

# REACT NATIVE PRACTICE PROBLEMS

## Contents

INTRODUCTION (CLO-1) .....	1
JAVASCRIPT (CLO-2) .....	4
REACT NATIVE CORE COMPONENTS (CLO-3) .....	10
REACT NATIVE STYLES (CLO-3) .....	29
.....	30

## INTRODUCTION (CLO-1)

### Question: 1

#### Define positive impacts of mobile apps on an organization

Mobile apps can have numerous positive impacts on an organization, including:

1. **Enhanced Customer Engagement:** Mobile apps provide a direct and personalized channel for organizations to engage with their customers. Through features like push notifications, businesses can send targeted messages, promotions, and updates to keep users engaged.
2. **Increased Accessibility:** Mobile apps allow users to access information and services anytime, anywhere, leading to improved accessibility. This is particularly beneficial for businesses aiming to reach a global audience and cater to users on the go.
3. **Improved Customer Satisfaction:** Well-designed mobile apps can enhance the overall customer experience by offering convenient and user-friendly interfaces. This, in turn, contributes to increased customer satisfaction and loyalty.
4. **Streamlined Operations:** Organizations can use mobile apps to streamline internal processes, improving communication and

collaboration among employees. Mobile apps can facilitate task management, document sharing, and other essential business functions.

5. **Data Collection and Analytics:** Mobile apps enable organizations to collect valuable user data, providing insights into user behavior and preferences. This data can be leveraged for informed decision-making, targeted marketing, and product/service improvement

## **Question: 2**

### **What are the negative impacts of mobile apps.**

While mobile apps bring many benefits, they can also have some negative impacts, such as:

1. **Security Concerns:** Mobile apps may be susceptible to security breaches, putting sensitive user data at risk. Organizations need to implement robust security measures to protect against potential threats.
2. **Compatibility Issues:** Developing apps for multiple platforms can lead to compatibility issues. Ensuring consistent performance across various devices and operating systems can be challenging.
3. **High Development and Maintenance Costs:** Creating and maintaining mobile apps can be expensive, especially when developing for multiple platforms. Regular updates and addressing compatibility issues can contribute to ongoing costs.
4. **Overdependence on Mobile Technology:** Organizations may become overly reliant on mobile apps, neglecting other channels. This could result in missed opportunities and limit the organization's overall reach.

## **Question: 3-4**

### **What are native app and Define hybrid apps.**

**Native Apps:** Native apps are applications specifically built for a particular mobile operating system (e.g., iOS or Android). They are designed to leverage the full capabilities of the device, offering optimal performance and a seamless user experience.

**Hybrid Apps:** Hybrid apps combine elements of both native and web applications. They are built using web technologies (HTML, CSS, and JavaScript) and are then encapsulated within a native container. Hybrid apps aim to provide a balance between cross-platform compatibility

and performance.

### **Question: 5**

#### **Define swift programming language.\**

Swift is a programming language developed by Apple for building iOS, macOS, watchOS, and tvOS applications. It is designed to be fast, modern, and safe, offering developers a more efficient and expressive way to write code compared to Objective-C.

### **Question: 6**

#### **Define flutter framework**

Flutter is an open-source UI software development toolkit created by Google. It is used for building natively compiled applications for mobile, web, and desktop from a single codebase. Flutter uses the Dart programming language and is known for its fast development and expressive UI.

### **Question: 7**

#### **Define Xamarin framework**

Xamarin is a Microsoft-owned framework for building cross-platform mobile applications. It allows developers to use C# for building apps that can run on iOS, Android, and Windows. Xamarin provides a single codebase for different platforms, streamlining development.

### **Question: 8**

#### **Define react native framework**

React Native is an open-source framework developed by Facebook for building cross-platform mobile applications. It allows developers to use React (a JavaScript library) to create native-like experiences on both iOS and Android platforms, sharing a significant portion of the codebase.

### **Question: 9**

#### **Discuss four benefits of react-native framework.**

Four Benefits of React Native Framework:

1. **Cross-Platform Development:** React Native enables developers to write code once and deploy it on both iOS and Android platforms, reducing development time and effort

compared to building separate native apps.

2. **Hot Reloading:** React Native supports hot reloading, allowing developers to instantly view the effects of code changes during development without restarting the entire application. This speeds up the development process and enhances productivity.
3. **Native-Like Performance:** React Native apps deliver performance comparable to native apps because they use native components. This is achieved through the use of a bridge that connects React Native code to native modules.

