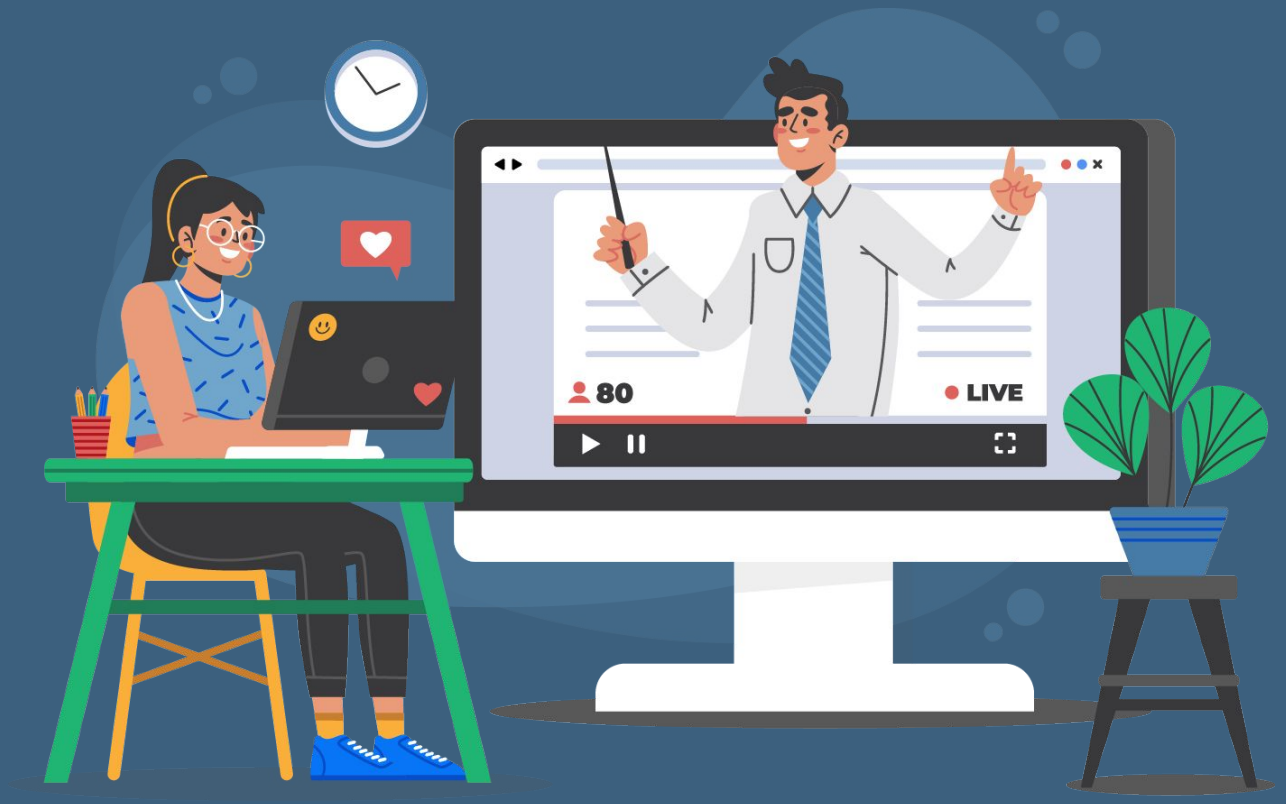
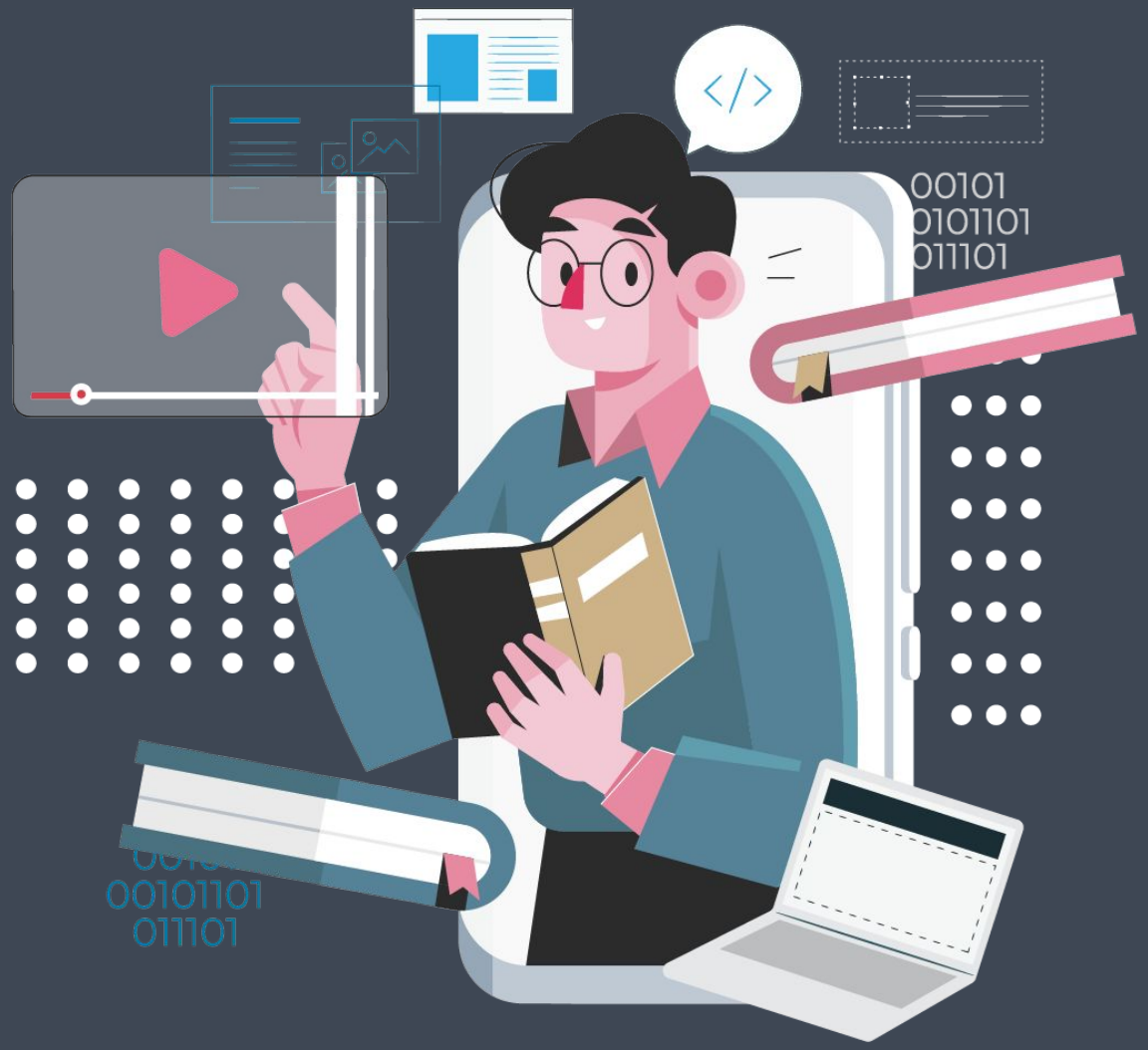


</talentlabs>

Lecture 3
String Algorithms (String as Arrays)



</talentlabs>

Agenga

- Treating Strings as Arrays
- Strings Question 1 - Finding Substring
- Strings Question 2- Reverse Word Order
- Strings Question 3 - Anagram

</talentlabs>

Strings as Array

Strings as Array

- Strings are actually stored as array or treated as array in a lot of programming languages
- i.e. you are access each of the letters using index

```
const a = "Hello World!"
const a = [“H”, “e”, “l”, …]
console.log(a[1])                // Print out the letter “e”
console.log(a[6])                // Print out the letter “W”
```

 </talentlabs>

Handling Strings Algorithm Questions

- Given that strings can be treated as arrays, you can actually consider most of the strings questions as array questions
- Still the same techniques:
  - Temporary Variables
  - Looping Twice
  - Nested Loops
  - Sort First

Strings Question 1 - Finding Substring

Finding Substring

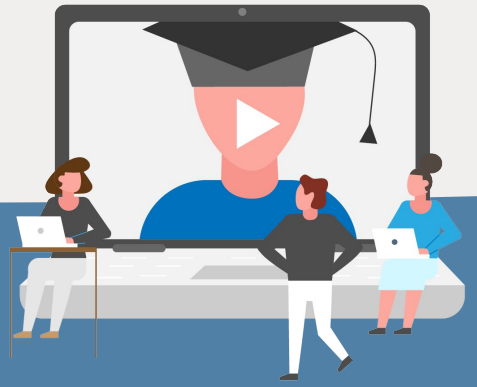
How can you check if a substring exists in a string? (How to implement the indexOf() function?)

Example 1 (Found case):
Input String: "Hello", Target: "lo"
Results: 3

Example 2 (Not found case):
Input Array: "Good Evening”, Target: "app"
Results: -1

</talentlabs>

What is your solution?



</talentlabs>

What is your solution?

Hint: Using 2 loops, nested loops

</talentlabs>

Example Algorithm

1. Use a for loop to loop through the input array
2. For each letter, use a second loop to loop thru the next couple letter. Check each letter against the target string. Go on and on until the whole “target string” is matched, or one of the letter it not matching
3. If the whole “target string” is matched, return true.
4. If the outer loop finished, but still no fully matched substring, return false

Part 1 - Creating Outer Loop

Very simple, just creating a simple for loop to loop through whole string.

```
originalString = "Hello"

for (let i = 0; i<originalString.length; i++){
  // Leave for Step 2


}
```


Break

Ending the loop immediately

```
1 for (let i = 0; i < 5; i++) {
2   if (i === 3) {
3     break;
4   }
5   console.log("It's now " + i);
6 }
```

Jump to here when the break line is hit

Console Output

```
It's now 0
It's now 1
It's now 2
```

 </talentlabs>

Continue

Ending the current iteration, but not ending the loop

Jump to here when `continue` is hit

```
1 for (let i = 0; i < 5; i++) {
2   if (i === 3) {
3     continue;
4   }
5   console.log("It's now " + i);
6 }
```

Console Output

```
It's now 0
It's now 1
It's now 2
It's now 4
```

 </talentlabs>

Part 2 - Creating the Inner Loop

In this step, we need to create an inner loop that would help us in checking if the next few letters (including the current letter) matched with the inner loop.

```
originalString = "Hello"
targetString = "lo"

for (let i = 0; i<originalString.length; i++){
  // Leave for Step 2
  for (let j = 0; j < targetString.length; j++){
   if  (originalString[i + j] !== targetString[j]){
     break;
   }
   else if (originalString[i + j] === targetString[j] && j ===targetString.length -1){
     console.log("Found at position " + i)
   }
  }
}
```

Case 1: Not Match, and end this iteration

Case 2: Match and it's the last character

Input Array: Hello
Target: lo

```
originalString = "Hello"
targetString = "lo"

for (let i = 0; i<originalString.length; i++){
  // Leave for Step 2
  for (let j = 0; j < targetString.length; j++){
   if  (originalString[i + j] !== targetString[j]){
     break;
   }
   else if (originalString[i + j] === targetString[j] &&
j === targetString.length -1){
     console.log("Found at position " + i)
   }
  }
}
```

Part 3 - Optimization

Double check the logic and see if there are anything your missed or can be removed?

```
originalString = "Hello"
targetString = "lo"

for (let i = 0; i<originalString.length; i++){
  // Leave for Step 2
  for (let j = 0; j < targetString.length; j++){
   if  (originalString[i + j] !== targetString[j]){
     break;
   }
   else if (originalString[i + j] === targetString[j] && j ===targetString.length -1){
     console.log("Found at position " + i)
   }
  }
}
```

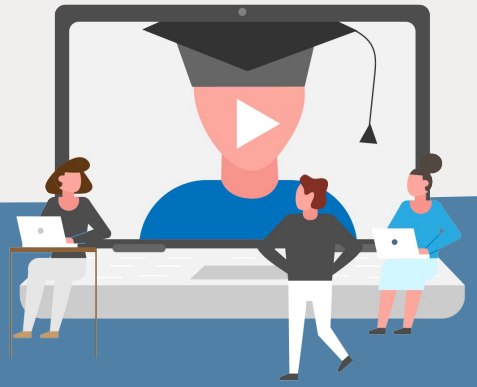

Part 3 - Optimization

Double check the logic and see if there are anything your missed or can be removed?

```
originalString = "Hello"
targetString = "lo"

for (let i = 0; i<originalString.length; i++){
  // Leave for Step 2
  for (let j = 0; j < targetString.length; j++){
   if  (originalString[i + j] !== targetString[j]){
     break;
   }
   else if (originalString[i + j] === targetString[j] && j ===targetString.length -1){
     console.log("Found at position " + i)
   }
  }
}
```


Techniques that we used

1. Nested Loops



</talentlabs>

Strings Question 2 - Reverse Word Order

Reverse Word Order

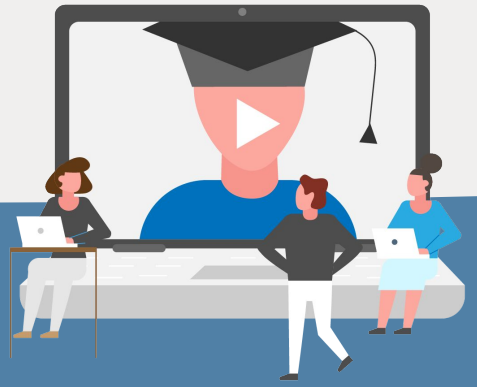
How can you reverse the word order in a string? (Implement the array.reverse() function for string)

Example 1:
Input String: Talentlabs, Output: sbaltnelaT

Example 2:
Input Array: Hello, Output: olleH

 </talentlabs>

What is your solution?



</talentlabs>

What is your solution?

Hint: Use a loop with temp variable

 </talentlabs>

Example Algorithm

1. Use a for loop to loop thru the inputString **from the end** of the array
2. For each letter, append it to a temp string
3. Return temp string as result

Assignment Question 1

Part 1 - Loop Thru the Array

First, we need to loop through the array

```
const inputString = "Talentlabs"

let result = ""
for (let i = inputString.length-1; i >= 0; i--){
  result += inputString[i]
}

console.log(result)
```

count down instead of count up

Start from the end of the array

 </talentlabs>

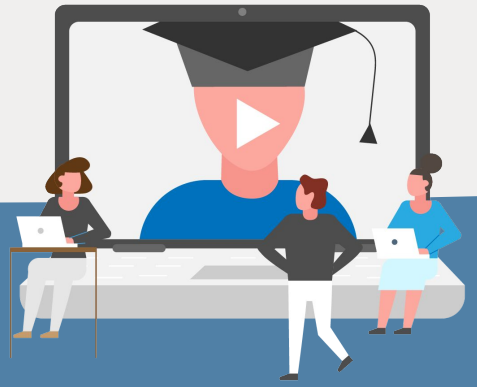
```
const inputString = "Talentlabs"

let result = ""
for (let i = inputString.length-1; i >= 0; i--){
  result += inputString[i]
}

console.log(result)
```

i: ~~9~~ ~~8~~ ~~7~~ ~~6~~ ~~5~~ ~~4~~ ~~3~~ ~~2~~ ~~1~~ ~~0~~ -1

result: sbaltnelaT

Techniques that we used

1. Loop
2. Temp Variable



</talentlabs>

Strings Question 3 - Anagram

Anagram

How to check if two strings are anagram to each other?

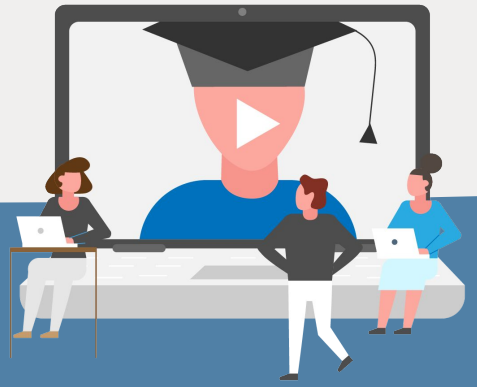
Definition of anagram: words with same letters, but different word.
(e.g. Paris/Pairs are anagram, arc/car are anagram)

Example 1 (Found case):
Input String 1: Paris, Input String 2: Pairs
Result: true

Example 2 (Not found case):
Input String 1: Hello, Input String 2: Bye
Result: false

What is your solution?

abb
ba



</talentlabs>

What is your solution?

Hint: Use sorting

</talentlabs>

Example Algorithm

1. Convert the two strings into array by using “split()” function
2. Sort the two arrays using the “sort()” function
3. Convert the two arrays back to strings
4. Compare the two strings and see they the two strings equals to each other

Revision - split()

```
1  const inputString = "Hello"
2
3  const convertedToArray = inputString.split("")
4  console.log(convertedToArray)
```

Console Output:

```
[ 'H', 'e', 'l', 'l', 'o' ]
```


Revision - sort()

```
1  const inputArray = ["a", "c", "b", "f", "z", "v"]
2
3  const sortedArray = inputArray.sort()
4  console.log(sortedArray)
```

Console Output:

```
[ 'a', 'b', 'c', 'f', 'v', 'z' ]
```


Revision - join()

```
1  const inputArray = ["a", "c", "b", "f", "z", "v"]
2
3  const joinedString = inputArray.join("")
4  console.log(joinedString)
```

