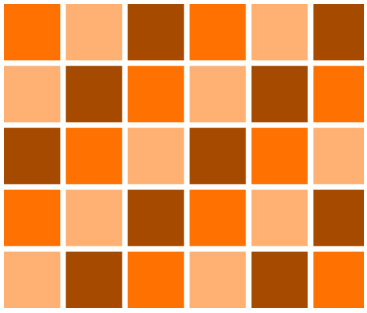
FED PROGRAMS

1. Try to recreate the following patterns using HTML and CSS only.



Program:

```
<html>
    <head>
        <style>
            .container{
                display: grid;
                grid-template-columns: repeat(6,25px);
                grid-template-rows: repeat(5,25px);
                grid-gap : 2px;
            }
            .a1{
                background-color: rgb(244, 105, 13);
            }
            .a2{
                background-color: rgb(242, 174, 128);
            }
            .a3{
                background-color: rgb(127, 82, 51);
            }
        </style>
        <title>CSS Grid</title>
    </head>
    <body>
        <div class="container">
            <div class="a1"></div>
            <div class="a2"></div>
            <div class="a3"></div>
            <div class="a1"></div>
            <div class="a2"></div>
            <div class="a3"></div>

            <div class="a2"></div>
            <div class="a3"></div>
            <div class="a1"></div>
            <div class="a2"></div>
            <div class="a3"></div>
            <div class="a1"></div>
```

```
            <div class="a3"></div>
            <div class="a1"></div>
            <div class="a2"></div>
            <div class="a3"></div>
            <div class="a1"></div>
            <div class="a2"></div>

            <div class="a1"></div>
            <div class="a2"></div>
            <div class="a3"></div>
            <div class="a1"></div>
            <div class="a2"></div>
            <div class="a3"></div>

            <div class="a2"></div>
            <div class="a3"></div>
            <div class="a1"></div>
            <div class="a2"></div>
            <div class="a3"></div>
            <div class="a1"></div>
        </div>
    </body>
</html>
```

Output:

2. Implement Drag n Drop feature in HTML 5

Program:

```
<html>
    <body>
        <h1>this is an example of drag and drop</h1>
        <div style="height:200;width:250;border:solid 1px red"
        ondragover="allowdrop(event)" ondrop="drop(event)"></div>
        <br><br>
        <img src="main.jpeg"
        width="250" height="180" alt="image" draggable="true"
```

```
        ondragstart="drag(event)" id="img"/>
        <script>
            function drag(event){
                event.dataTransfer.setData("text",event.target.id)
            }
            function allowdrop(event){
                event.preventDefault()
            }
            function drop(event){
                event.preventDefault()
                var data=event.dataTransfer.getData("text")
                event.target.appendChild(document.getElementById(data))
            }
        </script>
    </body>
</html>
```

Output:

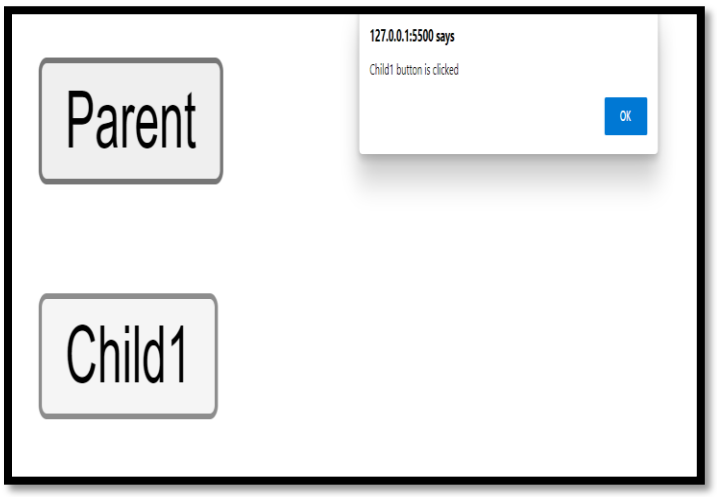
3. Demonstrate Event bubbling with necessary examples.

