Q 1. Explain Python ?

Ans. Python is an interpreted, object-oriented, high-level programming language.Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse.

Q2. What is Pythonpath ?

Ans. A Pythonpath tells the Python interpreter to locate the module files that can be imported into the program. It includes the Python source library directory and source code directory.

Q3. Can we preset Pythonpath?

Ans. Yes, we can preset Pythonpath as a Python installer.

Q4. **What are the supported standard data types in Python?**

Ans. The supported standard data types in Python include the following.

1. List.
2. Number.
3. String.
4. Dictionary.
5. Tuples.

Q5. Define tuple in Python ?

Ans. Tuples is a sequence data type in Python. The number of values in tuples are separated by commas. Tuples are immutable.

Q.6 **What are the positive and negative indices?**

Ans. In the positive indices are applied the search beings from left to the right. In the case of the negative indices, the search begins from right to left. For example, in the array list of size n the positive index, the first index is 0, then comes 1 and until the last index is n-1. However, in the negative index, the first index is -n, then -(n-1) until the last index will be -1.

Q.7 **Define Pass statement in Python?**

Ans. A Pass statement in Python is used when we cannot decide what to do in our code, but we must type something for making syntactically correct.

Q.8 **Why do we need a break in Python?**

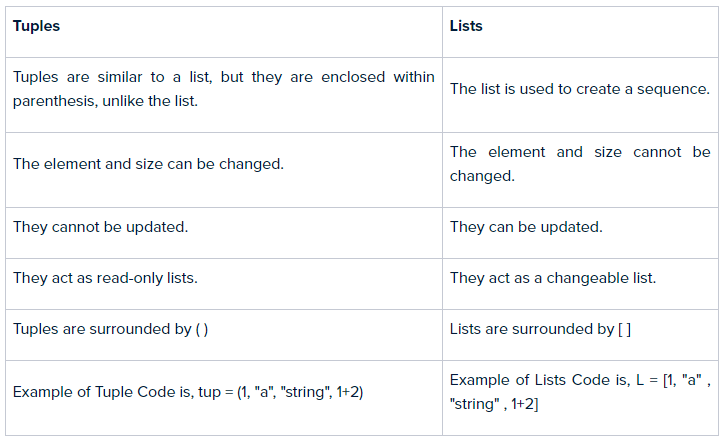
Ans. Break helps in controlling the Python loop by breaking the current loop from execution and transfer the control to the next block.

Q.9 **Why do we need a continue in Python?**

Ans. A continue also helps in controlling the Python loop but by making jumps to the next iteration of the loop without exhausting it.

Q10. **What is the major difference between tuples and lists in Python?**

Ans.



Q11. **What is the purpose of relational operators in Python?**

Ans. The purpose of relational operators in Python is to compare values.

**Q12**. **What are assignment operators in Python?**

Ans. The assignment operators in Python can help in combining all the arithmetic operators with the assignment symbol.

**Q13.** **Explain Inheritance and its various types in Python?**

**Ans.** Inheritance enables a class to acquire all the members of another class. These members can be attributes, methods, or both. By providing reusability, inheritance makes it easier to create as well as maintain an application. The class which acquires is known as the child class or the derived class. The one that it acquires from is known as the superclass or base class or the parent class.

Q14. **Python has something called the dictionary. Explain using an example.**

Ans. A dictionary in Python programming language is an unordered collection of data values such as a map. Dictionary holds key:value pair. It helps in defining a one-to-one relationship between keys and values. Indexed by keys, a typical dictionary contains a pair of keys and corresponding values.

Let us take an example with three keys, namely Website, Language, and Offering. Their corresponding values are hackr.io, Python, and Tutorials. The code for the example will be:

**dict={‘Website’:‘hackr.io’,‘Language’:‘Python’:‘Offering’:‘Tutorials’}**

**print dict[Website] #Prints hackr.io**

**print dict[Language] #Prints Python**

**print dict[Offering] #Prints Tutorials**

#### Q15. Python supports negative indexes. What are they and why are they used?

Ans. The sequences in Python are indexed. It consists of positive and negative numbers. Positive numbers use 0 as the first index, 1 as the second index, and so on. Hence, any index for a positive number n is n-1.

Unlike positive numbers, index numbering for the negative numbers start from -1 and it represents the last index in the sequence. Likewise, -2 represents the penultimate index. These are known as negative indexes. Negative indexes are used for:

* Removing any new-line spaces from the string, thus allowing the string to except the last character, represented as S[:-1]
* Showing the index to representing the string in the correct order

**Q16.** **What is IIFEs (Immediately Invoked Function Expressions)?**

**Ans.** It’s an Immediately-Invoked Function Expression, or IIFE for short. It executes immediately after it’s created:

(function IIFE(){

console.log( "Hello!" );

})();

// "Hello!"

