# Overview

In this lab you will write a program that implements a Java interface with some search-related methods . You will implement the class **ArraySearch** that implements the interface **Search**. Your class will work with a list of Comparable objects with the underlying implementation of the state of the class you program being an array.

# Objectives

* Practice creating a class that implements an interface
* Creating and using a constructor
* Creating and using non-static methods
* Perform comparisons of type Comparable
* Apply test cases to your program

# Steps

1. Create a folder on your local machine for your Java program, you can name it whatever you like
2. Start Visual Studio Code (VS Code)
3. In VS Code, Open that newly created folder.
4. Download the following starter files
   1. [Search.java](https://cs.unh.edu/~cs416/public/19L/Search.java)You **do not modify this file** which contains the interface Search that your program will implement
   2. Save the downloaded file in your project into the folder called **src**
5. Create a new class called **ArraySearch** in a file named **ArraySearch.java**.

## ArraySearch Class Requirements

* ArraySearch implements the Search interface. This implementation uses generics. The correct class declaration is given below. Here is [additional information](https://docs.oracle.com/javase/tutorial/java/generics/types.html) on declaring/using generic types when developing a class .

| public class ArraySearch<T extends Comparable<? super T>> implements Search { |
| --- |

* Class has only one constructor with the following signature (You need to use the name T as the generic type identifier in this assignment)

| public ArraySearch(T[] list) |
| --- |

* You are to have at least one instance variable of type **Comparable[]** (an array of Comparable objects.) The constructor should set it equal to the parameter named **list**.
* Implement all of the methods in the Search interface based on the comments in the interface using the **Comparable** array instance variable.
* For determining if the list is sorted, you may want to review the Java API for the [Comparable interface](https://docs.oracle.com/javase/8/docs/api/). Look for **Comparable** under the **All Classes** listing on the left hand side of the web page. The sort order is ascending, meaning that an example list of **Integers** (which are a Comparable type) could be **2, 10, 12, 20, 100**.
* For the index method, you may use any search technique of your choosing. But remember if you try a binary search, it requires the list to be sorted.
* The Comparable array (list) can contain no elements, but if it contains elements each object is unique (meaning there are no duplicate values in list passed to the constructor for testing.)