Program:

```
<html>
    <head>
        <title>Event Bubbling</title>
    </head>
    <body>
        <div id="parent">
            <button>Parent</button> <br> <br>
            <button id="child1">Child1</button>
```

```
        </div>

        <script>
            document.getElementById("parent").addEventListener('click', function(){
                alert("Parent button is clicked")
            })
            document.getElementById("child1").addEventListener('click', function(){
                alert("Child1 button is clicked")
            })

        </script>
    </body>
</html>
```

Output:



4.Design a Calculator using Java script and relevant CSS.

(	CE	)	C
1	2	3	+
4	5	6	-
7	8	9	x
.	0	=	÷

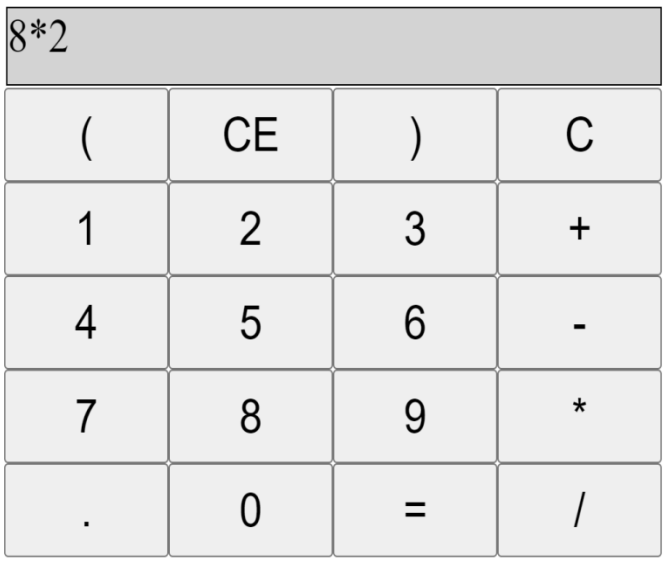
Program:

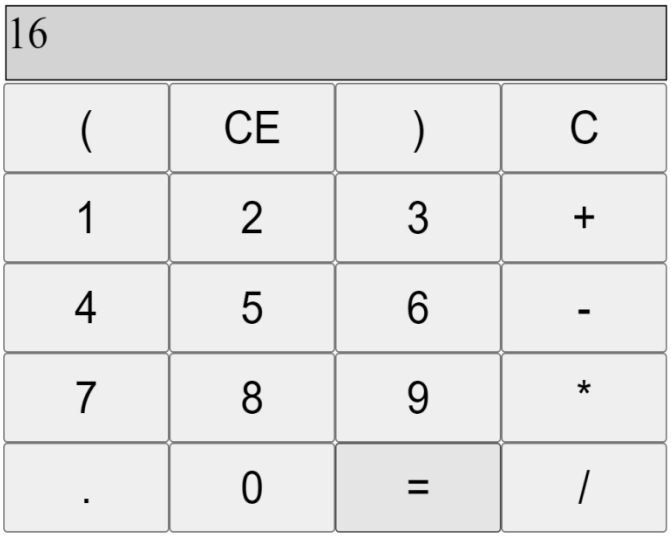
```
<html>
    <head>
    <style>
        .container {
            display:grid;
            grid-template-columns: 100px 100px 100px 100px;
            grid-template-rows: 50px 50px 50px 50px 50px;
        }
        div div{
            margin:2px 2px;
            border:2px solid black;
        }
        .a1{
            text-align:center;
            font-size:25px;
        }
        .a2{
            margin:1px 1px;
            border:1px solid black;
            width:397px;
            height:40px;
            background:lightgrey;
            font-size:25px;
        }
    </style>
    </head>
    <body>
        <div class="a2" id="screen"></div>
        <div class="container">

            <button class="a1" id="b1" onclick="sendNum('(')">(</button>
            <button class="a1" id="b2" onclick="clearScr()">CE</button>
            <button class="a1" onclick="sendNum(')')">)</button>
            <button class="a1" onclick="c()">C</button>
            <button class="a1" onclick="sendNum('1')">1</button>
            <button class="a1" onclick="sendNum('2')">2</button>
            <button class="a1" onclick="sendNum('3')">3</button>
            <button class="a1" onclick="sendNum('+')">+</button>
            <button class="a1" onclick="sendNum('4')">4</button>
            <button class="a1" onclick="sendNum('5')">5</button>
            <button class="a1" onclick="sendNum('6')">6</button>
            <button class="a1" onclick="sendNum('-')">-</button>
            <button class="a1" onclick="sendNum('7')">7</button>
            <button class="a1" onclick="sendNum('8')">8</button>
            <button class="a1" onclick="sendNum('9')">9</button>
            <button class="a1" onclick="sendNum('*')">*</button>
            <button class="a1" onclick="sendNum('.')">.</button>
            <button class="a1" onclick="sendNum('0')">0</button>
```

```
            <button class="a1" onclick="equalTo()">=</button>
            <button class="a1" onclick="sendNum('/')">/</button>

        </div>
    </body>
    <script>

        var a = '';
        var b = '';
        var num = [];
        var ans;

        function sendNum(digit){

            num.push(digit);

            if(num.length != 1){
                a = '';
                document.getElementById('screen').innerHTML = a;        // clearing the
screen.
            }


            for(i=0; i<num.length ; i++){

                a = a + num[i];

            }

            document.getElementById('screen').innerHTML = a;    // displaying the
concatenated string



        }

        function c(){

            num.pop();
            a='';
            document.getElementById('screen').innerHTML =a;
            for(i=0; i<num.length ; i++){

                a = a + num[i];

            }

            document.getElementById('screen').innerHTML = a;
        }

        function equalTo(){
            document.getElementById('screen').innerHTML = '';
```

```
            for(i=0; i<num.length ; i++){

                b += num[i];
            }

            ans = eval(b);

            document.getElementById('screen').innerHTML = ans;

            while(num.length > 0){
                num.pop();
            }

            num.push(ans.toString());
            a=ans;

        }


        function clearScr(){
            document.getElementById('screen').innerHTML = '';

            while(num.length > 0){
                    num.pop();
            }

            a ='';
            b ='';
        }
    </script>
</html>
```

Output:





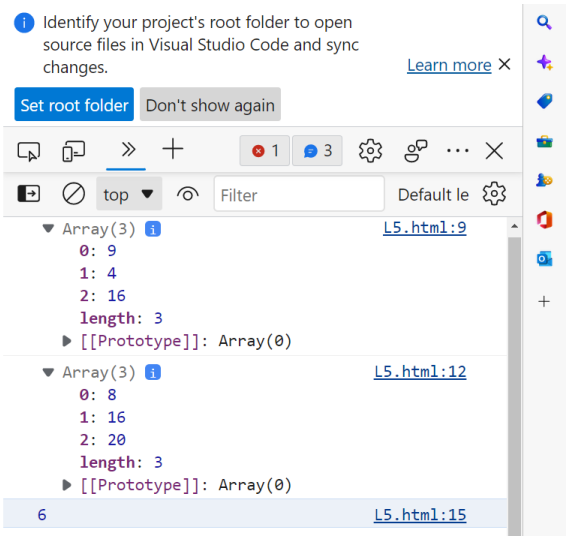
5. Demonstrate Higher order functions with necessary examples – filter(), reduce() and map()

Program:

```
<html>
    <head>
        <title>Higher Order Functions Example</title>
    </head>
    <body>
    <script>
    let a=[3,2,4]
    let b=a.map((x)=>{return x*x})
console.log(b)
    let a1=[8,16,20]
    let c=a1.filter((x)=>{return x%2==0})
    console.log(c)
    let a2=[1,2,3]
    let d=a2.reduce((sum,i)=>{return sum+i})
    console.log(d)

     </script>
     </body>
</html>
```

Output:

