1. **A non-empty array A consisting of N integers is given. The unique number is the number that occurs exactly once in array A. For example, the following array A: A[0] = 4 A[1] = 10 A[2] = 5 A[3] = 4 A[4] = 2 A[5] = 10 contains two unique numbers (5 and 2). You should find the first unique number in A. In other words, find the unique number with the lowest position in A. For above example, 5 is in second position. The function should return −1 if there are no unique numbers in A.**

function uniqueNumber(arr) {

const num = arr.find(num => arr.filter(x => x === num).length === 1)

return num ?? -1;

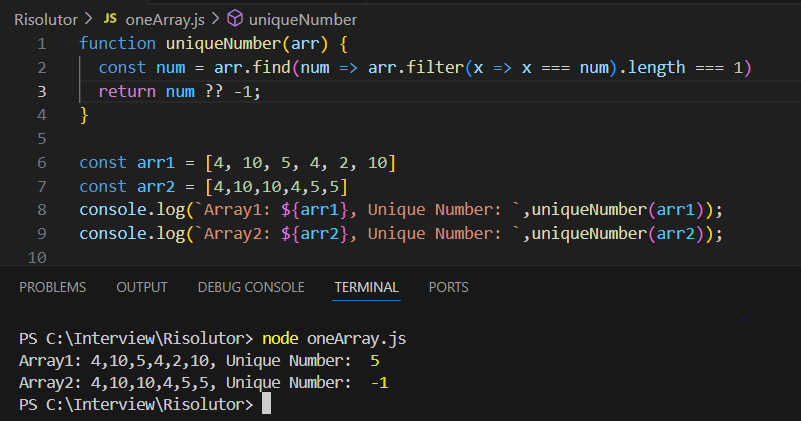
}

const arr1 = [4, 10, 5, 4, 2, 10]

const arr2 = [4,10,10,4,5,5]

console.log(`Array1: ${arr1}, Unique Number: `,uniqueNumber(arr1));

console.log(`Array2: ${arr2}, Unique Number: `,uniqueNumber(arr2));



**2) Write a function:**

**function solution(S);**

**that, given a string S, returns the index (counting from 0) of a character such that the part of the string to the left of that character is a reversal of the part of the string to its right. The function should return −1 if no such index exists. Note: reversing an empty string (i.e. a string whose length is zero) gives an empty string.  For example, given a string: “racecar” the function should return 3, because the substring to the left of the character “e” at index 3 is “rac” and the one to the right is “car”.**

function solution(S) {

    if (S === " " || S === "") return " ";

    return S.split("").findIndex((\_, i) =>

        S.substring(0, i) === S.substring(i + 1).split("").reverse().join("")

    );

}

const str1 = "racecar";

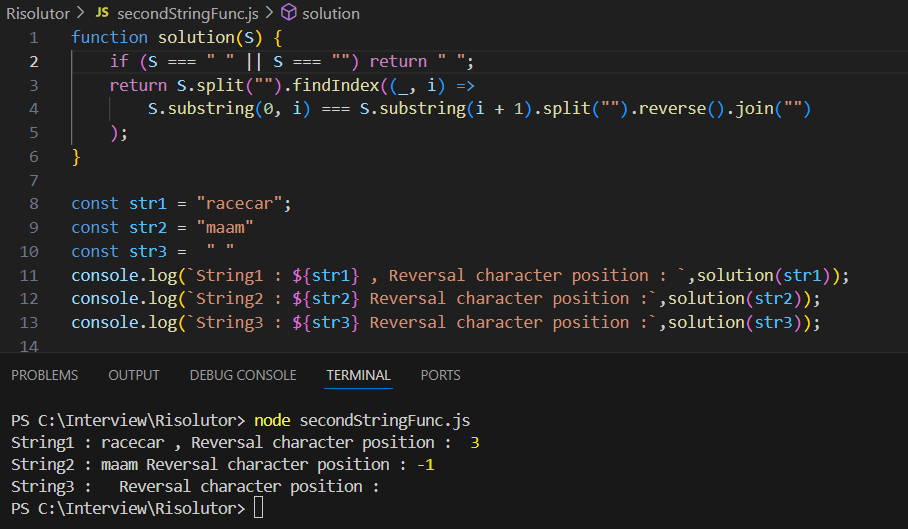
const str2 = "maam"

const str3 =  " "

console.log(`String1 : ${str1} , Reversal character position : `,solution(str1));

console.log(`String2 : ${str2} Reversal character position :`,solution(str2));

console.log(`String3 : ${str3} Reversal character position :`,solution(str3));