## **Q17. What's a typical use case for anonymous functions?**

**Ans.** They can be used in IIFEs to encapsulate some code within a local scope so that variables declared in it do not leak to the global scope.

(function() {

// Some code here.

})();

As a callback that is used once and does not need to be used anywhere else. The code will seem more self-contained and readable when handlers are defined right inside the code calling them, rather than having to search elsewhere to find the function body.

setTimeout(function() {

console.log('Hello world!');

}, 1000);

Arguments to functional programming constructs or Lodash (similar to callbacks).

const arr = [1, 2, 3];

const double = arr.map(function(el) {

return el \* 2;

});

console.log(double); // [2, 4, 6]

**Q18. What Is react ?**

**Ans** React is a front-end and open-source JavaScript library which is useful in developing user interfaces specifically for applications with a single page. It is helpful in building complex and reusable user interface(UI) components of mobile and web applications as it follows the component-based approach.

The important features of React are:

* It supports server-side rendering.
* It will make use of the virtual DOM rather than real DOM (Data Object Model) as RealDOM manipulations are expensive.
* It follows unidirectional data binding or data flow.
* It uses reusable or composable UI components for developing the view.

**Q19. What are the advantages of using react ?**

**Ans.**

* **Use of Virtual DOM to improve efficiency:**React uses virtual DOM to render the view. As the name suggests, virtual DOM is a virtual representation of the real DOM. Each time the data changes in a react app, a new virtual DOM gets created. Creating a virtual DOM is much faster than rendering the UI inside the browser. Therefore, with the use of virtual DOM, the efficiency of the app improves.
* **Gentle learning curve:** React has a gentle learning curve when compared to frameworks like Angular. Anyone with little knowledge of javascript can start building web applications using React.
* **SEO friendly:** React allows developers to develop engaging user interfaces that can be easily navigated in various search engines. It also allows server-side rendering, which boosts the SEO of an app.
* **Reusable components:**React uses component-based architecture for developing applications. Components are independent and reusable bits of code. These components can be shared across various applications having similar functionality. The re-use of components increases the pace of development.
* **Huge ecosystem of libraries to choose from:**React provides you with the freedom to choose the tools, libraries, and architecture for developing an application based on your requirement.

**Q20. What is useState() in React ?**

**Ans.** The useState() is a built-in React Hook that allows you for having state variables in functional components. It should be used when the DOM has something that is dynamically manipulating/controlling.

In the below-given example code, The useState(0) will return a tuple where the count is the first parameter that represents the counter’s current state and the second parameter setCounter method will allow us to update the state of the counter.

...

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

const [otherStuffs, setOtherStuffs] = useState(...);

...

const setCount = () => {

setCounter(count + 1);

setOtherStuffs(...);

...

};

We can make use of setCounter() method for updating the state of count anywhere. In this example, we are using setCounter() inside the setCount function where various other things can also be done. The idea with the usage of hooks is that we will be able to keep our code more functional and avoid class-based components if they are not required.

**Q21. What are keys in react ?**

**Ans.** A key is a special string attribute that needs to be included when using lists of elements.

Example of a list using key -

const ids = [1,2,3,4,5];

const listElements = ids.map((id)=>{

(

<li key={id.toString()}>

{id}

</li>

)

})

**Importance of keys -**

* Keys help react identify which elements were added, changed or removed.
* Keys should be given to array elements for providing a unique identity for each element.
* Without keys, React does not understand the order or uniqueness of each element.
* With keys, React has an idea of which particular element was deleted, edited, and added.
* Keys are generally used for displaying a list of data coming from an API.

**Q22. What is JSX ?**

**Ans.** JSX stands for JavaScript XML. It allows us to write HTML inside JavaScript and place them in the DOM without using functions like appendChild( ) or createElement( ).

As stated in the official docs of React, JSX provides syntactic sugar for React.createElement( ) function.

Note- We can create react applications without using JSX as well.

Let’s understand **how JSX works**:

Without using JSX, we would have to create an element by the following process:

const text = React.createElement('p', {}, 'This is a text');

const container = React.createElement('div','{}',text );

ReactDOM.render(container,rootElement);

**Using JSX**, the above code can be simplified:

const container = (

<div>

<p>This is a text</p>

</div>

);

ReactDOM.render(container,rootElement);

As one can see in the code above, we are directly using HTML inside JavaScript.

**Q23. Explain React State and Props ?**

**Ans.**

| **Props** | **State** |
| --- | --- |
| Immutable | Owned by its component |
| Has better performance | Locally scoped |
| Can be passed to child components | Writeable/Mutable |
|  | has setState() method to modify properties |
|  | Changes to state can be asynchronous |
|  | can only be passed as props |

* **React State**  
  Every component in react has a built-in state object, which contains all the property values that belong to that component.  
  In other words, the state object controls the behaviour of a component. Any change in the property values of the state object leads to the re-rendering of the component.

Note- State object is not available in functional components but, we can use React Hooks to add state to a functional component.

* **React Props**

Every React component accepts a single object argument called props (which stands for “properties”).  These props can be passed to a component using HTML attributes and the component accepts these props as an argument.

Using props, we can pass data from one component to another.

*Passing props to a component:*

While rendering a component, we can pass the props as an HTML attribute:

<Car brand="Mercedes"/>

The component receives the props:

*In Class component:*

class Car extends React.Component {

constructor(props) {

super(props);

this.state = {

brand: this.props.brand,

color: "Black"

};

}

}

*In Functional component:*

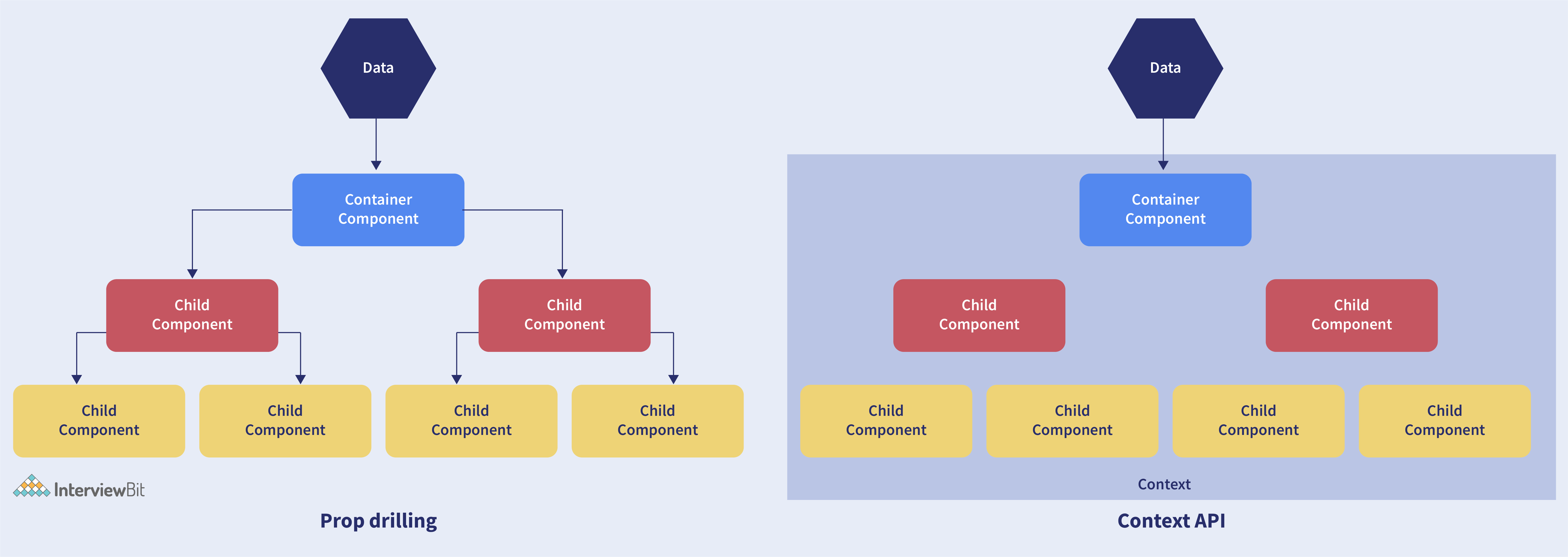
function Car(props) {

let [brand, setBrand] = useState(props.brand);

}

Note- Props are read-only. They cannot be manipulated or changed inside a component.

**Q.24 What is prop drilling in react ?**

**Ans.** 

Sometimes while developing React applications, there is a need to pass data from a component that is higher in the hierarchy to a component that is deeply nested. To pass data between such components, we pass props from a source component and keep passing the prop to the next component in the hierarchy till we reach the deeply nested component.

**Q25.**  What are the rules that must be followed while using React Hooks?

**Ans.** There are 2 rules which must be followed while you code with Hooks:

* React Hooks must be called only at the top level. It is not allowed to call them inside the nested functions, loops, or conditions.
* It is allowed to call the Hooks only from the React Function Components.

**Q26. What is the use of useEffect react hooks ?**

**Ans.** The useEffect React Hook is used for performing the side effects in functional components. With the help of useEffect, you will inform React that your component requires something to be done after rendering the component or after a state change. The function you have passed(can be referred to as “effect”) will be remembered by React and call afterwards the performance of DOM updates is over. Using this, we can perform various calculations such as data fetching, setting up document title, manipulating DOM directly, etc, that don’t target the output value. The useEffect hook will run by default after the first render and also after each update of the component. React will guarantee that the DOM will be updated by the time when the effect has run by it.

The useEffect React Hook will accept 2 arguments: useEffect(callback[, dependencies]);

Where the first argument callback represents the function having the logic of side-effect and it will be immediately executed after changes were being pushed to DOM. The second argument dependencies represent an optional array of dependencies. The useEffect() will execute the callback only if there is a change in dependencies in between renderings.

**Example:**

import { useEffect } from 'react';

function WelcomeGreetings({ name }) {

const msg = `Hi, ${name}!`; // Calculates output

useEffect(() => {

document.title = `Welcome to you ${name}`; // Side-effect!

}, [name]);

return <div>{msg}</div>; // Calculates output

}

The above code will update the document title which is considered to be a side-effect as it will not calculate the component output directly. That is why updating of document title has been placed in a callback and provided to useEffect().

Consider you don’t want to execute document title update each time on rendering of WelcomeGreetings component and you want it to be executed only when the name prop changes then you need to supply name as a dependency to useEffect(callback, [name]).

**Q27. What are the different ways to style a React component?**

* **Ans. Inline Styling:**We can directly style an element using inline style attributes. Make sure the value of style is a JavaScript object:

class RandomComponent extends React.Component {

render() {

return (

<div>

<h3 style={{ color: "Yellow" }}>This is a heading</h3>

<p style={{ fontSize: "32px" }}>This is a paragraph</p>

</div>

);

}

}

* **Using JavaScript object:**We can create a separate JavaScript object and set the desired style properties. This object can be used as the value of the inline style attribute.

class RandomComponent extends React.Component {

paragraphStyles = {

color: "Red",

fontSize: "32px"

};

headingStyles = {

color: "blue",

fontSize: "48px"

};

render() {

return (

<div>

<h3 style={this.headingStyles}>This is a heading</h3>

<p style={this.paragraphStyles}>This is a paragraph</p>

</div>

);

}

}

* **CSS Stylesheet:**We can create a separate CSS file and write all the styles for the component inside that file. This file needs to be imported inside the component file.

import './RandomComponent.css';

class RandomComponent extends React.Component {

render() {

return (

<div>

<h3 className="heading">This is a heading</h3>

<p className="paragraph">This is a paragraph</p>

</div>

);

}

}

* **CSS Modules:** We can create a separate CSS module and import this module inside our component. Create a file with “.module.css”‘ extension, styles.module.css:

.paragraph{

color:"red";

border:1px solid black;

}

We can import this file inside the component and use it:

import styles from './styles.module.css';

class RandomComponent extends React.Component {

render() {

return (

<div>

<h3 className="heading">This is a heading</h3>

<p className={styles.paragraph} >This is a paragraph</p>

</div>

);

}

}

**Q28. Differentiate React Hooks and classes.**

**Ans.**

| **React Hooks** | **Classes** |
| --- | --- |
| It is used in functional components of React. | It is used in class-based components of React. |
| It will not require a declaration of any kind of constructor. | It is necessary to declare the constructor inside the class component. |
| It does not require the use of this keyword in state declaration or modification. | Keyword this will be used in state declaration (this.state) and in modification (this.setState()). |
| It is easier to use because of the useState functionality. | No specific function is available for helping us to access the state and its corresponding setState variable. |
| React Hooks can be helpful in implementing Redux and context API. | Because of the long setup of state declarations, class states are generally not preferred. |

**Q29. What is react Router ?**

**Ans.** React Router refers to the standard library used for routing in React. It permits us for building a single-page web application in React with navigation without even refreshing the page when the user navigates. It also allows to change the browser URL and will keep the user interface in sync with the URL. React Router will make use of the component structure for calling the components, using which appropriate information can be shown. Since React is a component-based framework, it’s not necessary to include and use this package. Any other compatible routing library would also work with React.

The major components of React Router are given below:

* **BrowserRouter:** It is a router implementation that will make use of the HTML5 history API (pushState, popstate, and event replaceState) for keeping your UI to be in sync with the URL. It is the parent component useful in storing all other components.
* **Routes:**It is a newer component that has been introduced in the React v6 and an upgrade of the component.
* **Route:**It is considered to be a conditionally shown component and some UI will be rendered by this whenever there is a match between its path and the current URL.
* **Link:** It is useful in creating links to various routes and implementing navigation all over the application. It works similarly to the anchor tag in HTML.

**Q30. Explain Conditional rendering in react?**

**Ans.** Conditional rendering refers to the dynamic output of user interface markups based on a condition state. It works in the same way as JavaScript conditions. Using conditional rendering, it is possible to toggle specific application functions, API data rendering, hide or show elements, decide permission levels, authentication handling, and so on.

There are different approaches for implementing conditional rendering in React. Some of them are:

* Using if-else conditional logic which is suitable for smaller as well as for medium-sized applications
* Using ternary operators, which takes away some amount of complication from if-else statements
* Using element variables, which will enable us to write cleaner code.