PES University, Bengaluru

(Established under Karnataka Act No. 16 of 2013)

UE19CS204

OCTOBER 2020: IN SEMESTER ASSESSMENT B Tech 3rd SEMESTER
TEST – 1

UE19CS204 (4 credits) – WEB TECHNOLOGIES

Time: 2 Hrs	Answer All Questions	Max Marks: 60

1.	a)	Write an HTML page for displaying the Shopping Cart which consists of a table of Shopping Items and a Form to update the shopping cart as follows:  Note: Assume the styling is handled for the th tag in CSS, and write only the HTML code for the h2, table and form tags. Ignore the CSS styling and form properties as well. **Solution:** Shopping Cart <form> <table> <tr> <th>Quantity</th> <th>Item</th> <th>Order Code</th> <th>Unit Price</th> <th>Total</th> <th>Remove</th> </tr> <tr> <td><input type="text" value="2" name="qty"/></td> <td>My Product(HSD-KSE)</td> <td>HSD-KSE</td> <td>$9.99</td> <td>$19.98</td> <td><input type="checkbox" name="rmitem"/></td> </tr> </table> <input type="submit" value="Update Cart"/> </form> **h2 - 1 mark** **th - 1 mark** **td -> input-text - 1 mark** **td -> input-checkbox - 1 mark** **input-submit - 1 mark**	5
	b)	Write a line about the following position properties: static, relative, absolute, fixed, sticky **Solution:** static The element is positioned according to the normal flow of the document. relative The element is positioned according to the normal flow of the document, and then offset relative to itself based on the values of top, right, bottom, and left. absolute The element is removed from the normal document flow, and no space is created for the element in the page layout. Its final position is determined by the values of top, right,	5

bottom, and left.

fixed

Fixed positioning is similar to absolute positioning, used to create a "floating" element that stays in the same position regardless of scrolling.

sticky

Sticky positioning can be thought of as a hybrid of relative and fixed positioning. A stickily positioned element is treated as relatively positioned until it crosses a specified threshold, at which point it is treated as fixed until it reaches the boundary of its parent.

(1 mark each)

2. a)

College Reopen Date Guessing Game (guess the random date) — 5

Write a JavaScript code, which will generate a random date in the range 01-01-2021 and 31-12-2021. Now prompt the user to enter a date in that range. If the date matches the user wins, if not, show a message “The guess was greater than or less than reopen date.” appropriately.

Note: Write only the JavaScript code. Message can be shown in console or alert dialog. Use new date(yyyy,mm,dd) or new Date(datestr) to construct the date object.

Solution:

```
dt = Math.floor(Math.random()*30)+1
mn = Math.floor(Math.random()*11)
date = new Date(2021,mn,dt);

console.log(date)
ind = prompt("Enter Date in mm/dd/yyyy format","1/1/2021")
indate = new Date(ind)
console.log(indate)
if(date.getTime() == indate.getTime())
        console.log("Guess was right")
else if(date.getTime() < indate.getTime())
        console.log("Guess was more than random date")
else if(date.getTime() > indate.getTime())
        console.log("Guess was less than random date")
```

Random date – 2 mark

Input date – 1 mark

Date comparison – 2 marks

b) — 5

Consider the following object “chesspiece” with methods display and moveto.

```
function chesspiece(name, color, position){
        this.name = name;
        this.color = color;
        this.position = position;
}
chesspiece.prototype.display = function(){
        document.write(this.name+" "+this.color+" "+this.position+"<br/>");
}
chesspiece.prototype.moveto = function(newpos){
        console.log("Checking if "+newpos+" is valid...");
        this.position = newpos;
}
```

Write an object “king” which all the properties of a chesspiece and additionally has a flag called “castled”. Create an instance k1 of king with the values “king”, “black”, “E8” and false. Add a setcastled method, which checks if the position is “G8” then sets castled to true. To be able to call k1.moveto or k1.display, what needs to be done?

Solution:

```
function king(name, color, position, castled){
        chesspiece.call(this, name, color, position);
        this.castled = castled;
}

king.prototype = chesspiece.prototype;
king.prototype.constructor = king;
```

```
king.prototype.setcastled = function(){
        if(this.position == "G8")
                this.castled=true;
        }
let k1 = new king("king", "black", "E8", false);

k1.display();
k1.moveto("G8");
k1.setcastled();
k1.display();
document.write(k1.name+" "+k1.castled);
```

Constructor – 1 mark
Set prototype and constructor – 2 mark
setcastled – 1 mark
Instance creation – 1 mark

3. a) **Post it** — 6

Write JavaScript code that reads an input field with id “message” and show a “div” with the entered text message where the mouse is clicked anywhere on the page. Also, click on div should remove it without triggering the show div on the page.

Solution:

```
function init(){
        inp = document.getElementById("message");
        document.body.onclick=showdiv;
}
function showdiv(event){
        console.log("in showdiv")
        var div = document.createElement("div")
        div.innerHTML = inp.value;
        div.style.position="absolute";
        div.style.left=event.clientX;
        div.style.top=event.clientY;
        div.onclick=deldiv;
        document.body.appendChild(div)
}
function deldiv(event){
        event.stopPropagation(); //return false //event.cancelBubble=true
        event.target.remove();
}
```

onclick – 1 mark
div create – 1 mark
div style properties – 2 marks
appendChild – 1 mark
stopPropagation and remove – 1 mark

b) Write briefly (3-4 lines) about the significance of specifying multiple source tags in video/audio tag. — 4

Solution:

The <source> tag is used to specify multiple media resources for media elements, such as <video> and <audio>.

The <source> tag allows you to specify alternative video/audio files which the browser may choose from, based on browser support. The browser will choose the first <source> it supports.

(4 marks)

4. a) Write jQuery code to accomplish the following: — 1 + 2 + 2

1. For the paragraphs within the div with id “contents”, set the color to red.
2. On moving the mouse over any table cell, the text font size should be increased by 2 times
3. When the first image on the page is clicked, it fades out in 3 seconds and then fades in 3 seconds

Solution:

1. $("div#contents p").css("color","red") **– 1 mark**
2. $("td").mouseover(function(){
   $(this).animate({
   fontSize: 2em
   }
   } **– 2 marks**
3. $("img:first").click(function(event){
   $(this).fadeOut(3000,function(){
   $(this).fadeIn(3000, function(){});
   });
   }); **– 2 marks**

b) Read a json at location http://localhost/student.json in the format {"srn":"123", "marks":[1,2,3,4,5]} and display the avg marks — 5

Write only the function to send the asynchronous request using XHR and the function to process the response.

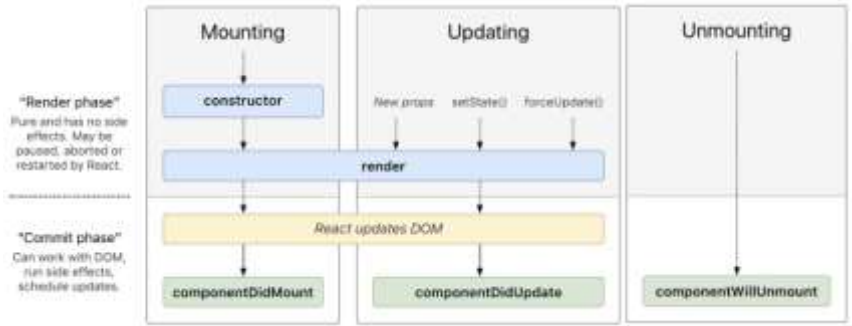
Solution:

```
function getdata(){
    xhr = new XMLHttpRequest();
    xhr.open("get", "http://localhost/sample.json",true);
    xhr.onreadystatechange = showdata;
    xhr.responseType = "json";
    xhr.send();
}
function showdata(){
    if(this.readyState == 4 && this.status == 200){
        var m = this.response['marks'];//this.response.marks
        var avg = 0, sum = 0;
        for (i in m)
            sum += m[i]
        avg = sum / m.length
        document.querySelector("#container").innerHTML = "Avg: "+avg
    }
}
```

open, onreadystatechange, responseType, send and this.response['marks'] **– 1 mark each**

5. a) Write briefly about the React Components Life Cycle with a neat diagram. — 4

Solution:



componentDidMount,
componentDidUpdate,
componentWillUnmount **– 1 mark each**
diagram **– 1 mark**

b) An App component consists of two input fields, Quantity and Unit Price, a Submit button and a placeholder div to show the total price. When quantity and unit price is entered and submit is clicked, the total price should be displayed in a placeholder div. Write the React JSX code using Controlled Form (state and onClick) — 6

Solution:

```
class Calc extends React.Component{
    constructor(props){
        super(props);
```

```
                this.state={
                        quantity:0,
                        unitprice:0,
                        total:0,
                }
                this.handleSubmit=this.handleSubmit.bind(this);
                this.handleChange=this.handleChange.bind(this);
        }
        handleChange=(event)=>{
                var name=event.target.name;
                var value=event.target.value;
                this.setState({
                        [name]:value
                })
        }
        handleSubmit=function(event){
                var quantity = parseInt(this.state.quantity);
                var unitprice=this.state.unitprice-0;
                this.state.total=quantity*unitprice;
                this.statusoutput.innerHTML="Total: "+this.state.total;
                event.preventDefault();
        }
        render(){
           return(
              <div>
                 <form onSubmit={this.handleSubmit}>
                    <label>Quantity:</label>
                    <input name="quantity" value={this.state.quantity} type="text"
onChange={this.handleChange}/>
                    <label>Unit Price: </label>
                    <inputname="unitprice" value={this.state.unitprice} type = "text" onChange
= {this.handleChange}/>
                    <input type="submit" value="submit"/>
                  </form>
                  <div ref={(el)=>{this.statusoutput=el}}></div>
              </div>
            ) } }
ReactDOM.render (<Calc/>, document.querySelector("#container"))
```

constructor with initial state – 1 mark
handleChange – 1 mark
handleSubmit – 2 marks
form, input and div with ref – 2 marks

6 a) Image Zoom In/Out (increase/decrease height and width) — 10

An App component has an image, zoomin and zoomout buttons. When zoomin or zoomout button is clicked the image height and width is increased or decreased by 10 pixels. The initial values for height and width are set to 200 and 100 pixels respectively. Use state, click event and ref to accomplish this functionality. Note: Image src property should be passed from a property in the App Component.

Solution:

```
        class App extends React.Component {
                constructor(props){
                        super(props);
                        this.state={
                                height:200,
                                width:100
                        }
//two functions for zoomin and zoomput accepted
                        this.doClick=(event)=>{
                                var sign;
```

```
                                    if(event.target.innerHTML == "ZoomIn")
                                            sign=1
                                    else
                                            sign=-1
                                    this.setState((prevState)=>{
//without prevState also accepted
                                            return({
                                                    height:prevState.height+sign*10,
                                                    width:prevState.width+sign*10
                                            });
                                    })
                            }
                    }

                    componentDidUpdate(){
                            this.image.style.height = this.state.height;
                            this.image.style.width = this.state.width;
                    }
                    render(){
                            var imgstyle={
                                    height:this.state.height,
                                    width:this.state.width
                            }
                            return (
                                    <div>
                                       <img style={imgstyle} src={this.props.src} ref= { (el) =>
                                            {this.image=el}}/>
                                       <button onClick={this.doClick}>ZoomIn</button>
                                       <button onClick={this.doClick}>ZoomOut</button>
                                    </div>
                            ) } }
            ReactDOM.render(<App src="tiger.jpg"/>, document.querySelector("#container"))
```

render app with src prop – 1 mark
image tag with style and src and ref – 3 marks
constructor with initial style – 2 marks
initial style – 1 mark
setState with prevState – 3 marks