6 Types Of Arrow Functions

1. No Parameters

If the functions takes no parameters, you use empty parentheses.

```
1   const greet = () => "Hello!";
2   console.log(greet());
3   // Outputs: Hello!
```

2. Single Parameter

If there are one parameter, parantheses are optional.

```
1   const square = x => x * x;
2   console.log(square(4));
3   // Outputs: 16
```

3. Multiple Parameters

Functions can accept more than one parameter. The parameters are separated by commas, both in the function definition and in the function call.

```
const add = (a, b) => a + b;
console.log(add(2, 3));
// Outputs: 5
```

4. Function body with Multiple Statements

If the Function body has more then one statement you need to use curley braces and specify the return keyword (if you want return something)

```
const greetPerson = name => {
  const greeting = "Hello, " + name + "!";
  return greeting;
}
console.log(greetPerson("Alice"));
// Outputs: Hello, Alice!
```

5. Returning Object Literals

When directly returning an object literal, wrap the literal in parantheses to differentiate it from the function block.

```
1  const makePerson = (firstName, lastName) =>
   ({ first: firstName, last: lastName });
2  console.log(makePerson("John", "Doe"));
3  // Outputs: { first: 'John', last: 'Doe' }
```

6. High order function and Callbacks

Arrow functions are particularly popular when used as short callbacks.

```
1  const numbers = [1, 2, 3, 4];
2  const doubled = numbers.map(num => num * 2);
3  console.log(doubled);
4  // Outputs: [2, 4, 6, 8]
```

FOLLOW