//! React Interview Questions

// 1. what is React

// 2. what are the features and advantage and disadvantages of React

// 3. difference between Lirary and framework

// 4. what is single page application

// 5. what is declarative

// 6. what is virtual dom and how many virtual dom react maintain Explain

// 7. what is difference between virtual dom and real dom

// 8. what is recosilation

// 9. what is fiber

// 10. difference between reconsilation and fiber

// 11. Explain folder structure of React

// 12. what the use of scripts in package.json

// 13. what is component and what are component and types of components

// 14. difference between cbc and fbc

// 15. what are the life cycle method

// 16. what is Hook

// 17. what is createRoot and React-Dom

// 18. what is cdn link od react

// 19. what is render

// 20. what is useState and explain

// 21. what is props and children

// 22. difference between props and state

// 23. What is props drilling

// 24. how to avoid props drilling (what is HOC,context api,redux/reduce)

//! JavaScript interview concepts

// functions-> types

// i) arrow

// ii) HOF

// iii) Callback

// iv) Genarator

// This keyword working

//! 1] Library :

// Library is a collection of reusable block of Code.

// Libraries are light weight.

// Libraries are loosely typed.

// Libraries are always particular functional oriented.

// Using libraries we can create fast & scalable applications.

// The application which  are developed by using library are easy to debug.

// We can use multipe external libraries.

// Ex. React

// It focuses on front end development only.

//! 2] Framework :

// Framework is a collection of library.

// Framework are heavy weight.

// Framework are strictly typed.

// Framework focuses on multiple funtionality.

// If we need erternal packages or libraries then we can install them.

// The application which are developed by using framework are hard to debug.

// Ex. Next JS

// It focuses on front end as well as backend development.

//! Single Page Application:

// The application which does not require the page reload for updating the page content.

// Usually single page application contains only one HTML file

// Single page applications are faster than multi-page application

// AJAX is the hero of single page application.

//! AJAX ( Asychronous JavaScript And XML)

// AJAX will send only the necessary request to the server. So that bandwidth  will be very very less & beacause of that your request cycle will becames very very faster & ultimately your application becomes faster.

//It will bundel the all request info based on the extension.

// package bundels will help use to reduce the Bandwidth.

// If you sending as possible n number of request from the frontend that multiple requests are bundeled into as minimum number of requests as possible based on their extension by package bundler  so that Bandwidth will be related.

// package bundler webpack,rollup and parser

//! Bandwidth :

// It is number of request & response cycle per second

// As minimum is your bandwidth that much of fast your application.

//! Steps to create React JS Project :

// 1] Download Node js & add path to Environment variables.

// 2] Check in CMD if it is installed or not by using node -v

// 3] Create an empty folder and open CMD

// 4] use npm create vite - it will give latest stable version

//    npm create vite@latest - it will give us latest version

//    npm create vite@\*\*version\*\* - for specific version

// 5] Enter Y

// 6] Enter project name without any white space

// 7] Enter package name ( optional )

// 8] Select React and Enter

// 9] Select Plane Javascript & Enter

// 10] Use below commads one by one

// cd first Installation

// npm run dev

// 11] Search on brower 'http://localhost:5173/'

// ! 1) default Export :

// ! In default export we have to use default keyword.

// ! We can write default keyword only once.

// ! If we want to export multiple variables in default way, We can wrap the variables into an object L then export that object.

// ! When it is default export, while importing we 'can use any name.

// ! Mien it is default export, while importing we sould not take curly braces ({})

// ! When we are exporting as default, default keyword should not be on the same line of variable declaration.

// import planes from "./russia.js";

// log(planes);

//!  2] Named Export;

//! In Named export we shoud not use default keyword.

//! We can do named export multiple times.

//! When it is named export, while importing we should take same exported name.

//! When named export, while importing we must take curly braces ({})

//! When we are exporting as named export, export keyword can be on the same line of variable declaration

// import { plane1, plane2 } from './russia,js";

// log(plane1);

// log(plane2);

// log("hello"\* 3);

// ! 1 ) default Export :

// let planel = "Fighter 3et";

// Iet plane2 = "Cargo Plane";

// export default

// {planel,

// plane2, };

// ! 2) Named Export :

export let planel = "Fighter Plane";

export let plane2 = "Cargo Plane";

// ! 1) node\_mdules folder :

// It is the folder •that holds the all required packages that we have installed during react app installation.

// This packages •are required to run React application.

// Ex: @babal, @eslint, @rollup, @vitejs ,

// We don •t have to make any changes in this folder.

// ! 2) public folder :

// It is used to store static assests like image, audio, & video.