Console Output:

```
acbfzv
```


</talentlabs>

Part 1 - Convert string to array

Convert the string to array so we can use the sort function later.

```
const inputString1 = "Paris"
const inputString2 = "Pairs"

stringInArray1 = inputString1.split("")
stringInArray2 = inputString2.split("")
```


</talentlabs>

Part 2 - Sort the two arrays

Sort the arrays so we can compare the two arrays compositions.

```
const inputString1 = "Paris"
const inputString2 = "Pairs"

stringInArray1 = inputString1.split("")
stringInArray2 = inputString2.split("")

const sortedArrar1 = stringInArray1.sort()
const sortedArrar2 = stringInArray2.sort()
```


</talentlabs>

Part 3 - Compare the sorted array

Join the sorted array back to a string, and compare if the two strings are the same.

```
const inputString1 = "Paris"
const inputString2 = "Pairs"

stringInArray1 = inputString1.split("")
stringInArray2 = inputString2.split("")

const sortedArray1 = stringInArray1.sort()
const sortedArrar2 = stringInArray2.sort()

if (sortedArray1.join("") === sortedArray2.join("")){
  console.log("Anagram")
}else {
  console.log("Not Anagram")
}
```

 </talentlabs>

```
const inputString1 = "Paris"
const inputString2 = "Pairs"

stringInArray1 = inputString1.split("")
stringInArray2 = inputString2.split("")

const sortedArray1 = stringInArray1.sort()
const sortedArrar2 = stringInArray2.sort()

if (sortedArray1.join("") === sortedArray2.join("")){
  console.log("Anagram")
}else {
  console.log("Not Anagram")
}
```

Input: abb, ba

Step 1, after split:
[a, b, b]
[b, a]

Step 2, Sorting:
a, b, b
a, b

Step 3 Join back:
abb
ab

Anagram 2.0

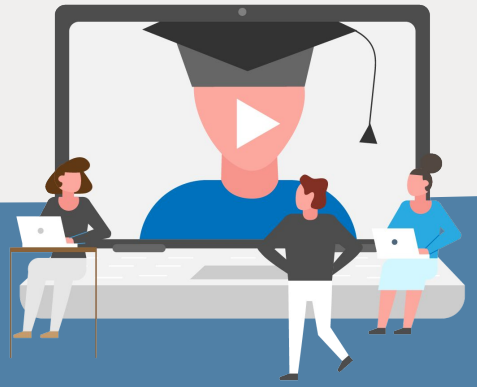
How to ignore cases when checking for anagram?

Example 1 (Found case):
Input String 1: Paris, Input String 2: pairs
Results: true

Example 2 (Not found case):
Input String 1: Paris, Input String 2: paris
Results: false (because they are the same word)

</talentlabs>

What is missing for our solution 1?



</talentlabs>

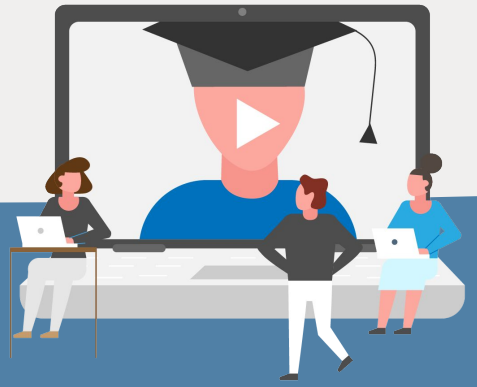
What is missing for our solution 1?

1. ignore cases
2. check if the two words are actually the same word

Solution:

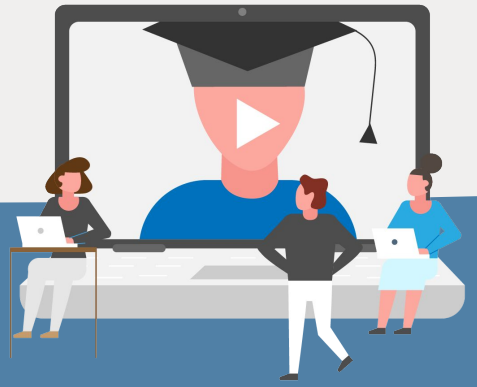
Convert the string to lowercase letters:

```
let str = "Hello World!";
str.toLowerCase();
```



Techniques that we used

1. Sorting



</talentlabs>