To get the o/p PRESS ctrl , shift ,j



6. Create a Class Component for Counter in React JS

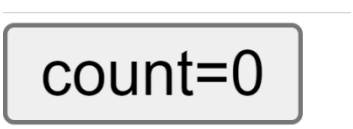
Counter.js:

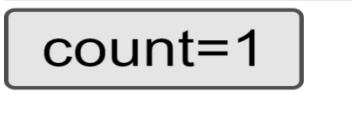
```
import React,{Component} from 'react';
  class Counter extends Component{
    constructor(props){
        super(props)
        this.state={count:0}
    }
    render(){
        return(<button onClick={()=>this.setState
            ({count:this.state.count+1})}>count={this.state.count}
            </button>)
    }
}

export default Counter;
```

APP.JS

```
import Counter from "./components/Counter";

function App() {
  return (
    <div>

      <Counter/>
    </div>

  );
}
```

Output:





Every time we click on the count button it increases by 1

7. Create a Class component for Changing the color of the text given in React JS

ColorChange.js :

```
import React,{Component} from 'react';
class Colorchange extends Component{
    constructor(props)
    {
        super(props);
        this.state={color:"red"}
    }
    handlecolor=()=>{
        this.setState({color:"green"})
    }
    handlecolor1=()=>{
        this.setState({color:"blue"})
    }

    render(){
        return(<div>
                <h1 onMouseOver={this.handlecolor} onMouseOut={this.handlecolor1}
                style={this.state}>WELCOME</h1>
        </div>)
    }
}

export default Colorchange;
```

APP.JS

```
import Colorchange from "./components/Colorchange";


function App() {
  return (
    <div>
      <Colorchange />
    </div>

  );
}

export default App;
```

Output:

WELCOME

WELCOME

WELCOME

8. Class a Class Component for viewing an array of objects in a tabular form.

Arraytable.js

```
import React,{Component} from 'react';
class Arraytable extends Component{
    constructor(props)
    {
        super(props)
        this.state=[{id:101,name:"alice",age:20},
                    {id:102,name:"john",age:23},
                    {id:103,name:"bob",age:19}];
    }
    render()
    {
        return(
            <div><br></br>
            <h3 align = "center">Array values in tabular form</h3>
                <table border="2px solid black" align="center">
                    <tr>{Object.keys(this.state[0]).map((x) => {return <th>{x}</th>})}</tr>
                    {this.state.map((ob)=>(<tr>{Object.values(ob).map((y)=>{return
<td>{y}</td>})}</tr>))}
                </table>
            </div>
        );
    }
}
export default Arraytable;
```

APP.JS

```
import Arraytable from "./components/Arraytable";


function App() {
  return (
    <div>
      <Arraytable />
    </div>
```

```
  );
}
```

Output:

Array values in tabular form

id	name	age
101	alice	20
102	john	23
103	bob	19

9. Display a digital clock in React JS.

Clock.js

```
import React,{Component} from 'react';
class Clock extends Component
{
    constructor(props)
    {
        super(props);
        this.state={time:""};
    }
    componentDidMount()
    {
        this.tick();
    }
    tick=()=>
    {
        const hours=new Date().getHours();
        const min=new Date().getMinutes();
        const sec=new Date().getSeconds();
        const updateTime=`${hours}:${min}:${sec}`;
        this.setState({time:updateTime})

    };
    componentDidUpdate(){
        this.interval=setInterval(()=>{this.tick();},1000);
```

```
    }
    componentWillUpdate()
    {
        clearInterval(this.Interval);
    }
    render(){
        return(
            <div>{this.state.time}</div>
        )
    }
}
export default Clock;
```

App.js

```
import Clock from "./components/Clock";
function App() {
  return (
    <div>

      <Clock />
    </div>

  );
}

export default App;
```

Output:

21:17:17

10. Demonstrate useState Hook with the help sample text.

Statehook.js :

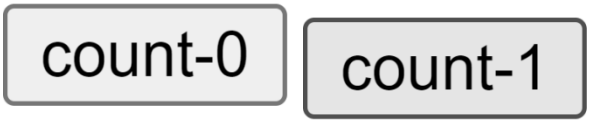
```
import React, { useState } from "react";
function Statehook()
{
    const [count,setCount] = useState(0)
    return(<div>
        <button onClick={()=>setCount(count+1)}>count-{count}</button>
```

```
    </div>);
}
export default Statehook;
```