// It is similar to the asset folder.

// We will not use public folder to store the assetd instead of that we will prefer asset folder preseht inside src folder.

// ! 3] .gitignore :

// It is used when we are pushing the code to github at that time the files that needs to ignore or don't to push on github.

// We can make changes in this file.

// ! 4) eslint.config.js :

 // It acts as compiler check the syntax error in code and stops the execution of the application.

// To write strictly typed code in react.

// ! 5] index.htnl :

// -    With the react installation only one html file

// -    In html file we have 2 tag

// i.   <div id=”root”> </div>

// ii.  <script type=”module” src=”/src/main.jsx”><script>

// Wee Will have only one HTML file.

// We don't have .to make any changes in this file.

// ! 6) . package-lock.json :

// It contains all the information related to packages in JSON .format.

// We don't need to make any changes in this file.

// ! 7) package.json :

// It contains the information related to limited packages (most used) in JSON format.

// We don't need to make any changes in this file.

// ----Package.json

// Verson – 0.0.0    ->Major-update.minor-update.pach

//? Dev dependencies (Development dependencies) -

// are library and packages that are necessary during developing the software only and that are not required after deployment

// Ex.@eslint,@vite

//? Dependencies -

// These are major dependencies that are necessary during development also and production phase

// Ex.react,react-dom

// ! 8) vite.config.js :

// It contains all information related to vitejs.

// We don't have to make. any changes in this file.

//! README.md :

// We can write all the information or description related -to project.

//! diffing algorithm

// it is process of find difference between real dom and virtual dom

// we 2 virtual dom one is updated and another one for previous

//! Reconcilliatio procces

// is the process of implementing changes from virtual to real dom that are found during the diffing algorithm

//! React fiber

// like javascript engine in react has engine which called as react fiber

//! what is babel ?

// babel is a transpiler

//! Transpilation

// It will  convert the mordern javscript code (jsx code) into browser understandeble javascript

// mordern javascript code means .jsx code

// Transpilation done by babel

// Transpilation occur during rendering

//! What is jsx (javascript and xml)

// jsx is html look like javascript code

//! Main.jsx

// \* Strict Mode

// it will chack syntax strict type

// and it checks code 2 times

//! History

//\* Angular (2009)

// by- Google

// It is - Framework

// It follows- MVC architecture

// support - javascript

//\* React (2011)

// by - Facebook ( jordan walke)

// It is - Library

// 2012 - Messenger

// 2013 - Instagram

//

// It follow - component Based Architecture

// It support - javascript/ typescript

//\* Angular 2 (2014-2015)

// by - Google

// It is  - Framework

// It follows- component Based Architecture

// It support - typescript

//\* Vue (2015-2016)

// Evan you

// It is - Framework

// It follow - Component based architecture

// It support - javascript/typescript

//! npm

// npm stand for node package manager

// it is the world largest registery

// npm is a package manager and packaage manager installer

// there more package installer except npm

// Ex. yarn,bun,pnpm,spm,cnpm,chocolatey

//! Component based architecture working

// index.html -> main.jsx -> App.jsx -> Demo1.jsx,Demo2.jsx,Demo2.jsx

// in index.html contains div tag with root id

// in main.jsx have a function createRoot() which is from 'react-dom/client'

// createRoot() function take the div element whose id is root from index.html and by using render function it will rendering the App.jsx into index.html

// index.css is imported into main.jsx where App jsx is import inside App.jsx we import all the jsx components

// render() -> render method is used to display jsx code on the UI. during rendering multiple inbuild procees happpen like transpilation,diffing,reconsilation and evry thing

//! Jsx

// jsx is a javascript and xml

// jsx is a html look like structure

// while writing jsx we have to follow rule

//! Rules

// 1. compulsory to write jsx

// 2. more than one element should wrap into single parent element. at a time only one

// 3. while writing jsx each and every jsx tag must be closed (without slash ' / ')

// 4. in jsx we can't use attribute like "class" and "for" while wrtting the jsx instead of this we use "className" and "htmlFor"

// 5. React Fragment - fragment are nothing but empty angular braces (<></>) react fragment are use to avoid the extra node

// 6. Jsx Expression

// 7. return statement except statement on same line and at a time only one can return

// 8. in jsx components are return in paired and unpaired tags also importing them

//! Jsx Expression Rules

// 1.we should not write console statements inside jsx XPathExpression beacause it will not displayed

// 2.we cannot write conditional statement inside jsx expression except ternary operator and short circuit operator

// 3.in jsx expression we cannot write the loop statement we can use only hof except those who can't return

