Кислый Илья 107а1

\\1

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

function Timer() {

const [count, setCount] = useState(10);

useEffect(() => {

const interval = setInterval(() => {

setCount(prevCount => prevCount - 1);

}, 1000);

return () => clearInterval(interval);

}, []);

useEffect(() => {

if (count === 0) {

clearInterval(interval);

}

}, [count]);

return (

<div>

<h1>Timer: {count}</h1>

</div>

);

}

export default Timer;

\\2

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

function Timer() {

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

const intervalRef = useRef(null);

const startTimer = () => {

intervalRef.current = setInterval(() => {

setCount(prevCount => prevCount + 1);

}, 1000);

};

const stopTimer = () => {

clearInterval(intervalRef.current);

};

return (

<div>

<h1>Timer: {count}</h1>

<button onClick={startTimer}>Start Timer</button>

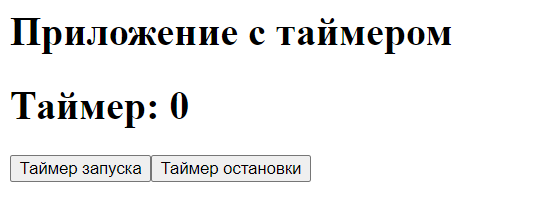
<button onClick={stopTimer}>Stop Timer</button>

</div>

);

}

export default Timer;



\\3

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

function PrimeNumbers() {

const [primeList, setPrimeList] = useState([]);

useEffect(() => {

const interval = setInterval(() => {

const nextPrime = findNextPrime(primeList.length > 0 ? primeList[primeList.length - 1] : 1);

setPrimeList(prevList => [...prevList, nextPrime]);

}, 1000);

return () => clearInterval(interval);

}, [primeList]);

const isPrime = (num) => {

if (num <= 1) return false;

if (num <= 3) return true;

if (num % 2 === 0 || num % 3 === 0) return false;

let i = 5;

while (i \* i <= num) {

if (num % i === 0 || num % (i + 2) === 0) return false;

i += 6;

}

return true;

};

const findNextPrime = (num) => {

let next = num + 1;

while (!isPrime(next)) {

next++;

}

return next;

};

return (

<div>

<h1>Prime Numbers:</h1>

<ul>

{primeList.map((prime, index) => (

<li key={index}>{prime}</li>

))}

</ul>

</div>

);

}

export default PrimeNumbers;

\\4

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

function TrafficLight() {

const [color, setColor] = useState("red");

useEffect(() => {

const interval = setInterval(() => {

switch (color) {

case "red":

setColor("yellow");

setTimeout(() => setColor("green"), 2000); // Желтый на 2 секунды, затем зеленый

break;

case "yellow":

setColor("red");

setTimeout(() => setColor("yellow"), 1000); // Красный на 1 секунду, затем желтый

break;

case "green":

setColor("yellow");

setTimeout(() => setColor("red"), 2000); // Зеленый на 2 секунды, затем красный

break;

default:

break;

}

}, 5000); // Каждые 5 секунд

return () => clearInterval(interval);

}, [color]);

return (

<div>

<div style={{ width: "100px", height: "100px", backgroundColor: "red", borderRadius: "50%", margin: "10px", boxShadow: color === "red" ? "0 0 20px red" : "none" }}></div>

<div style={{ width: "100px", height: "100px", backgroundColor: "yellow", borderRadius: "50%", margin: "10px", boxShadow: color === "yellow" ? "0 0 20px yellow" : "none" }}></div>

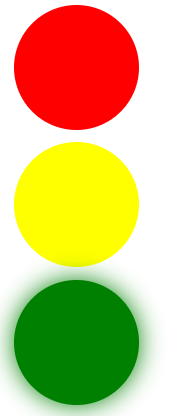
<div style={{ width: "100px", height: "100px", backgroundColor: "green", borderRadius: "50%", margin: "10px", boxShadow: color === "green" ? "0 0 20px green" : "none" }}></div>

</div>

);

}

export default TrafficLight;



\\5

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

function Revert({ s }) {

const [revertedString, setRevertedString] = useState(s);

useEffect(() => {

const interval = setInterval(() => {

setRevertedString(prevString => {

if (prevString.length <= 1) {

return prevString; // Если строка содержит один символ или меньше, нет необходимости в перестановке

}

const lastChar = prevString.charAt(prevString.length - 1); // Получаем последний символ строки

const newString = lastChar + prevString.slice(0, -1); // Переносим последний символ в начало строки

return newString;

});

}, 1000); // Каждую секунду

return () => clearInterval(interval);

}, []);

return (

<div>

{revertedString}

</div>

);

}

export default Revert;