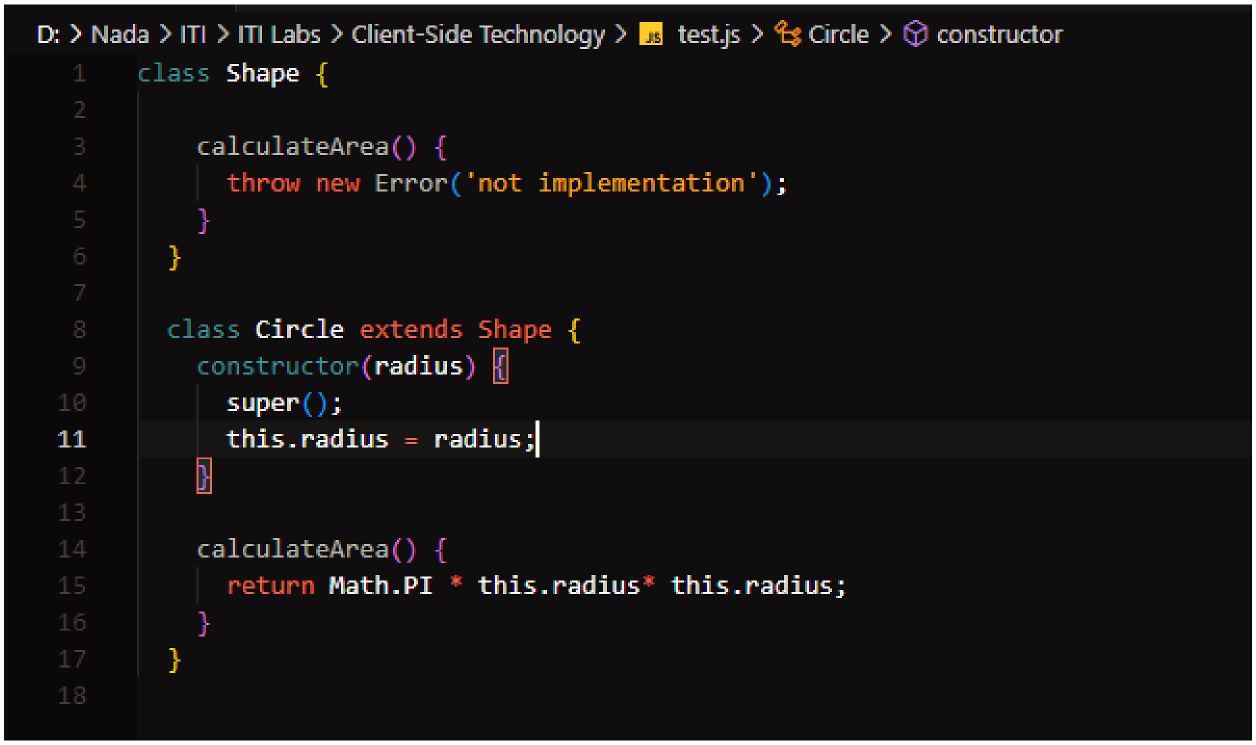
Table of Contents

Abstract in JavaScript:

- In JavaScript, abstraction is achieved through the use of abstract classes. An abstract class is a blueprint for other classes and cannot be instantiated itself.

- Abstract classes can have abstract methods, which are declared without providing an implementation. Subclasses must implement these abstract methods.



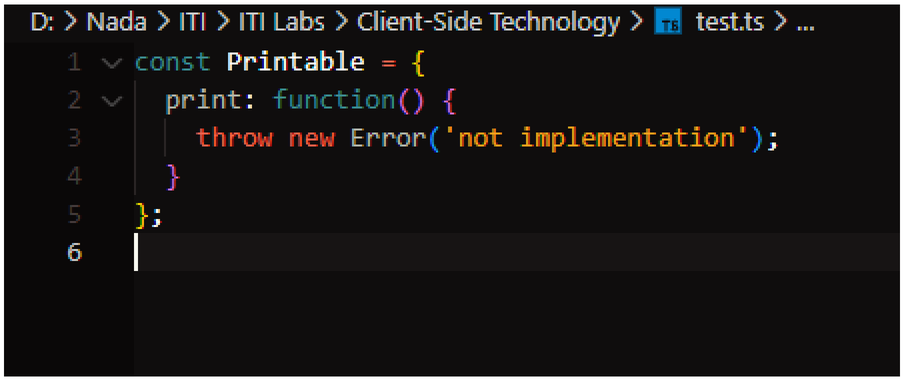
D: > Nada > ITI > ITI Labs > Client-Side Technology > test.js > Circle > constructor

```
class Shape {

    calculateArea() {
      throw new Error('not implementation');
    }
  }

  class Circle extends Shape {
    constructor(radius) {
      super();
      this.radius = radius;
    }

    calculateArea() {
      return Math.PI * this.radius* this.radius;
    }
  }

```

Interface in JavaScript:

- JavaScript does not have native support for interfaces, but developers can emulate them using a combination of objects and conventions.

- An interface defines a contract for classes, specifying the methods they must implement. If a class adheres to an interface, it guarantees the presence of certain methods.



D: > Nada > ITI > ITI Labs > Client-Side Technology > test.ts > ...

```
const Printable = {
  print: function() {
    throw new Error('not implementation');
  }
};
```

Differences between Abstract and Interface:

- Instantiation: Abstract classes cannot be instantiated directly(its child), while interfaces do not have instances.
- Implementation: Abstract classes can have both implemented and abstract methods, whereas interfaces only declare method signatures.
- Inheritance: A class can extend only one abstract class, but it can implement multiple interfaces.

References:

MDN web Docs