//! Component

// Component is resuable block of code

// Component always return jsx

// Compenent is contain jsx and javascript

//! Component

// Component is resuable block of code

// Component always return jsx

// Compenent is contain jsx and javascript

//! types of components

//1. function based componenet

//2. class based component

//! 2. Class Based Component

// class based component are the component that are created using classes

// cbc is the one of the traditional way create component in react

//it is important to understand class based component and function based component

// by using class syntax we create class based component

// it extends Component

// it has reder method

// it have constructor

// it is statefull component

//! state management in class based component

// constructor()

// {

//   ? we declare state inside the constructor

//     super()

//     this.state={

//         count : 0,

//     }

// }

// render(){

//     return(

//         <div>

//         <h1>

//             Class Based component example</h1>

//             <h1>Count : {this.state.count}</h1>

//             <button

//                 onClick={()=>{

//                     this.setState({

//                          count : this.state.count+1

//                         })

//                         console.log(this.state)

//                 }}

//             >increment</button>

//         </div>

//     )

// }

//!render()

// render method is used to display jsx code on the UI.

// during rendering multiple inbuild procees happpen like transpilation,diffing,reconsilation and evry thing

// different between class based and function based

// 1. we use classes to creating component

// 2. cbc extends Component

// 3. cbc has constructor in that we have to write super call stmt

// 4. In cbc we must use render method inside that we will have return statement

// 5. cbc is a statefull component means that it has inbuild state and to manage we have inbuild setState() method

// 6. in cannot use hooks inside class based  component

// 7. we have inbuild life cycle method in class based component //? i.e. constructor(),render() ,getDerivedStateFromProps(),componentDidMount()

// 1. we use function to creating component

// 2. fbc does not extends the Component

// 3. fbc does not have constructor and super calling stmt

// 4. In fbc does not require the render method it will return the jsx to parent element or caller

// 5. fbc are stateless component means that it does not have it own state and setState() method if we want to have state in fbc we have to use some extra HOOKS i.e. useState() Hook

// 6. we can use Hooks inside function based component

// 7. function based component does not have any inbuild methods for managing life cycle so to achieve life cycle method in fbc we have to use Hook called as useEffect() hook

//! what is Hook

// Hooks are nothing but the inbuild methods or functions.

// Hooks name starts with use

// i.e. useState(),useEffect(),useContext(),useDispatch(),useReducer(),useCallBack(),useMemo()

// we can create any function and will name it usrXYZ. such functions can be treated as -- Custom Hooks --

//? i.e. let useAddtion =(a,b)=>{ return a+b}

//? let sum=useAddition(10,20)

//? log(sum)

//! useState() Hook

// in useState the whole component will refresh

//? let [state,setState]=useState(0);

//?    return(

//?         <div>

//?           <h1>Use State Example 1</h1>

//?           <h2>count = {state} </h2>

//?             <button onClick={()=>{ setState(state+1)}}> increment</button>

//?         </div>

//?     );

//

//! Hot Module replacement

// HMR is one the most useful features offered by package bundler. it allow all kinds of modules to be upadated at runtime without the need for full refresh

// in html if we make changes in single element it affect whole page

// in react if make changes in single element it will affect desire element not whole page or component

// HMT is intended for use inproduction , meaning it should only be used in development

//! Feature or charactersticks of react

// 1. It follows component based architecture with diagram

// 2. React is declaritive in nature (we don't need write dom method becoz everything already declacred with jsx . we called react as declarative becoz react provide many inbuild hooksfor different functionality we just have to use them)

//   i.declarative we get all inbuild things/ everything is declared we just have to use it

//   ii. imparative means we have to write all these things /in imprative many of the things are not already declared we have to explicitly or manualy explain

// 3. write once use anywhere . you created component for such functionality that are repeatative and we can use such component n numbers of time. i.e. we will create the button component one and we will use it where we want any numbers of times

//! Props

// props is properties which is use to send data from one component to another component in the form of object

// Props are imutable

// Props are by default present both in class based component and function based component

// It is a unidirectional

//? Props.children

// when we passing props through component pair tag we will get children

//? default prop

// whenever parent don't send data and child is expecting a data that time child will create default prop

// whenever creating default prop we need to destructring

//! difference props and state

// props are imutable

// props are not in component level

// props are use in both cbc and fbc

// states are mutable

// state are component level ( state are only access only in component)

// state are in only cbc

//! Props drilling

// sending data to multiple nested component (nth child components)

// it is not a good practise

// To avoid this we have

// HOC higher order component

// context api

// Redux / Reduce toolkit

//! Types of css

// 1.Inline css by using style={{}} object

// 2.module css