ANSWER 10-

To define an asynchronous function in JavaScript using async/await, you need to use the async keyword before the function declaration. The async keyword tells JavaScript that the function is asynchronous and will always return a Promise. Here's an example of defining an asynchronous function using async/await:

```
async function myAsyncFunction() {
  // Asynchronous operations with await
  const result = await someAsyncOperation();
  return result;
}
```

In the example above, the myAsyncFunction is declared as an asynchronous function using the async keyword. Inside the function, you can perform asynchronous operations by using the await keyword before calling other functions that return Promises. The await keyword pauses the execution of the function until the Promise is resolved or rejected.

Note that when using await, it should always be inside an async function. The await keyword can only be used within an async function and allows you to pause the function execution until the Promise is settled.

When you call an asynchronous function defined with async, it returns a Promise. You can use .then() and .catch() or async/await to handle the result of the Promise.

Example using async/await to consume the asynchronous function:

```
myAsyncFunction()
  .then(result => {
    // Handle the result
  })
```

```
  .catch(error => {
    // Handle the error
  });
```

With async/await, you can write asynchronous code in a more synchronous-like style, making it easier to read and maintain. It allows you to write code that appears to be sequential and avoids the need for nested callbacks or complex Promise chaining.