APP.JS

```
import Statehook from "./Statehook";


function App() {
  return (
    <div>
      <Statehook/>
    </div>

  );
}

export default App;
```

Output: Every time we click on the count button it increases by 1



11. Demonstrate useContext Hook with necessary example.

UseContextHook.js

```
import React , { createContext, useContext, useState } from "react";
import Child1 from "./Child1";

import Child2 from "./Child2";
 export const global = createContext();
function UseContextHook()
{
    return(<global.Provider value="JESSY">
        <h1>My name is {"JESSY"}.</h1>
        <Child1 />
    </global.Provider>
    );
}
export default UseContextHook;
```

Child1.js

```
import React from "react";
import Child2 from "./Child2";

function Child1()
{
    return(<div>
        <h1>Child Component1.</h1>
        <Child2 />
    </div>

    )
}
export default Child1;
```

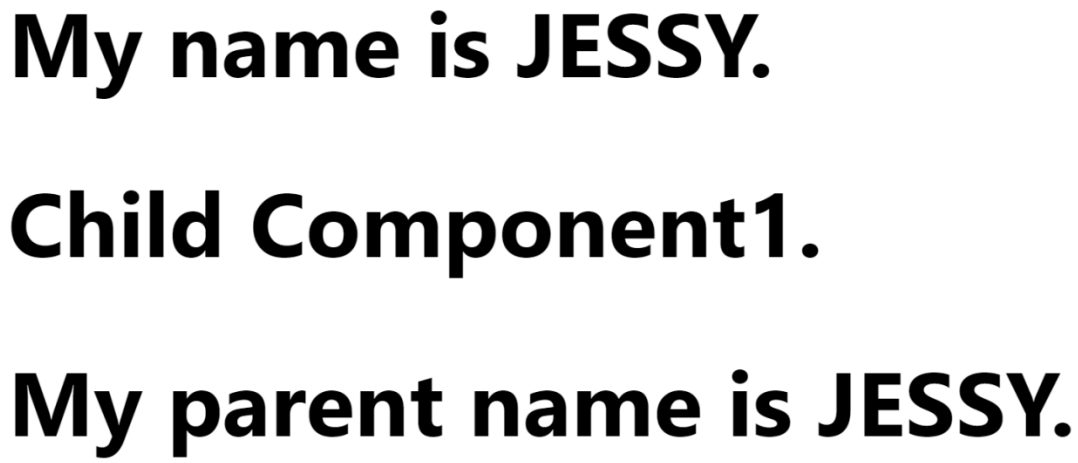
Child2.js

```
import React, { useContext } from "react";
import { global } from "./UseContextHook";
function Child2()
{
    const name = useContext(global);
    return(<div>
        <h1>My parent name is {name}.</h1>
    </div>

    )
}
export default Child2;
```

App.js

```
import UseContextHook from "./components/UseContextHook";
function App() {
  return (
    <div>

      <UseContextHook/>
    </div>

  );
}

export default App;
```

Output:



12. Demonstrate useEffect Hook with necessary example

UseEffectHook.js

```
import React,{useEffect, useState} from 'react'
function UseEffectHook(){
    const[count,setcount]=useState(0)
    const[flag,setflag]=useState('true')
    const handleflag=()=>{
        if(flag=='true')
           setflag('false')
        else
           setflag('true')
    }
    useEffect (()=>{setcount(count+1)},[flag])
    return(
        <div>
            <button onClick={handleflag}>flag:{flag}</button>
            <p>flag change {count} no of times</p>
        </div>
    );
}
export default UseEffectHook
```

APP.js

```
mport UseEffectHook from "./components/UseEffectHook";


function App() {
```

```
  return (
    <div>

      <UseEffectHook/>
    </div>

  );
}

export default App;
```

Output: