JAVASCRIPT CODING EXAMPLES

Q1. Reverse a given string using JavaScript?

```
var str = "Given String";

var output = str .split("") .reverse() .join("");

document.write(output);
```

Q2. Find the sum of all elements/numbers of a given array?

```
var arr = [1, 2, 5, 10, 20];

var sum = arr.reduce((a, i) => { return a + i; });

 document.write(sum);
```

Q3. Write a JavaScript program to sort a list of elements using Merge sort

```
function merge_sort(left_part,right_part)
{
    var i = 0;
    var j = 0;
    var results = [];

    while (i < left_part.length || j < right_part.length) {
        if (i === left_part.length) {
            // j is the only index left_part
            results.push(right_part[j]);
            j++;
        }
  else if (j === right_part.length || left_part[i] <= right_part[j]) {
            results.push(left_part[i]);
            i++;
        } else {
            results.push(right_part[j]);
            j++;
        }
    }
    return results;
}

console.log(merge_sort([1,3,4], [3,7,9]));
```

Q4. Write a JavaScript program to sort a list of elements using Insertion sort.

```
const insertion_Sort = (nums) => {
  for (let i = 1; i < nums.length; i++) {
    let j = i - 1
    let temp = nums[i]
    while (j >= 0 && nums[j] > temp) {
      nums[j + 1] = nums[j]
      j--
    }
    nums[j+1] = temp
  }
  return nums
}
console.log(insertion_Sort([3, 0, 2, 5, -1, 4, 1]));
console.log(insertion_Sort([2,6,5,12,-1,3,8,7,1,-4,0,23,1,-55,20,37,54,210,-
23,7,483,9339,29,-3,90,-2,81,54,7372,-92,93,93,18,-43,21]));
```

Q5. Write a JavaScript program to sort a list of elements using the Selection sort algorithm

The selection sort improves on the bubble sort by making only one exchange for every pass through the list.

```
// Selection sort with O(n^2) time complexity

function Selection_Sort(arr, compare_Function) {

  function compare(a, b) {
   return a - b;
   }
  var min = 0;
  var index = 0;
  var temp = 0;

 //{Function} compare_Function Compare function
  compare_Function = compare_Function || compare;

  for (var i = 0; i < arr.length; i += 1) {
    index = i;
    min = arr[i];

    for (var j = i + 1; j < arr.length; j += 1) {
      if (compare_Function(min, arr[j]) > 0) {
        min = arr[j];
        index = j;
      }
    }

    temp = arr[i];
    arr[i] = min;
    arr[index] = temp;
  }

  //return sorted arr
  return arr;
}

console.log(Selection_Sort([3, 0, 2, 5, -1, 4, 1], function(a, b) { return a - b; }));
console.log(Selection_Sort([3, 0, 2, 5, -1, 4, 1], function(a, b) { return b - a; }));
```

Q6. Write a JavaScript program to sort a list of elements using Bubble sort

```
function swap(arr, first_Index, second_Index){
    var temp = arr[first_Index];
    arr[first_Index] = arr[second_Index];
    arr[second_Index] = temp;
}

function bubble_Sort(arr){

    var len = arr.length,
        i, j, stop;

    for (i=0; i < len; i++){
        for (j=0, stop=len-i; j < stop; j++){
            if (arr[j] > arr[j+1]){
                swap(arr, j, j+1);
            }
        }
    }

    return arr;
}
console.log(bubble_Sort([3, 0, 2, 5, -1, 4, 1]));
```

Q7. Write a JavaScript program to check if a numeric array is sorted or not. if direction is +ve, the array is in ascending order, if it is -ve array is in descending order, otherwise it is not sorted.

```
const isSorted = arr => {
  if (arr.length <= 1) return 0;
  const direction = arr[1] - arr[0];
  for (let i = 2; i < arr.length; i++) {
    if ((arr[i] - arr[i - 1]) * direction < 0) return 0;
  }
  return Math.sign(direction);
};
console.log(isSorted([0, 1, 2, 2]));
console.log(isSorted([4, 3, 2]));
console.log(isSorted([4, 3, 5]));
console.log(isSorted([4]));
```

Q8. Write a JavaScript program to find the most frequent item of an array.

```
var arr1=[3, 'a', 'a', 'a', 2, 3, 'a', 3, 'a', 2, 4, 9, 3];
var mf = 1;
var m = 0;
var item;
for (var i=0; i<arr1.length; i++)
{
        for (var j=i; j<arr1.length; j++)
        {
                if (arr1[i] == arr1[j])
                 m++;
                if (mf<m)
                {
                  mf=m;
                  item = arr1[i];
                }
        }
        m=0;
}
console.log(item+" ( " +mf +" times ) ") ;
```

Q9. Write a JavaScript program to remove duplicate items from an array

```
function removeDuplicates(num) {
  var x,
      len=num.length,
      out=[],
      obj={};

  for (x=0; x<len; x++) {
    obj[num[x]]=0;
  }
  for (x in obj) {
    out.push(x);
  }
  return out;
}
var Mynum = [1, 2, 2, 4, 5, 4, 7, 8, 7, 3, 6];
result = removeDuplicates(Mynum);
console.log(Mynum);
console.log(result);
```

Q10. Write a JavaScript program to perform a binary search. Note : A binary search or half-interval search algorithm finds the position of a specified input value within an array sorted by key value.
Sample array :

```
var items = [1, 2, 3, 4, 5, 7, 8, 9];
```

Expected Output :

```
console.log(binary_Search(items, 1)); //0
console.log(binary_Search(items, 5)); //4
```

```
function binary_Search(items, value){
    var firstIndex  = 0,
        lastIndex   = items.length - 1,
        middleIndex = Math.floor((lastIndex + firstIndex)/2);

    while(items[middleIndex] != value && firstIndex < lastIndex)
    {
       if (value < items[middleIndex])
        {
            lastIndex = middleIndex - 1;
        }
      else if (value > items[middleIndex])
        {
            firstIndex = middleIndex + 1;
        }
        middleIndex = Math.floor((lastIndex + firstIndex)/2);
    }

 return (items[middleIndex] != value) ? -1 : middleIndex;
}
var items = [1, 2, 3, 4, 5, 7, 8, 9];
console.log(binary_Search(items, 1));
console.log(binary_Search(items, 5));
```

Q11. Write a program to print the Fibonacci series up to n terms

```
const number = parseInt(prompt('Enter the number of terms: '));
let n1 = 0, n2 = 1, nextTerm;

console.log('Fibonacci Series:');
```

```
for (let i = 1; i <= number; i++) {
    console.log(n1);
    nextTerm = n1 + n2;
    n1 = n2;
    n2 = nextTerm;
}
```

Q12. To check whether a given number is an Armstrong number or not.

```
let sum = 0;
const number = prompt('Enter a three-digit positive integer: ');

// create a temporary variable
let temp = number;
while (temp > 0) {
    // finding the one's digit
    let remainder = temp % 10;

    sum += remainder * remainder * remainder;

    // removing last digit from the number
    temp = parseInt(temp / 10); // convert float into integer
}
// check the condition
if (sum == number) {
    console.log(`${number} is an Armstrong number`);
}
else {
    console.log(`${number} is not an Armstrong number.`);
}
```

Q13. Write a JavaScript program to sort an array of JavaScript objects.

```
var library = [
  {
      title: 'Bill Gates',
      author: 'The Road Ahead',
      libraryID: 1254
  },
  {
      title: ‘Leo Tolstoy’,
      author: 'War and Peace',
      libraryID: 1259
```

```
   },
   {
       title: ‘Hamlet’,
       author: 'William Shakespeare',
       libraryID: 2354
   }];

var sort_by = function(field_name, reverse, initial){

   var key = initial ?
       function(x)
             {
               return initial(x[field_name]);
             } :
       function(x)
             {
               return x[field_name];
             };

   reverse = !reverse ? 1 : -1;

   return function (x, y) {
       return x = key(x), y = key(y), reverse * ((x > y) - (y > x));
     } ;
};


var newobj = library.sort(sort_by('libraryID', true, parseInt));

console.log(newobj);
```

Q14. Write a Recursive JavaScript function to compute the exponent of a number

```
var exponent = function(a, n)
{
   if (n === 0)
   {
    return 1;
    }
  else
  {
    return a * exponent(a, n-1);
  }
};
```

```
console.log(exponent(4, 2));
```

Q15. Write a JavaScript function to get difference between two dates in days.

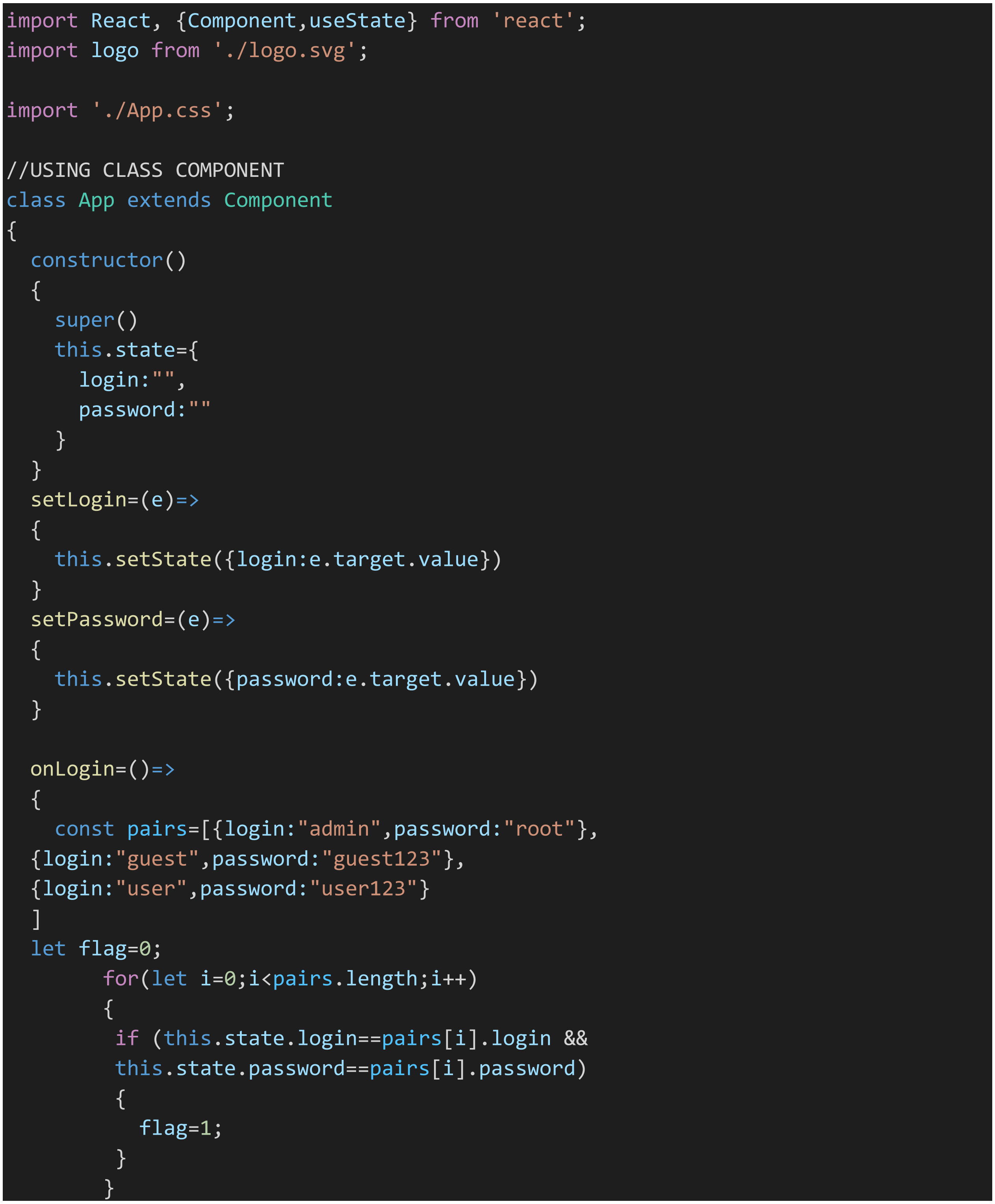
```
var date_diff_indays = function(date1, date2) {
        const dt1 = new Date(date1);
        const dt2 = new Date(date2);
        const utcDate1=Date.UTC(dt2.getFullYear(), dt2.getMonth(), dt2.getDate())
        const utcDate2=Date.UTC(dt1.getFullYear(), dt1.getMonth(), dt1.getDate())

        // The Date.UTC() method accepts parameters similar to the Date constructor,
        but treats them as UTC. It returns the number of milliseconds since January 1,
        1970, 00:00:00 UTC

        return Math.floor( (x1-x2) / (1000 * 60 * 60 * 24) );
}

console.log(date_diff_indays('04/02/2014', '11/04/2014'));
console.log(date_diff_indays('12/02/2014', '11/04/2014'));
```

Login Authentication Application in JavaScript

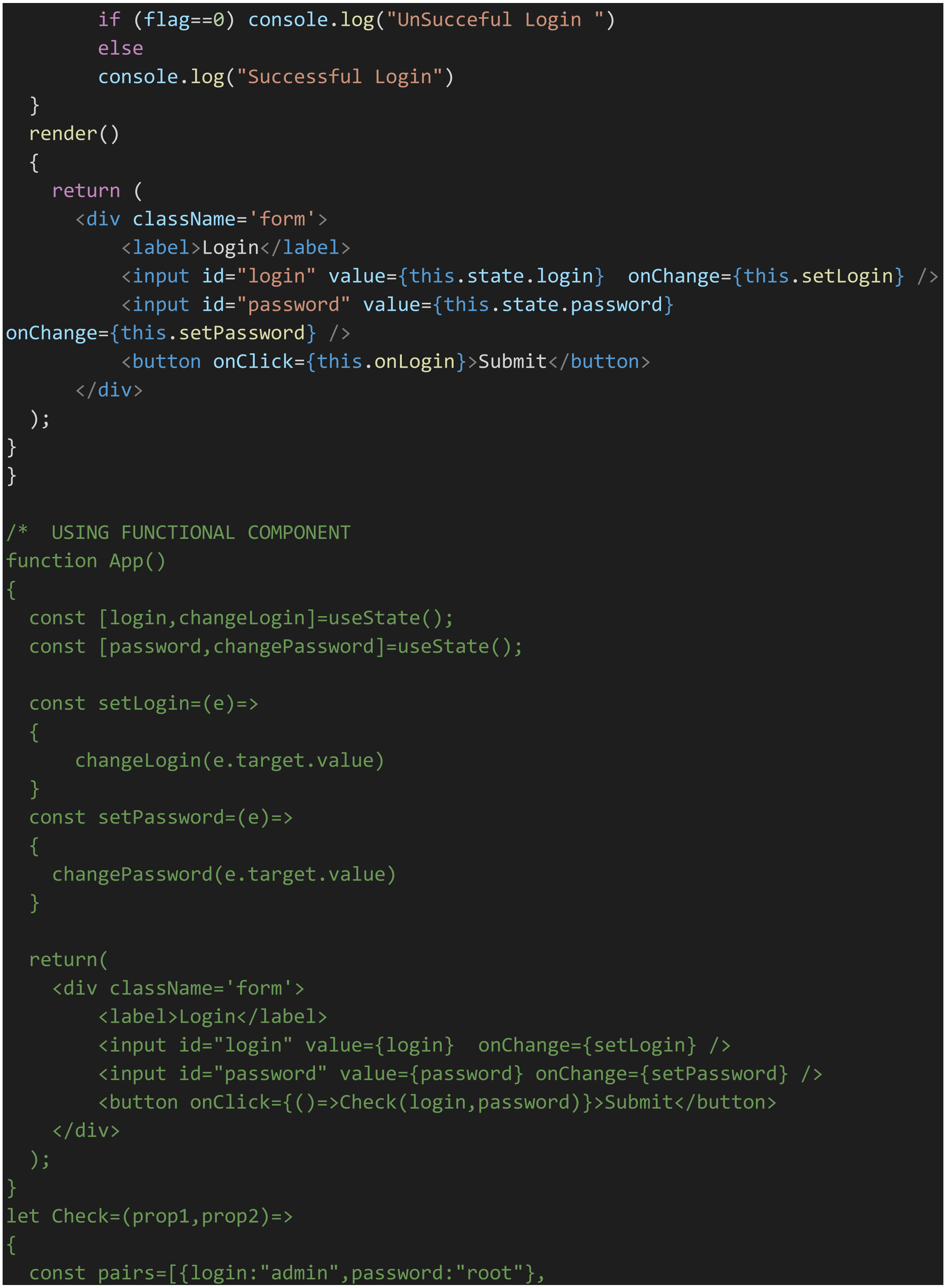
```
<script>
    const pairs=[{login:"a",password:"p"},{login:"b",password:"q"}]
    function submit()
    {
     let flag=0;
     for (let i=0;i<pairs.length;i++)
     {
     if (this.document.getElementById("login").value==pairs[i].login &&
         this.document.getElementById("password").value==pairs[i].password)
     {
       console.log("Correct");break;
     }
     else
     {
       console.log("Incocrrect");break;
     }
    }
    }
   </script>
```

Login Authentication Application in ReactJS using Function and Class Components

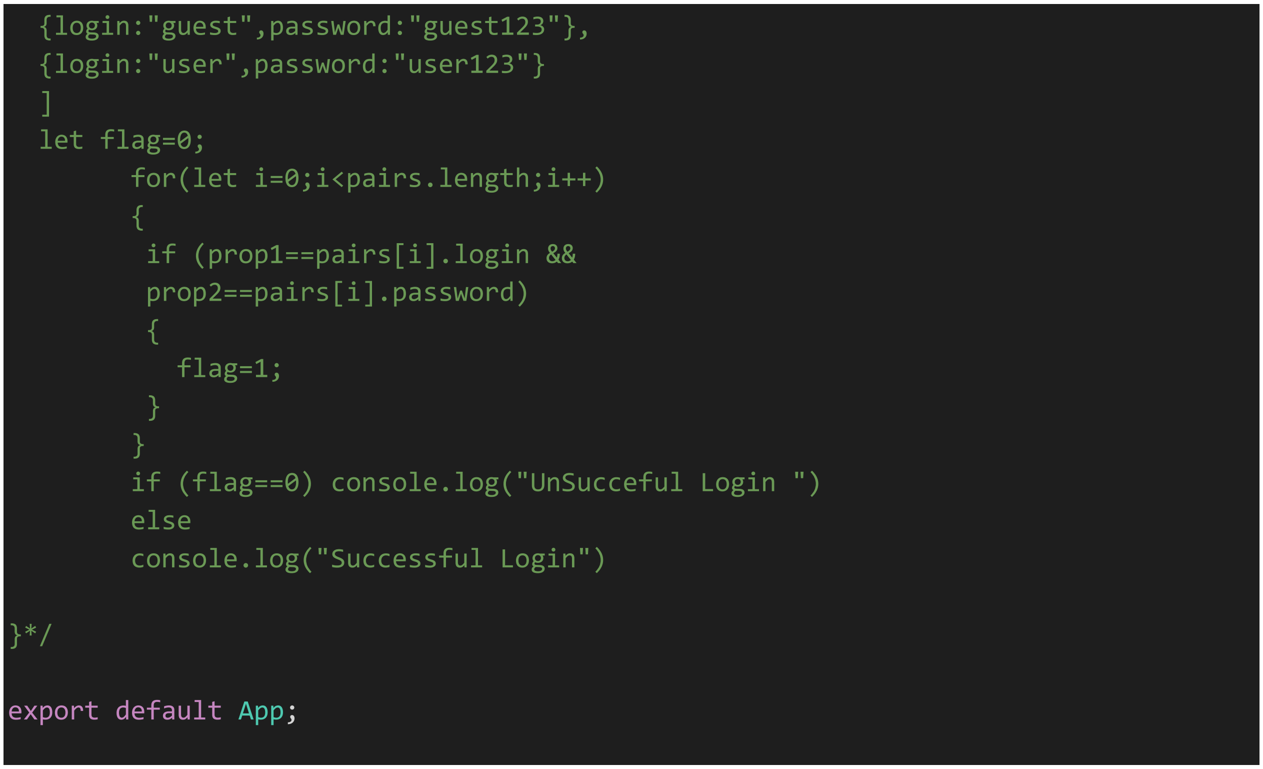
App.js



```
import React, {Component,useState} from 'react';
import logo from './logo.svg';

import './App.css';

//USING CLASS COMPONENT
class App extends Component
{
  constructor()
  {
    super()
    this.state={
      login:"",
      password:""
    }
  }
  setLogin=(e)=>
  {
    this.setState({login:e.target.value})
  }
  setPassword=(e)=>
  {
    this.setState({password:e.target.value})
  }

  onLogin=()=>
  {
    const pairs=[{login:"admin",password:"root"},
  {login:"guest",password:"guest123"},
  {login:"user",password:"user123"}
  ]
  let flag=0;
        for(let i=0;i<pairs.length;i++)
        {
         if (this.state.login==pairs[i].login &&
         this.state.password==pairs[i].password)
         {
           flag=1;
         }
        }
```



```
        if (flag==0) console.log("UnSucceful Login ")
        else
        console.log("Successful Login")
  }
  render()
  {
    return (
      <div className='form'>
          <label>Login</label>
          <input id="login" value={this.state.login}  onChange={this.setLogin} />
          <input id="password" value={this.state.password}
onChange={this.setPassword} />
          <button onClick={this.onLogin}>Submit</button>
      </div>
  );
}
}

/*  USING FUNCTIONAL COMPONENT
function App()
{
  const [login,changeLogin]=useState();
  const [password,changePassword]=useState();

  const setLogin=(e)=>
  {
      changeLogin(e.target.value)
  }
  const setPassword=(e)=>
  {
    changePassword(e.target.value)
  }

  return(
    <div className='form'>
        <label>Login</label>
        <input id="login" value={login}  onChange={setLogin} />
        <input id="password" value={password} onChange={setPassword} />
        <button onClick={()=>Check(login,password)}>Submit</button>
    </div>
  );
}
let Check=(prop1,prop2)=>
{
  const pairs=[{login:"admin",password:"root"},
```



```
  {login:"guest",password:"guest123"},
  {login:"user",password:"user123"}
  ]
  let flag=0;
        for(let i=0;i<pairs.length;i++)
        {
         if (prop1==pairs[i].login &&
         prop2==pairs[i].password)
         {
           flag=1;
         }
        }
        if (flag==0) console.log("UnSucceful Login ")
        else
        console.log("Successful Login")

}*/

export default App;
```