**Large Developer Community:** React Native has a large and active developer community, providing a wealth of resources, libraries, and third-party plugins. This community support can be advantageous

## JAVASCRIPT (CLO-2)

### Question: 1

**Create a javascript class Person with attributes: id, name, age. Derive two classes from person, named Student and Teacher.**

**The extra attributes of Student are cgpa, currently enrolled semester (e.g., FA22 or SP22, etc), admission date.**

**The extra attributes of Teacher are salary, designation (Lecturer, Assistant Professor, Professor, etc), department, and joining date.**

**Populate at least 3 records in each class use class objects.**

**A user should be able to search a student or teacher with the provided ID. To manage that you should store objects of Teacher and Student in an array.**

### Solution:

```
class Person {  
  
    constructor(id, name, age) {  
  
        this.id = id;  
  
        this.name = name;  

```

```

        this.age = age;
    }
}

class Student extends Person {

    constructor(id, name, age, cgpa, enrolledSemester, admissionDate) {

        super(id, name, age);

        this.cgpa = cgpa;

        this.enrolledSemester = enrolledSemester;

        this.admissionDate = admissionDate;

    }

}

class Teacher extends Person {

    constructor(id, name, age, salary, designation, department, joiningDate) {

        super(id, name, age);

        this.salary = salary;

        this.designation = designation;

        this.department = department;

        this.joiningDate = joiningDate;

    }

}

// Populating records

let students = [

    new Student(1, "John Doe", 20, 3.5, "FA22", "2022-01-01"),

    new Student(2, "Jane Doe", 22, 3.8, "SP22", "2021-09-15"),

    new Student(3, "Bob Smith", 21, 3.2, "FA21", "2021-02-28")

];

let teachers = [

    new Teacher(101, "Prof. Johnson", 35, 60000, "Professor", "Computer Science", "2020-07-10"),

```

```

    new Teacher(102, "Dr. Williams", 40, 75000, "Assistant Professor", "Mathematics", "2019-05-20"),

    new Teacher(103, "Lecturer Davis", 28, 50000, "Lecturer", "Physics", "2022-01-15")

];

// Searching function

function searchPersonById(array, id) {

    return array.find(person => person.id === id);

}

// Example usage

console.log(searchPersonById(students, 2)); // Search for Student with ID 2
console.log(searchPersonById(teachers, 101)); // Search for Teacher with ID

```

## Question: 2

**Write arrow functions for the following equations:**

$$A = x^2 + 2xy + P \cdot z$$

$$A = n^2 + qn + 1$$

$$Z = x^2 + 4y^2 - 8y + 2x$$

Solution:

// Equation 1

```
const calculateA1 = (x, y, P) => x ** 2 + 2 * x * y + P * z;
```

// Equation 2

```
const calculateA2 = (n, q) => n ** 2 + q * n + 1;
```

// Equation 3

```
const calculateZ = (x, y) => x ** 2 + 4 * y ** 2 - 8 * y + 2 * x;
```

## Question: 3

**Suppose the equation is:  $Z = x^2 + 4y^2 - 8N + 2x$**

**Where N is represented by a separate equation:  $N = p^2z + rq^2 + s$**

**Solve 'Z' with arrow function. Note, here you are calling an arrow function within an arrow function.**

```
// Equation for N

const calculateN = (p, z, q, s) => p ** (2 * z) + q * (s ** 2) + s;

// Arrow function for Z using N

const calculateZWithN = (x, y, p, q, s) => {

  const N = (z) => p ** (2 * z) + q * (s ** 2) + s;

  return x ** 2 + 4 * y ** 2 - 8 * N(x) + 2 * x;

};
```

**Question: 4** Suppose you have the following array of objects,

```
var myarray: [ { 'name': 'ali', 'age': '45' }, { 'name': 'noman', 'age': '34' } ]
```

**Display the values of array using map function.**

```
var myarray = [

  { 'name': 'ali', 'age': '45' },

  { 'name': 'noman', 'age': '34' }

];

// Using map function to display values

var displayValues = myarray.map(person => {

  console.log(`Name: ${person.name}, Age: ${person.age}`);

})

// The above code will log:

// Name: ali, Age: 45

// Name: noman, Age: 34
```

**Question: 5** Suppose we have the following arrays in JavaScript `var myArray1 = [3, 4, 5]`

`var myArray2 = [6, 7, 8]`

**Write code to append the myArray2 into myArray1.**

```
var myArray1 = [3, 4, 5];
var myArray2 = [6, 7, 8];
// Using spread operator to append myArray2 into myArray1
var combinedArray = [...myArray1, ...myArray2];
console.log(combinedArray);
// The output will be: [3, 4, 5, 6, 7, 8]
```

