**Day 3 Lab Assignments**

**Do the following assignments using TypeScript, and make an html page to test all of them (Use tsc command and watch to transpiler from TypeScript to JS):**

1. Using TypeScript to solve the following:
2. Implement Rectangle custom type:
   1. Constructor that width and height.
   2. CalcCircumference() function that returns the circumference of the rectangle.
   3. WhoAmI() Static function that prints text (“I am rectangle”).
3. Implement Square custom type, that inherits from rectangle, and change the following:
   1. Make the constructor of square takes one parameter only.
   2. In the square constructor, make width and height of the parent equal.
      1. You’ll need to call the super() constructor in the square constructor to access the parent object properties.
   3. Override circumference () to return circumference of the Square.
   4. Override WhoAmI() function to print text (“I am square”).
4. Create objects from rectangle and square and test them, and make sure the square inherited rectangle members.
5. Try the following Typescript features:
6. Types annotation, union types
7. Function with typed arguments and return type.
8. Interfaces & classes & inheritance
9. Generics
10. Enums
11. Namespaces
12. Modules
13. Implement one at least of the design patterns in JavaScript (or TypeScript).
14. Complete Day.2 Async call assignments and any other missing assignments.

**Bonus:**

1. Implement Queue or Stack using typescript and OOP.
2. Implement linked list using typescript and OOP [create class node, and class linkedList].
3. Implement the following design patterns using Javascript (You can follow the provided links):
   1. Singleton design pattern
      1. <https://scotch.io/bar-talk/4-javascript-design-patterns-you-should-know#singleton>
   2. Factory design pattern
      1. <https://www.joezimjs.com/javascript/javascript-design-patterns-factory>
      2. <https://www.joezimjs.com/javascript/javascript-design-patterns-factory-part-2>
   3. Prototype design pattern
      1. <https://scotch.io/bar-talk/4-javascript-design-patterns-you-should-know#prototype-design-pattern>
   4. Dependency Injection design pattern
      1. <https://www.joezimjs.com/javascript/javascript-design-patterns-observer>
   5. Decorator design pattern
      1. <https://www.joezimjs.com/javascript/javascript-design-patterns-decorator>
   6. Observer design pattern
      1. <https://scotch.io/bar-talk/4-javascript-design-patterns-you-should-know#observer-design-pattern>
   7. Module design pattern
      1. <https://scotch.io/bar-talk/4-javascript-design-patterns-you-should-know#module-design-pattern>

**<Script>document.write(“Thank YOU”) </Script>**