📓 Set 2: React Functions & Components

Using `function` keyword:

```
function greet(name) {
  return "Hello " + name;
}
```

Using arrow function:

```
const greet = (name) ⇒ {
  return "Hello " + name;
};
```

Short version (one-liner):

```
const greet = name ⇒ "Hello " + name;
```

✅ MCQs – Functions

Q1. Which is the correct syntax for an arrow function? a) `function ⇒ (param) {}` b) `const fn = (param) ⇒ {}` c) `⇒ function(param) {}` d) `const fn = function ⇒ {}`

Q2. How do you call a function named `sayHi`? a) `sayHi[]` b) `sayHi();` c) `call sayHi()` d) `sayHi{}`

Q3. What is the main benefit of arrow functions? a) Global access b) Shorter syntax c) Slower code d) Bigger size

Q4. Which of these correctly defines a function using the `function` keyword? a) `let fn = ⇒ {}` b) `function fn() {}` c) `fn() = function {}` d) `create function()`

Q5. What is the return type of a function that has no `return` statement? a) `undefined` b) `null` c) `false` d) `0`

✅ 2. Return Statement – Basics

🔷 Explanation:

- `return` is used to output a value from a function.
- If you omit `return`, the function returns `undefined`.

```
function add(a, b) {
  return a + b;
}
```

✅ MCQs – Return Statement

Q1. What does `return` do in a function? a) Restarts loop b) Stops program c) Returns value and exits function d) Imports another file

Q2. What is returned here?

```
function say() {
  return "Hi";
}
```

a) Nothing b) undefined c) Hi d) Error

Q3. Can arrow functions return without using the `return` keyword? a) No b) Yes, in one-line expressions c) Only in `var` d) Only in React

Q4. What is the result?

```
const x = () ⇒ 5;
console.log(x());
```

a) `x` b) `undefined` c) `5` d) `NaN`

Q5. What happens after `return` in a function? a) Next code runs b) Function continues c) Function exits d) Function stops, value is lost

---

✅ 3. Modules – `export` and `export default`

🔷 Explanation:

You can break JS code into modules (files) and use `export` / `import`.

Named export:

```
export function greet() { ... }
```

Default export:

```
export default function main() { ... }
```

Importing:

```
import { greet } from './file.js';
import main from './file.js';
```

✅ MCQs – Modules & Exports

Q1. What does `export` do? a) Delete variables b) Show output c) Make a function usable in other files d) Make function global

Q2. What is the correct way to import a named export? a) `import function greet from './file'` b) `import { greet } from './file'` c) `import greet = './file'` d) `export greet from './file'`

Q3. Can you export multiple functions using `export` keyword? a) No b) Yes c) Only in Node d) Only one at a time

Q4. What does `export default` mean? a) Export only to frontend b) Marks as primary export c) Exports in uppercase d) Prevents import

Q5. Which one is correct default import? a) `import { myFunc } from './mod.js'` b) `import * from './mod.js'` c) `import myFunc from './mod.js'` d) `use myFunc './mod.js'`

✅ 4. Components (Intro – React based)

🔷 Explanation:

In React, components are functions that return JSX.

```
function Welcome() {
  return <h1>Hello!</h1>;
}
```

Or using arrow function:

```
const Welcome = () => <h1>Hello!</h1>;
```

✅ MCQs – Components (for React)

Q1. What is a React component? a) A loop b) A reusable UI block c) A variable d) A backend API

Q2. How do you define a functional component? a) `component Welcome() {}` b) `class Welcome()` c) `function Welcome() { return JSX }` d) `use Welcome()`

Q3. What should a component always return? a) Array b) String c) JSX d) JSON

Q4. Components in React should start with...? a) lowercase b) uppercase c) underscore d) number

Q5. Can you use arrow functions to create components? a) No b) Yes c) Only in ES5 d) Only in backend