**Question: 6** Suppose we have an object `var myObject1 = { name: 'Devin', hairColor: 'brown' }` Write code to change value of hairColor using spread syntax (...) three dots.

```
var myObject1 = { name: 'Devin', hairColor: 'brown' };
// Using spread syntax to change the value of hairColor
var updatedObject = { ...myObject1, hairColor: 'blonde' };
console.log(updatedObject);
// The output will be: { name: 'Devin', hairColor: 'blonde' }
```

**Question: 7** Write an example of defining an arrow function within another arrow function.

```
// Outer arrow function
const outerFunction = (x) => {
  // Inner arrow function
  const innerFunction = (y) => {
    return x + y;
  };
  return innerFunction;
};
// Example usage
const innerFunctionInstance = outerFunction(5);
```

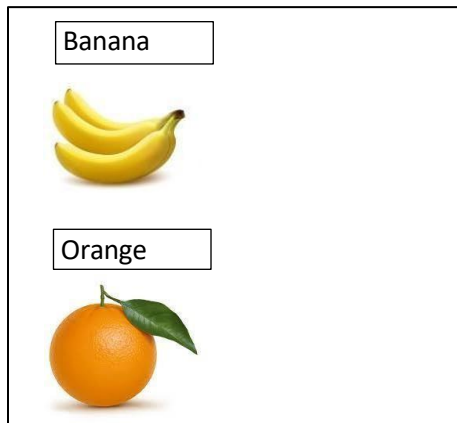


```
console.log(innerFunctionInstance(3)); // Ou
```

# REACT NATIVE CORE COMPONENTS (CLO-3)

## Question: 1

Make an app in react native that shows the following on screen:



For this task, create a folder “images” and place the banana.jpg and orange.jpg in the images folder.

```
// ImageList.js
import React from 'react';
import { View, Image, Text, StyleSheet } from 'react-native';

const ImageList = () => {
  return (
    <View style={styles.container}>
      <Image source={require('./images/banana.jpg')} style={styles.image} />
      <Text style={styles.text}>Banana</Text>

      <Image source={require('./images/orange.jpg')} style={styles.image} />
      <Text style={styles.text}>Orange</Text>
    </View>
  );
};
```

```
// App.js
import React from 'react';
import ImageList from './ImageList';

const App = () => {
  return <ImageList />;
};

export default App;
```

## Question: 2

Write a function component to show grade of a student for the given marks. The marks are provided to a javascript arrow function as argument, e.g., `calculateGrade(marks)` which is called in the `<Text></Text>` of function component. Here is the grade distribution:

`< 50 --- F`

`>= 50 and < 60 --- E`

`>= 60 and < 70 --- D`

`>= 70 and < 80 --- C`

`>= 80 and < 90 --- B`

`>= 90 --- A`

`// GradeComponent.js`

`import React from 'react';`

`import { Text } from 'react-native';`

`const calculateGrade = (marks) => {`

`if (marks < 50) return 'F';`

`else if (marks < 60) return 'E';`

`else if (marks < 70) return 'D';`

`else if (marks < 80) return 'C';`

`else if (marks < 90) return 'B';`

`else return 'A';`

`};`

`const GradeComponent = ({ marks }) => {`

`const grade = calculateGrade(marks);`

`return <Text>{`Grade: ${grade}`}</Text>;`

`};`

`export default GradeComponent;`

### QUESTION 3

Use props and pass names of students from a Name() function component to Attendance() function component. The following should be the output by Attendance() function component.

<b>Ali Khan</b>	<b>Present</b>
<b>Noman</b>	<b>Present</b>
<b>Faisal</b>	<b>Absent</b>
<b>Javed</b>	<b>Absent</b>

```
// NameComponent.js
```

```
import React from 'react';
```

```
import AttendanceComponent from './AttendanceComponent';
```

```
const NameComponent = () => {
```

```
  const students = ['Ali Khan', 'Noman', 'Faisal', 'Javed'];
```

```
  return <AttendanceComponent names={students} />;
```

```
};
```

```
export default NameComponent;
```

```
// AttendanceComponent.js
```

```
import React from 'react';
```

```
import { Text } from 'react-native';
```

```
const AttendanceComponent = ({ names }) => {
```

```
  const attendanceStatus = names.map((name) => (
```

```
    <Text key={name}>{'${name} ${Math.random() > 0.5 ? 'Present' : 'Absent'}'}</Text>
```

```
  ));
```

```
  return <>{attendanceStatus}</>;
```

```
};
```

```
export default AttendanceComponent;
```

