What is the difference between function declaration and Function expression.

Function Declaration:

- Can be called before it appears in the code because of hoisting.
- Syntax: function functionName() { ... }.

Function Expression:

- Only available after its definition, not hoisted.
- Syntax: const functionName = function() { ... };.

Explain Hoisting in terms of function.

There are two types of function definitions in JavaScript that behave differently when it comes to hoisting: **function declarations** and **function expressions**.

Function Declarations: Both the function's name and body are hoisted to the top, making the function callable even before it's defined.

Function Expressions: Only the variable declaration is hoisted (if it's assigned to a variable), but the function itself is not available until runtime when the expression is executed. Thus, it cannot be invoked before it's initialized.

What is arrow function?

- **Concise syntax**: Shorter way to write function expressions.
- **Lexical this**: Inherits this from the surrounding context, which makes it useful in callbacks and event handlers.
- **No arguments object**: Use rest parameters if you need to handle function arguments.
- **Cannot be used as constructors**: You cannot instantiate an arrow function with new.
- **No prototype**: Arrow functions do not have a prototype property.

Arrow functions are great for scenarios where you need a simple, concise function, and you don't need the dynamic this context or arguments object typically found in traditional functions.