**Question- React component that shows the current time to the user.**

import {useEffect, useState} from 'react';

function Time(){

const [date, setDate] = useState(new Date());

useEffect(() => {

const timer = setInterval(() => setDate(new Date()), 1000)

return () => clearInterval(timer);

});

return (

<p>{`${date.toLocaleTimeString()}`}</p>

);

}

export default Time;

**Question- JavaScript that shows the current time to the user.**

var time = null;

function start(){

time = setInterval(

() => {

const d = new Date();

console.log(d.toTimeString());

}

, 1000);

}

function stop(){

clearInterval(time);

}

**Output of this?**

console.log(1);

function foo() {

console.log(2);

setTimeout(() => console.log('3/1'), 1000)

setTimeout(() => console.log('3/0'), 0)

console.log(4);

}

console.log(5);

foo();

console.log(6);

1

5

2

4

6

3/0

3/1000

let num = 1;

const interval = setInterval(

() => {

if(num <=5){

console.log(num);

num += 1;

}else{

console.log('Inside Here')

clearInterval(interval);

}

}

, 1000);

// 1 2 3 4 5 Inside Here

const person = {

firstName: 'saranj',

print: function(){

console.log(this.firstName); // this => person object

},

print2: () => {

console.log(this.firstName); // this => global object = window

}

}

person.print(); // saranj

person.print2(); // undefined

***Everything in JavaScript is an object***

function foo(param) {

const name = "Bar";

const x = 'saranj';

console.log(name, param);

}

foo('muku') // bar muku

foo.age = 50;

// attached a user defined attribute to JavaScript object (i.e., function)

console.log(foo.x); // undefined

console.log(foo.name); // foo

// return name of function (inbuild name method)

console.log(foo.age); // 50

//return user defined attribute

console.log(foo.prototype); // {constructor: f}

// to check all the methods and attributes of function foo

foo.name = 12; // no update will happen to inbuild name property

console.log(foo.name); // foo

foo.age= 1243; // update user defined age property

console.log(foo.age); // 1243

function createUser(name) {

return {

name,

greet: function () {

console.log(`Hello ${this.name}`)

}

}

}

const user1 = createUser("Foo")

const user2 = createUser("Bar")

console.log(user1) // {name: 'Foo', greet: f {}}

console.log(user2) // {name: 'Bar', greet: f {}}

user1.greet(); // hello Foo

user2.greet(); // hello Bar

***The Object.create() method creates a new object using the prototype of the given object.***

let Student = {

name: "Lisa",

age: 24,

marks: 78.9,

display() {

console.log("Name:", this.name);

}

};

// create object from Student prototype

let std1 = Object.create(Student);

std1.name = "saranj";

std1.display(); // Name: saranj

console.log(std1.age) // 24

console.log(std1.marks) // 78.9

console.log(std1) // {name: ‘saranj’}

function createUser(name) {

const userObj = Object.create(userFunctionStore);

userObj.name = name;

return userObj;

}

const userFunctionStore = {

greet: function() {

console.log(`Hello ${this.name}`)

},

}

const user1 = createUser("Foo")

console.log(user1) //{name: 'Foo'}

console.log(user1.name) // Foo

user1.greet() // Hello Foo

***Exact Center div***

**Using flex layout module**

.parent {

  display: flex;

  justify-content: center;

  align-items: center;

  width: 100%;

  height: 100vh;

}

**Using grid layout module**

.parent {

  display: grid;

  width: 100%;

  height: 100vh;

  place-items: center;

}

[How to Center a Div Using CSS Grid — SitePoint](https://www.sitepoint.com/css-grid-center-element/#:~:text=Setting%20Up%201%201.%20Center%20a%20Div%20with,Center%20a%20Div%20with%20CSS%20Grid%20and%20place-items)

**Using position properties**

.parent {

  position: relative;

  width: 100%;

  height: 100vh;

}

.child {

  position: absolute;

  height: 100px;

  top: 50%;

  left: 50%;

  border: 1px solid black;

  transform: translate(-50%, -50%);

  width: 100px;

}

**React decrement timer**

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

function Timer() {

  const [val, setVal] = useState(10);

  const timer = useRef();

  const start = () => {

    timer.current = setInterval(() => {

      setVal((prevVal) => {

        if (prevVal > 0) {

          return prevVal - 1;

        }

        clearInterval(timer.current);

        return 0;

      });

    }, 1000);

  };

  const stop = () => {

    clearInterval(timer.current);

  };

  const reset = () => {

    setVal(10);

  };

  return (

    <>

      <p>{val? val: 'Time out'}</p>

      <button onClick={start}>Start</button>

      <button onClick={stop}>Stop</button>

      <button onClick={reset}>Reset</button>

    </>

  );

}

export default Timer;

**React increment timer**

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

function Timer(){

  const [val, setVal] = useState(0);

  const timer = useRef();

  const start = () => {

    timer.current = setInterval( () => {

      setVal(prevVal => prevVal + 1);

    }, 1000);

  }

  const stop = () => {

    clearInterval(timer.current);

  }

  const reset = () => {

    setVal(0);

  }

  return (

    <>

      <p>{val}</p>

      <button onClick={start}>Start</button>

      <button onClick={stop}>Stop</button>

      <button onClick={reset}>Reset</button>

    </>

  );

}

export default Timer;