#### QUESTION 4

**Write code to add a button in React Native. The text showing in the button should be Click Here. When the button is clicked, an alert dialog should be shown with message “hello world”.**

```
// ButtonWithAlert.js
import React from 'react';
import { View, Button, Alert } from 'react-native';

const ButtonWithAlert = () => {
  const showAlert = () => {
    Alert.alert('Hello World');
  };

  return (
    <View>
      <Button title="Click Here" onPress={showAlert} />
    </View>
  );
};

export default ButtonWithAlert;
```

---

#### QUESTION 5

**Write the code of App function.**

**When a user enters any text in the TextInput, it is also automatically written in another TextInput.**

```
// TextInputHandling.js
import React, { useState } from 'react';
import { View, TextInput } from 'react-native';

const TextInputHandling = () => {
  const [inputText, setInputText] = useState("");

  return (
    <View>
      <TextInput
        placeholder="Type here..."
        onChangeText={(text) => setInputText(text)}
        value={inputText}
      />
      <TextInput placeholder="Automatically updated" value={inputText} />
    </View>
  );
};

export default TextInputHandling
```

## QUESTION 6

Create a simple registration page in react native asking for users username, email, name, and cell number. When the user click on register button, the information should be shown using `<Text></Text>` elements. However, if any input is missing, message should be shown about the missing element.

```
// FunctionalRegistrationPage.js
import React, { useState } from 'react';
import { View, Text, TextInput, Button, Alert } from 'react-native';

const FunctionalRegistrationPage = () => {
  const [username, setUsername] = useState("");
  const [email, setEmail] = useState("");
  const [name, setName] = useState("");
  const [cellNumber, setCellNumber] = useState("");

  const handleRegister = () => {
    if (!username || !email || !name || !cellNumber) {
      Alert.alert('Missing Information', 'Please fill in all fields');
    } else {
      // Displaying information using <Text> elements
      console.log('Username:', username);
      console.log('Email:', email);
      console.log('Name:', name);
      console.log('Cell Number:', cellNumber);
    }
  };

  return (
    <View>
      <TextInput
        placeholder="Username"
        value={username}
        onChangeText={(text) => setUsername(text)}
      />
      <TextInput
        placeholder="Email"
        value={email}
        onChangeText={(text) => setEmail(text)}
      />
      <TextInput
        placeholder="Name"
        value={name}
        onChangeText={(text) => setName(text)}
      />
    </View>
  );
};
```

```

    <TextInput
      placeholder="Cell Number"
      value={cellNumber}
      onChangeText={(text) => setCellNumber(text)}
    />
    <Button title="Register" onPress={handleRegister} />
  </View>
);
};

export default FunctionalRegistrationPage;

```

## QUESTION 7

**Use the class component to do the following:**

**Create a simple registration page in react native asking for users username, email, name, and cell number. When the user click on register button, the information should be shown using <Text></Text> elements. However, if any input is missing, message should be shown about the missing element.**

```

// ClassRegistrationPage.js
import React, { Component } from 'react';
import { View, Text, TextInput, Button, Alert } from 'react-native';

class ClassRegistrationPage extends Component {
  constructor(props) {
    super(props);
    this.state = {
      username: "",
      email: "",
      name: "",
      cellNumber: "",
    };
  }

  handleRegister = () => {
    const { username, email, name, cellNumber } = this.state;

    if (!username || !email || !name || !cellNumber) {
      Alert.alert('Missing Information', 'Please fill in all fields');
    } else {
      // Displaying information using <Text> elements
      console.log('Username:', username);
      console.log('Email:', email);
      console.log('Name:', name);
    }
  }
}

```

```

        console.log('Cell Number:', cellNumber);
    }
};

render() {
    return (
        <View>
            <TextInput
                placeholder="Username"
                value={this.state.username}
                onChangeText={(text) => this.setState({ username: text })}
            />
            <TextInput
                placeholder="Email"
                value={this.state.email}
                onChangeText={(text) => this.setState({ email: text })}
            />
            <TextInput
                placeholder="Name"
                value={this.state.name}
                onChangeText={(text) => this.setState({ name: text })}
            />
            <TextInput
                placeholder="Cell Number"
                value={this.state.cellNumber}
                onChangeText={(text) => this.setState({ cellNumber: text })}
            />
            <Button title="Register" onPress={this.handleRegister} />
        </View>
    );
}
}

export default ClassRegistrationPage;

```

## QUESTION 8

**Suppose that you have a string defined in strings.js. The name of the string is `country_name` and value is “Pakistan”. Write a program that shows the value of string in your app function. (You need to import the string from strings.js)**

```

// strings.js
export const country_name = 'Pakistan';

// App.js
import React from 'react';
import { Text } from 'react-native';
import { country_name } from './strings';

const App = () => {
    return <Text>{country_name}</Text>;
};

```



```
export default App;
```

## QUESTION 9

**Suppose you have two TextInputs, each containing a number, and a button to add the values of the two TextInputs. When the button is clicked, the values of the TextInputs are added and result should be shown in console.**

```
// AddValuesAndShowResult.js
import React, { useState } from 'react';
import { View, TextInput, Button } from 'react-native';

const AddValuesAndShowResult = () => {
  const [value1, setValue1] = useState("");
  const [value2, setValue2] = useState("");

  const handleAdd = () => {
    const result = parseFloat(value1) + parseFloat(value2);
    console.log('Result:', result);
  };

  return (
    <View>
      <TextInput
        placeholder="Enter value 1"
        keyboardType="numeric"
        value={value1}
        onChangeText={(text) => setValue1(text)}
      />
      <TextInput
        placeholder="Enter value 2"
        keyboardType="numeric"
        value={value2}
        onChangeText={(text) => setValue2(text)}
      />
      <Button title="Add" onPress={handleAdd} />
    </View>
  );
};

export default AddValuesAndShowResult;
```

## QUESTION 10

**Suppose we have data in this format:**

```
{ [
  { "name": "ali", "age": "23" },
  { "name": "shazia", "age": "22" }
]}
```

**Write code to show the above array of objects in a <FlatList>**

```

// YourComponent.js

import React from 'react';
import { FlatList, Text, View } from 'react-native';

const data = [
  { "name": "ali", "age": "23" },
  { "name": "shazia", "age": "22" }
];

const YourComponent = () => {
  const renderItem = ({ item }) => (
    <View>
      <Text>Name: {item.name}</Text>
      <Text>Age: {item.age}</Text>
    </View>
  );
  return (
    <FlatList
      data={data}
      renderItem={renderItem}
      keyExtractor={(item) => item.name} // Assuming each name is unique
    />
  );
};

export default YourComponent;

```

## QUESTION 11

Suppose we have a predefined function that has following prototype:

**function GetComputerChoice().**When this function is called in a button click event, it returns either “fire”, “wood”, or “water”. In the same button click event, a random number is generated from 1 to 3, such that if 1 is generated, this means, the user gets “fire”, if 2 is generated, the user gets “wood”, and if 3 is generated, the user gets “water”. You need to check against the button click event that which of the computer or user has WON. Show name of winner in alert. **NOTE: Wood > Water; Water > Fire; Fire > Wood**

```
function GetComputerChoice() {  
    const choices = ["fire", "wood", "water"];  
    const randomIndex = Math.floor(Math.random() * choices.length);  
    return choices[randomIndex];  
}  
  
function handleButtonClick() {  
    const computerChoice = GetComputerChoice();  
    const userChoiceIndex = Math.floor(Math.random() * 3) + 1;  
    let userChoice;  
    if (userChoiceIndex === 1) {  
        userChoice = "fire";  
    } else if (userChoiceIndex === 2) {  
        userChoice = "wood";  
    } else {  
        userChoice = "water";  
    }  
  
    // Check the winner based on the rules  
    let winner;  
    if (  
        (userChoice === "wood" && computerChoice === "water") ||
```

```

    (userChoice === "water" && computerChoice === "fire") ||
    (userChoice === "fire" && computerChoice === "wood")
  ) {
    winner = "User";
  } else if (
    (computerChoice === "wood" && userChoice === "water") ||
    (computerChoice === "water" && userChoice === "fire") ||
    (computerChoice === "fire" && userChoice === "wood")
  ) {
    winner = "Computer";
  } else {
    winner = "It's a tie!";
  }

  // Show the winner in an alert
  alert(`Winner:   ${winner}\nComputer's   Choice:   ${computerChoice}\nUser's   Choice:
${userChoice}`);
}

// Example usage in a button click event
// You can call handleButtonClick() when the button is clicked

```

## QUESTION 12

**Suppose we have a layout like this:**

The image shows a simple UI component within a rectangular border. At the top is a text input field containing the word "Pakistan". Below it is another empty text input field. At the bottom is a button with the text "CLICK" inside it.

