CS1112 Exercise 12 Name: _____ NetID: _____

You have until *Sunday, 5/3, at 9pm EDT* to submit problems 2, 3.2 and 3.3 on MATLAB Grader.

1 Review class `Interval`

1.1 Is the following implementation of instance method `overlap` for class `Interval` correct? Why or why not?

```
function Inter = overlap(self, other)
% If self and the other Interval overlap, then Inter is the handle of the
% overlapped Interval; otherwise Inter is an empty array of type Interval.
    left= max(self.left, other.left);
    right= min(self.right, other.right);
    if left >= right
        Inter= Interval.empty();
    end
end
```

1.2 Download the file `Interval.m` from the *Exercises* page. Complete/revise the three methods `getWidth`, `scale`, and `spanWith`. Then in the Command Window write some code to try out the new class definition, e.g.,

```
a= Interval(3,7);
w= a.getWidth()    % w should be 4
a.scale(2)
disp(a)            % a should be (3,11)
b= Interval(0,2);
c= a.spanWith(b)   % c should be (0,11)
```

2 Is one `Interval` in some array of `Intervals`?

Suppose we have an array of `Intervals` (.1,.5), (.2,.7), and (.8,.9). If we ask whether an `Interval` (.3,.4) "is in" the array of `Intervals`, we will answer "yes, yes, no" since
(.3,.4) is in (.1,.5),
(.3,.4) is in (.2,.7), but
(.3,.4) is not in (.8,.9).

Recall that in class `Interval` we have an `isIn` method that would be useful here. Make effective use of the `isIn` method in implementing the following function (in its own file). *Solve this problem using the full* MATLAB *environment first.*

```
function tf = isInRange(inter, interArray)
% inter: an Interval
% interArray: 1-d array of Intervals; interArray is not empty
% tf: a logical vector the same length as interArray where tf(k) is true
%     if inter "is in" interArray(k); otherwise tf(k) is false.
```

Test your function by typing the following in the *Command Window* (you will first need to download and read `intervalArray.m`):

```
a = intervalArray(4);  % Create a length 4 array of Intervals using the
                       %   function implemented in lecture.
b = Interval(.3,.5)
v = isInRange(b,a)     % Why isn't the function call   something.isInRange(...)  ?
                       % Answer: isInRange isn't an instance method; it's its own function.
                       % Check the type of v; it should be logical (not double).
```

Copy the body of your function `isInRange` into the code box for Problem 3.1 in MATLAB Grader. Test (and correct if necessary) your function.

3 More on class `LocalWeather`

Download file `LocalWeather.m` and read it. Ask questions if there are parts from what we did in lecture (constructor, method `showMonthData`) that you do not understand. Two of the methods, `getAnnualPrecip` and `getMonthlyAveTemps`, are incomplete (contains "dummy code" that does no calculation and only assigns a value to the return parameter); you will complete them later *using the full* MATLAB *environment* first.

3.1 Experiment! First, download the file `ithacaWeather.txt` which contains weather data for the City of Ithaca. In the *Command Window*, instantiate (create) a `LocalWeather` object using the data file:

```
ithaca= LocalWeather('ithacaWeather.txt')
```

You should see the properties of `ithaca` displayed. Note that one of the properties, `temps`, is an *array of* `Interval` *objects*. Type the following commands in the *Command Window*; make sure you understand the syntax for accessing values.

```
disp(ithaca.city)        % display the value in the property city

disp(ithaca.precip)      % display the values in the property precip--a vector!

disp(ithaca.precip(11))  % What is displayed?  What is it? ______________________________

disp(ithaca.temps)       % Matlab says it's a 1-by-12 array of INTERVALs

disp(ithaca.temps(11))   % Notice that the disp method in class Interval is
                         % used to show the data using Interval notation.

disp(ithaca.temps(11).left)  % What is displayed?  What is it? ___________________________
```

3.2 Implement function `getAnnualPrecip` which calculates and returns the total annual precipitation. If any month's precipitation data is missing, the returned value should be `NaN`, a value in `Matlab` of type `double` that indicates that a value is not-a-number.

Test your updated class. Save class `LocalWeather`, and type the following in the *Command Window*:

```
ithaca= LocalWeather('ithacaWeather.txt')  % instantiate object
% Which of the following two method calls is correct?  Try them!
a = getAnnualPrecip()
b = ithaca.getAnnualPrecip()
```

Change some values in the data file `ithacaWeather.txt` and call your method again. Test your work thoroughly.

3.3 Implement function `getMonthlyAveTemps` which returns the vector (length 12) of monthly average temperatures. Calculate a month's average temperature as the average between the month's high and low temperatures. See the function comment of `getMonthlyAveTemps` for how any missing temperature (`NaN`) should be handled. The built-in function `isnan` can be used to check whether a variable stores the value `NaN`: `isnan(x)` returns `true` if `x` is `NaN` and `false` otherwise.

Again, save and test your updated class. Back in the *Command Window*, call the instance method `getMonthlyAveTemps` to make sure that it works. Then change some data in the data file and test your method again.

Copy the contents of your completed file `LocalWeather.m` into the code box for Problem 3 in MATLAB Grader. Test (and correct if necessary) your class definition.