**3) 1. There should be a like button:**

**The content of the like button should be in the following format: &quot;Like | 100&quot;, where 100 is the total number of likes. It should have a &quot;like-button&quot; class. Wrap the number of likes in a span with a &quot;likes-counter&quot; class. The initial number of likes in the counter should be 100.**

**2. Users can add a like. By clicking the button:**

**The number of likes should increase by one. Like button should have “liked” class in addition to the “like-button” class (you can use the classnames tool for that).**

**3. Users can undo their like by clicking again on the button:**

**The counter should decrease by one.”liked” class should be removed.**

**App.js**

import React, { useState, useEffect } from "react";

import classNames from "classnames";

import "./App.css";

const App = () => {

const [likes, setLikes] = useState(100);

const [liked, setLiked] = useState(false);

const [click, setClick] = useState(false);

const toggleLike = () => {

setLiked(!liked);

setLikes(likes + (liked ? -1 : 1));

setClick(true);

};

useEffect(() => {

if (click) {

const timeout = setTimeout(() => setClick(false), 200);

return () => clearTimeout(timeout);

}

}, [click]);

return (

<div className="box">

<button

className={classNames("like-button", { liked, "text-blue": liked, "text-black": !liked, "bg-dark-gray": click })}

onClick={toggleLike}

>

Like | <span className="likes-counter">{likes}</span>

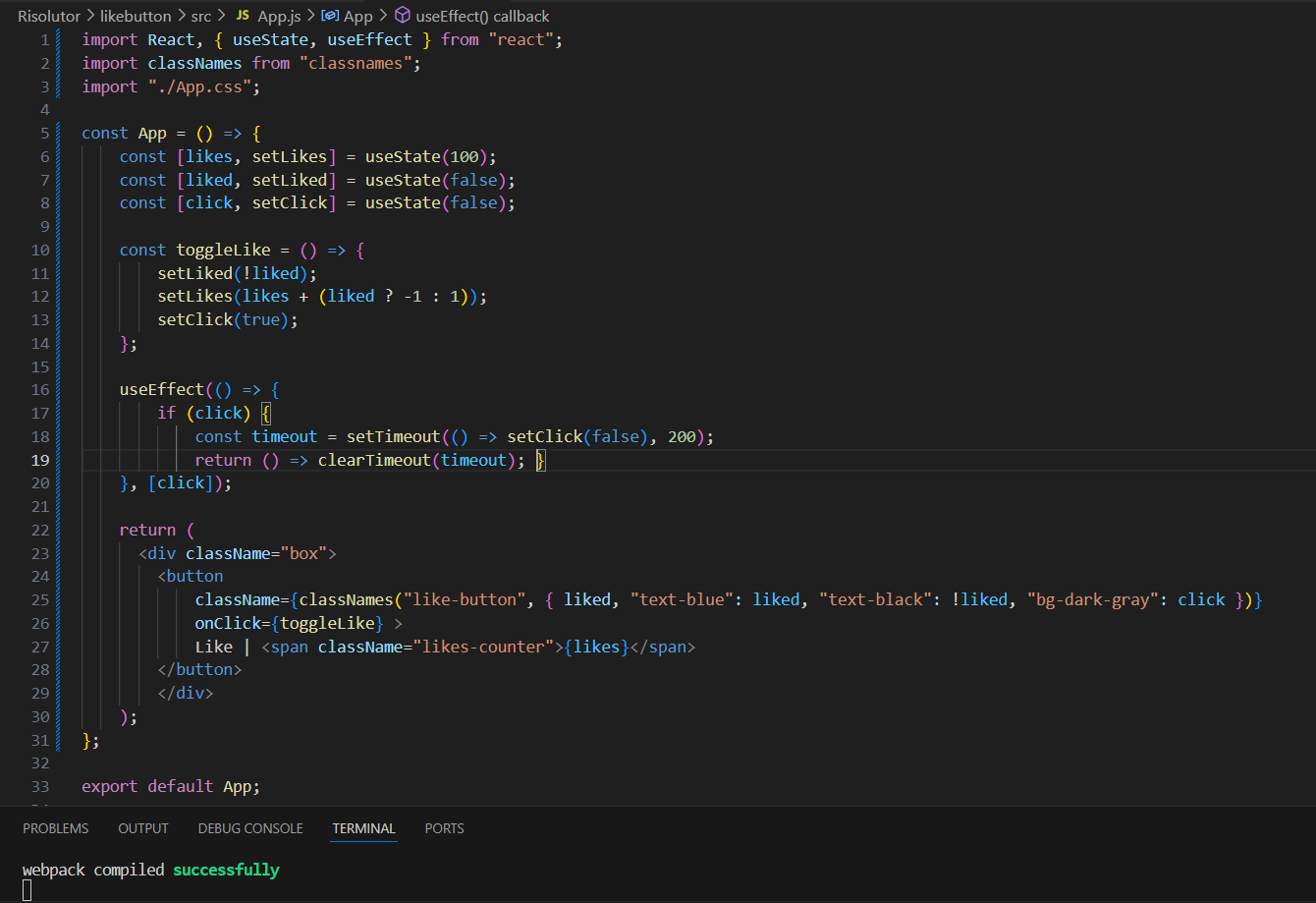
</button>

</div>

);

};

export default App;



**App.css**

.like-button {

font-size: 25px;

padding: 10px;

cursor: pointer;

border: 3px solid grey;

background-color: whitesmoke;

}

.box{

display: flex;

align-items: center;

justify-content: center;

position: absolute;

top: 50%;

left: 50%;

}

.text-black {

color: black;

}

.text-blue {

color: rgb(44, 119, 199);

}

.like-button.liked {

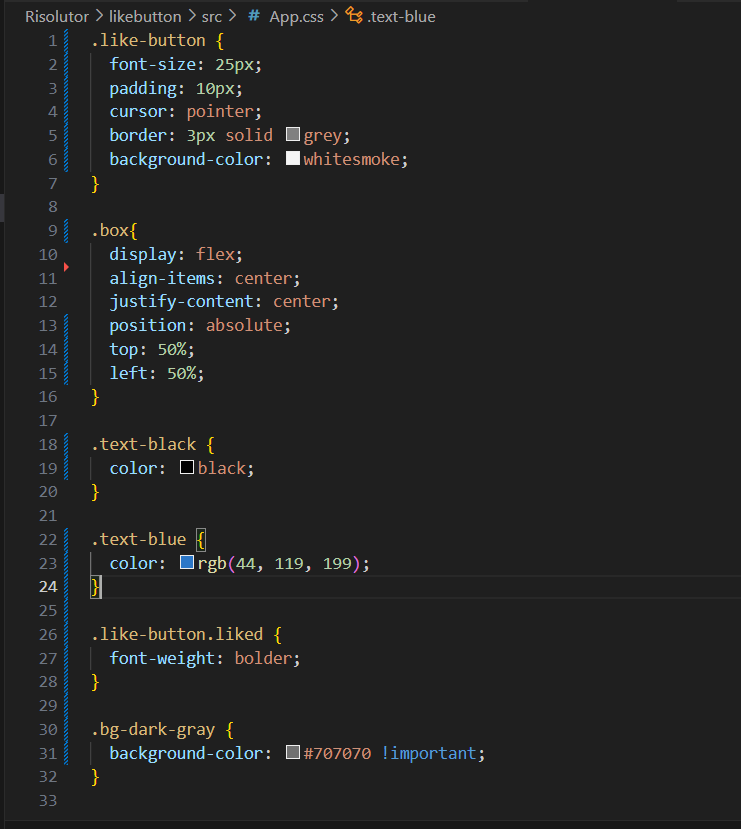
font-weight: bolder;

}

.bg-dark-gray {

background-color: #707070 !important;

}



**OUTPUT:**