**When the button is clicked, the value from above TextInput should be written to the below TextInput. You need to write the code for the event handler defined for button.**

```
import React, { useState } from 'react';
import { View, TextInput, Button } from 'react-native';

const YourComponent = () => {
  const [inputValue, setInputValue] = useState("");
  const [outputValue, setOutputValue] = useState("");

  const handleButtonClick = () => {
    // Set the value of the second TextInput using the state
    setOutputValue(inputValue);
  };

  return (
    <View>
      { /* First TextInput */}
      <TextInput
        placeholder="Enter text"
        value={inputValue}
        onChangeText={(text) => setInputValue(text)}
      />

      { /* Button to trigger the event handler */}
      <Button title="Copy Text" onPress={handleButtonClick} />
    </View>
  );
};
```

```

    { /* Second TextInput */}

    <TextInput
      placeholder="Copied text will appear here"
      value={outputValue}
      // You can make the second TextInput read-only if needed
      editable={false}
    />

  </View>

);

};

export default YourComponent;

```

### QUESTION 13

**Create a program with following layout:**



**When the button is clicked, the text “hello world” should be shown in the textbox, and the button should be disabled. You can use hooks and state variables.**

```

import React, { useState } from 'react';
import { View, TextInput, Button, Alert } from 'react-native';

const YourComponent = () => {
  const [textValue, setTextValue] = useState("");
  const [buttonDisabled, setButtonDisabled] = useState(false);

```

```

const handleButtonClick = () => {
  // Show "hello world" in the textbox
  setValue('hello world');

  // Disable the button
  setButtonDisabled(true);

  // Optionally, show an alert
  Alert.alert('Button Clicked', 'Text set to "hello world", button disabled');
};

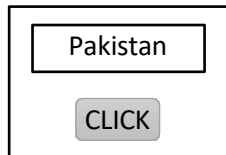
return (
  <View>
    { /* TextInput */}
    <TextInput
      style={{ height: 40, borderColor: 'gray', borderWidth: 1, marginBottom: 10, padding: 10 }}
      placeholder="Text will appear here"
      value={textValue}
      editable={false} // To make TextInput read-only
    />
    { /* Button */}
    <Button
      title="Click Me"
      onPress={handleButtonClick}
      disabled={buttonDisabled}
    />
  </View>
);
};

export default YourComponent;

```

## QUESTION 14

Suppose we have layout like this:



When the button is clicked, the message in the TextInput should display in a dialog. Write code for event handler of button.

```
import React, { useState } from 'react';
import { View, TextInput, Button, Alert } from 'react-native';

const YourComponent = () => {
  const [message, setMessage] = useState("");

  const handleButtonClick = () => {
    // Display the message in a dialog
    Alert.alert('Message', message);
  };

  return (
    <View>
      {/* TextInput */}
      <TextInput
        style={{ height: 40, borderColor: 'gray', borderWidth: 1, marginBottom: 10,
padding: 10 }}
        placeholder="Type your message here"
        value={message}
        onChangeText={(text) => setMessage(text)}
      />

      {/* Button */}
      <Button
        title="Show Message"
        onPress={handleButtonClick}
      />
    </View>
  );
};

export default YourComponent;
```



## QUESTION 15

Suppose you want to build a game in which a user either can press fire or wood, and then a random choice is generated for computer. The player that gets fire is the winner. Write the code of the program. In case both user and computer get same value, it will be a draw.

Your choice	<input type="text" value="fire"/>
Computer choice	<input type="text" value="wood"/>
Winner	<input type="text" value="user"/>

FIRE

WOOD

```
import React, { useState } from 'react';
import { View, Text, Button, Alert } from 'react-native';

const YourComponent = () => {
  const [userChoice, setUserChoice] = useState("");
  const [computerChoice, setComputerChoice] = useState("");
  const [winner, setWinner] = useState("");

  const handleChoice = (choice) => {
    // User's choice
    setUserChoice(choice);

    // Generate random choice for computer
    const computerChoices = ['fire', 'wood'];
    const randomIndex = Math.floor(Math.random() * computerChoices.length);
    const computerChoice = computerChoices[randomIndex];
    setComputerChoice(computerChoice);

    // Determine the winner
    if (choice === 'fire' && computerChoice === 'wood') {
      setWinner('You');
    } else if (choice === 'wood' && computerChoice === 'fire') {
      setWinner('Computer');
    } else {
      setWinner('Draw');
    }
  }
}
```

```

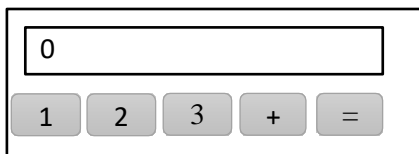
    // Show result in an alert
    Alert.alert('Game Result', `Your choice: ${choice}\nComputer choice:
    ${computerChoice}\nWinner: ${winner}`);
  };

  return (
    <View>
      <Text>Your choice: {userChoice}</Text>
      <Text>Computer choice: {computerChoice}</Text>
      <Text>Winner: {winner}</Text>
      <Button title="Fire" onPress={() => handleChoice('fire')} />
      <Button title="Wood" onPress={() => handleChoice('wood')} />
    </View>
  );
};
export default YourComponent;

```

## QUESTION 16

The following layout has three number buttons, a plus and equal operator, and a TextInput initialized with a zero “0”.



The user should be able to enter a string of numbers like 12232213. The user need to enter a number, click on + operator, and then input another number. When user click on equal, the result of sum should display in TextInput (Hint: Check eval method of javascript).

```

import React, { useState } from 'react';
import { View, TextInput, Button, StyleSheet } from 'react-native';

const CalculatorApp = () => {
  const [inputText, setInputText] = useState('0');

  const handleNumberClick = (number) => {
    // Update TextInput with the entered number
    setInputText((prevText) => (prevText === '0' ? number : prevText + number));
  };

```

```

const handlePlusClick = () => {
  // Append the plus operator to the existing text
  setInputText((prevText) => prevText + '+');
};

const handleEqualClick = () => {
  try {
    // Evaluate the expression using eval method
    const result = eval(inputText);
    setInputText(result.toString());
  } catch (error) {
    // Handle invalid expressions
    setInputText('Error');
  }
};

return (
  <View style={styles.container}>
    <TextInput style={styles.input} value={inputText} editable={false} />

    <View>
      <Button title="1" onPress={() => handleNumberClick('1')} />
      <Button title="2" onPress={() => handleNumberClick('2')} />
      <Button title="3" onPress={() => handleNumberClick('3')} />
    </View>

    <View>
      <Button title="+" onPress={handlePlusClick} />
      <Button title="=" onPress={handleEqualClick} />
    </View>
  </View>
);
};

```

## QUESTION 17

The capital of Pakistan is <TextInput>

KARACHI

LAHORE

ISLAMABAD

You have a layout as given in the following. You need to write a “single method” for all the three buttons. The prototype of method is:

**function button\_Click().**

In this method, you need to get the text of the button clicked. If the text is matching with the string “ISLAMABAD”, the <TextInput> should be assigned value ISLAMABAD, otherwise it remains blank.

```

import React, { useState } from 'react';
import { View, TextInput, Button, Text, StyleSheet } from 'react-native';

const YourComponent = () => {
  const [capital, setCapital] = useState("");
  const button_Click = (clickedText) => {
    if (clickedText === 'ISLAMABAD') {
      setCapital('ISLAMABAD');
    } else {
      setCapital("");
    }
  };
  return (
    <View style={styles.container}>
      <Text>The capital of Pakistan is</Text>
      <TextInput
        style={styles.input}
        value={capital}
        editable={false} // Make TextInput read-only
      />
      <View>
        <Button title="Karachi" onPress={() => button_Click('Karachi')} />
        <Button title="Lahore" onPress={() => button_Click('Lahore')} />
        <Button title="Islamabad" onPress={() => button_Click('ISLAMABAD')} />
      </View>
    </View> );
};

```

## REACT NATIVE STYLES (CLO-3)

### QUESTION 01

**Suppose you have an <Text> field and two buttons. The first button is labeled as BLUE and the second button is labeled as GREEN. When the BLUE button is clicked, the color of text in <Text> should changed to BLUE, and when GREEN button is clicked, the color of text in <Text> should change to GREEN. Write also the code of defining the style classes of two colors.**

```
import React, { useState } from 'react';

import { View, Text, Button, StyleSheet } from 'react-native';

const TextColorChanger = () => {

  const [textColor, setTextColor] = useState('black');

  const styles = StyleSheet.create({

    blueText: {

      color: 'blue',

    },

    greenText: {

      color: 'green',

    },

  });

  const handleColorChange = (color) => {

    setTextColor(color);

  };

  return (

    <View>

      <Text style={{ color: textColor }}>This is the colored text.</Text>

      <Button title="BLUE" onPress={() => handleColorChange('blue')} />

      <Button title="GREEN" onPress={() => handleColorChange('green')} />

    </View>

  );
}
```

```

    </View>

  );

};

export default TextColorChanger;

```

---

## QUESTION 02

**Suppose you have a layout like the above. In the below example, the blue button is clicked, and its text size is increased and text color is changed to black.**

**In the above layout, the buttons are touchable opacity. The buttons are created by using array of color names, and the text in the buttons is shown in upper case. When a button is clicked, the color of the text below is changed and the name of color is shown as shown in the above example. Moreover, the button that is clicked has font weight changed to bold and font size increased to indicate which button is currently clicked.**



```

import React, { useState } from 'react';
import { View, Text, TouchableOpacity, StyleSheet } from 'react-native';

const ColorButton = ({ color, onPress, isSelected }) => {
  const styles = StyleSheet.create({
    button: {
      backgroundColor: color,
      padding: 10,
      borderRadius: 5,
      margin: 5,
      alignItems: 'center',
    },
    buttonText: {
      color: isSelected ? 'black' : 'white',
      textTransform: 'uppercase',
      fontWeight: isSelected ? 'bold' : 'normal',
      fontSize: isSelected ? 16 : 14,
    },
  });

  return (
    <TouchableOpacity onPress={onPress} style={styles.button}>
      <Text style={styles.buttonText}>{color}</Text>
    </TouchableOpacity>
  );
};

```

```

const ColorChangerApp = () => {
  const [selectedColor, setSelectedColor] = useState("");

  const handleColorChange = (color) => {
    setSelectedColor(color);
  };

  return (
    <View>
      {/* Display the selected color name */}
      <Text style={{ color: selectedColor }}>{selectedColor}</Text>

      {/* Row of color buttons */}
      <View style={{ flexDirection: 'row' }}>
        <ColorButton color="red" onPress={() => handleColorChange('red')}
isSelected={selectedColor === 'red'} />
        <ColorButton color="blue" onPress={() => handleColorChange('blue')}
isSelected={selectedColor === 'blue'} />
        <ColorButton color="green" onPress={() => handleColorChange('green')}
isSelected={selectedColor === 'green'} />
      </View>
    </View>
  );
};

export default ColorChangerApp;

```

### Question: 03

Show a list of students, such that :

ID	Name	CGPA
1	Javed	3.0
2	Noman	2.7
3	Ali	3.7
4	Faisal	3.3
5	Shahid	4.0
6	Kamal	3.1
7	Zahid	2.3

The students whose CGPA are in the range between 2 and less than 3 should be shown in bold and red font.

The students whose CGPA are in the range between 3 and less than 3.7 should be shown in blue font without bold

The students whose CGPA are greater than and equal to 3.7 should be shown in italic, bold, and green font.

```
import React from 'react';
```

```
const StudentList = () => {  
  const students = [  
    { id: 1, name: 'Javed', cgpa: 3.0 },  
    { id: 2, name: 'Noman', cgpa: 2.7 },  
    { id: 3, name: 'Ali', cgpa: 3.7 },  
    { id: 4, name: 'Faisal', cgpa: 3.3 },  
    { id: 5, name: 'Shahid', cgpa: 4.0 },  
    { id: 6, name: 'Kamal', cgpa: 3.1 },  
    { id: 7, name: 'Zahid', cgpa: 2.3 },  
  ];  
  const getStudentStyle = (cgpa) => {  
    if (cgpa >= 2 && cgpa < 3) {
```



```

    return { color: 'red', fontWeight: 'bold' };
  } else if (cgpa >= 3 && cgpa < 3.7) {
    return { color: 'blue' };
  } else if (cgpa >= 3.7) {
    return { color: 'green', fontWeight: 'bold', fontStyle: 'italic' };
  }
  return {};
};

return (
  <div>
    <table>
      <thead>
        <tr>
          <th>ID</th>
          <th>Name</th>
          <th>CGPA</th>
        </tr>
      </thead>
      <tbody>
        {students.map((student) => (
          <tr key={student.id} style={getStudentStyle(student.cgpa)}>
            <td>{student.id}</td>
            <td>{student.name}</td>
            <td>{student.cgpa}</td>
          </tr>
        ))}
      </tbody>
    </table>
  </div>

```

```
);  
};  
export default StudentList;
```