

## Quiz on TypeScript Functions

### 1. What is the correct syntax for defining a named function in TypeScript?

- A) `let sum = (a: number, b: number) => { return a + b; };`
- B) `function sum(a: number, b: number): number { return a + b; }`
- C) `const sum = function(a, b) { return a + b; };`
- D) `sum(a: number, b: number) { return a + b; }`

### 2. What is an anonymous function in TypeScript?

- A) A function with no return type
- B) A function that does not have a name
- C) A function that cannot take parameters
- D) A function that only works inside a class

### 3. Which of the following is an example of an arrow function?

- A) `let greet = function(name: string) { return "Hello " + name; };`
- B) `function greet(name: string): string { return "Hello " + name; }`
- C) `let greet = (name: string) => "Hello " + name;`
- D) `const greet(name: string) => { return "Hello " + name; };`

### 4. What is the output of the following TypeScript code?

```
let add = (a: number, b: number) => a + b;  
console.log(add(5, 10));
```

- A) 5
- B) 10
- C) 15
- D) undefined

### 5. How do you specify the return type of a function in TypeScript?

- A) `function multiply(a: number, b: number) { return a * b; }`
- B) `function multiply(a: number, b: number): number { return a * b; }`
- C) `function multiply(a, b): number { return a * b; }`
- D) `function multiply(a: number, b: number) -> number { return a * b; }`

**6. What is the purpose of the `=>` symbol in TypeScript functions?**

- A) It defines a named function
- B) It defines an anonymous function
- C) It is used to define an arrow function
- D) It is used to return a value from a function

**7. Which of the following is a correct example of an anonymous function in TypeScript?**

- A) `let sum = function(a: number, b: number): number { return a + b; };`
- B) `function sum(a: number, b: number): number { return a + b; }`
- C) `let sum = (a: number, b: number) => a + b;`
- D) `const sum: function(a: number, b: number) { return a + b; };`

**8. Can an arrow function be assigned to a variable?**

- A) Yes
- B) No

**9. What is the return type of the following function?**

```
let greet = (name: string): string => `Hello, ${name}!`;
```

- A) number
- B) string
- C) void
- D) boolean

**10. Which of the following statements about named functions is true?**

- A) Named functions must always have a return type
- B) Named functions cannot be used before they are declared
- C) Named functions can be called using their function name
- D) Named functions can only return numbers

**11. What will happen if a function does not specify a return type?**

- A) TypeScript will automatically infer the return type
- B) The function will not compile
- C) The function will always return undefined
- D) TypeScript will throw an error

**12. How do you write a multi-line arrow function in TypeScript?**

A)

```
let multiply = (a: number, b: number) => return a * b;
```

B)

```
let multiply = (a: number, b: number) => { return a * b; };
```

C)

```
let multiply = (a: number, b: number): { return a * b; };
```

D)

```
let multiply = (a: number, b: number) => { a * b; };
```

**13. What is the correct syntax for a function with no parameters?**

- A) let greet = () => { return "Hello!"; };
- B) function greet { return "Hello!"; }
- C) let greet = (void) => { return "Hello!"; };
- D) function greet(): void { return "Hello!"; }

**14. What will be the output of the following code?**

```
let divide = (a: number, b: number): number => a / b;  
console.log(divide(10, 2));
```

- A) 5
- B) 10
- C) 2
- D) 0.5

**15. What is a key advantage of using arrow functions in TypeScript?**

- A) They are always faster than named functions
- B) They do not need a return type
- C) They provide a shorter syntax and automatically bind this
- D) They cannot be assigned to a variable

**Answers**

- 1. **B**
- 2. **B**
- 3. **C**
- 4. **C**
- 5. **B**
- 6. **C**
- 7. **A**
- 8. **A**
- 9. **B**
- 10. **C**
- 11. **A**
- 12. **B**
- 13. **A**
- 14. **A**
- 15. **C**