Day-20 Morning Assessment

Props:

1. function UserCard({name, age)}{

return <p>{name} is {age} </p>

}

1. function HobbiesList({[“singing”, “drawing”, “sleeping”]}){

return (

<ul>

{hobbies.map((hobby, h) => <li key={h}>{hobby}</li>)}

</ul>

);

}

1. function Button({label, color}){

return <button style={{backgroundColor: color}}>{label}</button>;

}

1. function Profile({user}){

return (

<p>{user.username}</p>

<p>{user.email}</p>

);

}

1. function ClickButton({func()}){

return <button onClick={() => alert(“Button clicked!”)}>Alert</button>

}

1. function Greeting({time}){

return <p>{time>12 ? “Good Evening” : “Good Morning”}</p>

}

1. function Counter({value}){

const [count, setCount] = useState(start);

return (

<>

<p>{count}</p>

<button onClick={() =>setCount(count+1)}>Increase</button>

</>

);

}

1. function Avatar({src}){

return (<div>

<img ={src} alt=”image”/>

</div>

);

}

1. function Card({title, children}){

return (

<div style={{border:”1px solid red”, margin:1, padding:1}}>

<h4>{title}</h4>

{children}

</div>

);

}

1. function Product({price, discount}){

const discounted\_price = price-(price\*discount)/100;

return(

<p>The discounted price: {discounted\_price}</p>

);

}

Hooks:

1. function Timer(){

const [seconds,setSeconds] = useState(0);

useEffect(()=>{

const interval = setInterval() => setSeconds(s=>s+1), 1000);

return () => clearInterval(interval);

}, []);

return <p>{seconds}s</p>;

}

1. function MouseTracker(){

const [pos, setPos] = useState({x: 0, y: 0});

useEffect(() => {

const handleMove = e => setPos({x: e.clientX, y: e.clientY});

window.addEventListener(“mousemove”, handleMove);

return () => window.removeEventListener(“mousemove”, handleMove);

}, []);

return <p>X: {pos.x}, y: {pos.y}</p>;

}

1. function Focus(){

const inputRef = useRef();

return (

<>

<input ref = {inputRef} />

<button onClick={() => inputRef.current.focus()}>Focus</button>

</>

);

}

1. function Form(){

const [values, setValues] = useState(0);

return (

<>

<input value= {form.value}

onChange={e => setValues({…form, values : e.target.value})}

/>

<p>The value of the form is {form.value}</p>

</p>

);

}

1. function ThemeSwitcher(){

const [theme, setTheme] = useState(localStorage.getItem(“theme”));

useEffect(() => {

localStorage.setItem(“theme”,theme);

document.body.style.background = theme === “light” ? “#fff” : “#333”;}, [theme]);

return (

<button onClick={() => setTheme(theme === “light” ? “dark” : “light”)} > Toggle Theme </button>

);

}

1. function ReducerCounter(){

function reducer(state, action){

switch (action.type){

case “increment” : return state+1;

case “decrement” : return state-1;

case “reset” : return 0;

default : return state;

}

}

const [count, dispatch] = useReducer(reducer, 0);

return (

<>

<p>{count}</p>

<button onClick={() => dispatch({ type: “increment})}>+</button>

<button onClick={() => dispatch({ type: “decrement”})}>-</button>

<button onClick={() => dispatch({type: “reset”})}>Reset>/button>

</>

);

}

1. function Prime({limit}){

const primes = useMemo(() => {

const isprime = num => {

for (let i = 2; i <= Math.sqrt(num); i++){

if num%i == 0 return false;

}

return num > 1;

};

return Array.from({length: limit}, (\_, i) => i).filter(isPrime);

}; [limit]);

return <p>{primes.join(“,”)}</p>;

}

1. function WindowSize(){

const [size, setSize] = useState({w: window.innerWidth, h: window.innerHeight});

useEffect(() => {

const handleResize = () => setSize({w: window.innerWidth, h: window.innerHeight});

window.addEventListener(“resize”, handleResize);

return () => window.removeEventListener(“resize, handleResize);

}, []);

return <p>{size.w} x {size.h}</p>;

}

1. function StopWatch(){

const [time, setTime] = useState(0);

const [running, setRunning] = useState(false);

useEffect(() => {

let interval;

if (running) interval = setInterval(() => setTime(t => t+1), 1000);

}, [running]);

return (

<>

<p>{time}s</p>

<button onClick={() =>setRunning(true)}>Start</button>

<button onClick={() =>setRunning(false)}>Stop</button>

<button onClick={() =>setTime(0)}>Reset</button>

</>

);

}

1. function DarkModeToggle(){

const[dark, setDark] = useState(false);

useEffect(() => {

document.body.style.background = dark ? “8000” : “#fff”;

}, [dark]);

return <button onClick=>{() => setDark(d => !d)}>Toggle Dark Mode</button>;

}

State:

1. function Counter(){

const [count, setCount] = useState(0);

return(

<>

<p>{count}</p>

<button onClick{()=>setCount(count+1)}>+</button>

<button onClick{() => setCount(count-1)}>-</button>

</>

);

}

1. function Switch(){

const [switch, setSwitch] = useState(false);

return <button onClick={() => setSwitch(switch)}>{ switch ? “OFF” : “ON”}</button>

}

1. function Form(){

const [info, setInfo] = useState({name: “”, email: “”, age: “”});

return (

<>

<input placeholder=”Name” onChange=(e => setInfo({…info, name: e.target.value})}/>

<input placeholder=”email” onChange=(e => setInfo({…info, email: e.target.value})}/>

<input placeholder=”Age” onChange=(e => setInfo({…info, age: e.target.value})}/>

</>

);

}

1. function RandomQuote(){

const quotes = [“Smile everyday like it’s your birthday”, Be young, Be wild”, “Young Forever”, Stars shine brightest at the darkest night”];

const [quote, setQuote] = setState(quotes(0)];

return <button onClick{() => setQuote(quotes[Math.floor(math.random()\*quotes.length)])}>{quotes}</button>;

}

1. function Todo(){

const [todo, setTodo] = useState([]);

const [input, setInput] = useState(“”);

return (

<>

<input value = {input} onChange={e => etInput(e.target.value)} />

<button onClick={() => {setTodos([…todos, input]); setInput(“”); }}>Add</>

<ul>{todo.map((t,i) => <li key={i}>{t}</li>)}</ul>

</>

);

}

1. function UpperCaseInput(){

const [text. selText] = useState(“”);

return (

<input value = {text} onChange={e => setText(e.target.value.toUpperCase())} />

);

}

1. function like(){

const [likes, setLikes] = useState(0);

return <button onClick ={() => setLikes(likes+1)}> Like {likes}</button>;

}

1. function ColorPicker(){

const[color, setColor] = useState(“#ffffff”);

return (

<>

<input type=”color” value={color} onChange={e => setColor(e.target.value)} />

<div style={{height: “100vh”, background: color}}></div>

</>

);

}

1. function ImageCarousel(){

const images = [“img1.jpg”, “img2.jpg”, “img3.jpg”];

const [index, setIndex] = useState(0);

return (

<>

<img src = {images[index]} alt= “” width={200) />

<button onClick = {() => setIndex((index-1+images.length) % images.length)}>Prev</button>

<button onClick => {() => setIndex((index+1)%images.length)}>Next</button>

</>

);

}

1. function CharCounter(){

const [text, setText] = useState(“”);

return (

<>

<textarea value= {text} onChange={e =>setText(e.target.value)} />

<p>{text.length} characters</p>

</>

);

}