Assignment No # 02

Name: Saba Bibi

Reg No: FA20-BCS-062

Submitted to: Mr. Muhammad Kamran

Dated: 26 March 2023

COMSATS UNIVERSITY ISLAMABAD, ATTOCK CAMPUS

String built-in functions:

1. toUpperCase ():

This function converts all the characters in a string to uppercase letters.

Example:

```
let str = "hello world";
let upperStr = str.toUpperCase();
console.log(upperStr);
```

Output:

```
node /tmp/scYKSrIHPl.js
HELLO WORLD
```

2. toLowerCase ():

This function converts all the characters in a string to lowercase letters.

Example:

```
let str = "FUNCTION";
let lowerStr = str.toLowerCase();
console.log(lowerStr);
```

Output:

```
node /tmp/scYKSrIHPl.js
function
```

3. charAt ():

This function returns the character at a specified index in a string.

Example:

```
let str = "my function";
let char = str.charAt(1);
console.log(char);
```

Output:

```
node /tmp/scYKSrIHPl.js
y
```

4. substring ():

This function returns a part of a string between two specified indices.

Example:

```
let str = "my function";
let subStr = str.substring(1, 6);
console.log(subStr);
```

Output:

```
node /tmp/OfFtMb9GOU.js
y fun
```

5. split ():

This function splits a string into an array of substrings based on a specified separator.

Example:

```
let str = "my function";
let arr = str.split(" ");
console.log(arr);
```

Output:

```
node /tmp/OfFtMb9GOU.js
[ 'my', 'function' ]
```

6. length:

This property returns the length of a string.

Example:

```
let str = "my function";
let len = str.length;
console.log(len);
```

Output:

```
node /tmp/0fFtMb9GOU.js
11
```

7. replace ():

The replace () method replaces a specified value with another value in a string.

Example:

```
const text = 'Hello, world!';
const newText = text.replace('world', 'JavaScript');
console.log(newText);
```

Output:

```
node /tmp/lp8AhJhrzO.js
Hello, JavaScript!
```

Array built-in Functions:

1. push():

The push() method adds one or more elements to the end of an array and returns the new length of the array.

Example:

```
const arr = ['Asmara', 'Aleeza'];
arr.push('Rabia', 'Iqra');
console.log(arr);
```

Output:

```
node /tmp/lp8AhJhrzO.js
[ 'Asmara', 'Aleeza', 'Rabia', 'Iqra' ]
```

2. pop():

The pop() method removes the last element from an array and returns that element.

Example:

```
const arr = ['Asmara', 'Aleeza', 'Rabia', 'Iqra'];
const lastArr = arr.pop();
console.log(lastArr);
console.log(arr);
```

Output:

```
node /tmp/lp8AhJhrzO.js
Iqra
[ 'Asmara', 'Aleeza', 'Rabia' ]
```

3. shift():

The shift() method removes the first element from an array and returns that element.

Example:

```
const names = ['Asmara', 'Aleeza', 'Rabia', 'Iqra'];
const firstName = names.shift();
console.log(firstName);
console.log(names);
```

Output:

```
node /tmp/kAOkXt50H0.js
Asmara
[ 'Aleeza', 'Rabia', 'Iqra' ]
```

4. unshift():

The unshift() method adds one or more elements to the beginning of an array and returns the new length of the array.

Example:

```
const names = ['Iqra', 'Aleeza'];
const newLength = names.unshift('Rabia', 'Asmara');
console.log(newLength);
console.log(names);
```

Output:

```
node /tmp/kAOkXt50H0.js
4
[ 'Rabia', 'Asmara', 'Iqra', 'Aleeza' ]
```

5. splice():

The splice() method adds or removes elements from an array at a specified index.

Example:

```
const names = ['Asmara', 'Aleeza', 'Rabia', 'Iqra'];
names.splice(3, 1, 'Momina', 'Sanawish');
console.log(names);
```

Output:

```
node /tmp/kAOkXt50H0.js
[ 'Asmara', 'Aleeza', 'Rabia', 'Momina', 'Sanawish' ]
```

6. slice():

The slice() method returns a shallow copy of a portion of an array into a new array object selected from start to end (end not included).

Example:

```
const names = ['Asmara', 'Aleeza', 'Rabia', 'Iqra'];
const selectedNames = names.slice(1, 3);
console.log(selectedNames);
```

Output:

```
node /tmp/9G3vdzpOxZ.js
[ 'Aleeza', 'Rabia' ]
```

7. concat():

The concat() method merges two or more arrays into a new array.

Example:

```
const arr1 = ['Asmara', 'Rabia'];
const arr2 = ['Aleeza', 'Iqra'];
const allArrays = arr1.concat(arr2);
console.log(allArrays);
```

Output:

```
node /tmp/9G3vdzpOxZ.js
[ 'Asmara', 'Rabia', 'Aleeza', 'Iqra' ]
```

8. indexOf():

The indexOf() method returns the first index at which a given element can be found in an array, or -1 if it is not present.

Example:

```
const names = ['Asmara', 'Aleeza', 'Rabia', 'Iqra'];
const index = names.indexOf('Aleeza');
console.log(index);
```

Output:

```
node /tmp/9G3vdzpOxZ.js
1
```

9. forEach():

The forEach() method executes a provided function once for each array element.

Example:

```
const names = ['Asmara', 'Aleeza', 'Rabia', 'Iqra'];
names.forEach(function(name) {
  console.log(name);
});
```

Output:

```
node /tmp/9G3vdzpOxZ.js
Asmara
Aleeza
Rabia
Iqra
```

Chess Board:

Code:

```
import React from 'react';
import { View, StyleSheet } from 'react-native';

const Board = () => {
  const board = [];

  for (let i = 0; i < 8; i++) {
    const row = [];

    for (let j = 0; j < 8; j++) {
      const color = (i + j) % 2 === 0 ? '#e6e6e6' : 'black';
      row.push(
        <View key={`${i}-${j}`} style={[styles.cell, { backgroundColor: color }]} />
      );
    }

    board.push(
      <View key={i} style={styles.row}>
        {row}
      </View>
    );
  }

  return <View style={styles.board}>{board}</View>;
};

const ChessScreen = () => {
  return (
    <View style={styles.container}>
      <Board />
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
```

```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    alignItems: 'center',
    justifyContent: 'center',
  },
  board: {
    flex: 1,
    flexDirection: 'column',
    justifyContent: 'center',
    alignItems: 'center',
  },
  row: {
    flexDirection: 'row',
    justifyContent: 'center',
    alignItems: 'center',
  },
  cell: {
    width: 40,
    height: 40,
  },
});

export default ChessScreen;
```

Output: