3. **Arrays**

(a) Write a method `MaxMin` that finds the maximum and minimum numbers in an array. You should use an accumulating parameter to store the results. [6]

(b) You are asked by a customer to re-implement their `BubbleSort` method. Bubble sort is a simple $O(n^2)$ sorting algorithm that works by repeatedly stepping through an array to be sorted comparing adjacent items and swapping them if they are in the wrong order. The customer specifies that the resulting array should be in ascending order. This means that after the first pass through the array the largest element should have bubbled its way to the top; the same procedure should be applied to the remaining $n - 1$ elements. [8]

To assist you, an example is provided below:

Bubble sorting the list $9, 8, 4, 6, 3$

First pass:

compare pair $(9, 8)$; swap as $9 > 8$; result $8, 9, 4, 6, 3$
compare pair $(9, 4)$; swap as $9 > 4$; result $8, 4, 9, 6, 3$
compare pair $(9, 6)$; swap as $9 > 6$; result $8, 4, 6, 9, 3$
compare pair $(9, 3)$; swap as $9 > 3$; result $8, 4, 6, 3, 9$

Second pass:

repeat process for list $8, 4, 6, 3$

Etc.

(c) Describe the terms *monomorphic* and *polymorphic* in the context of arrays. Using some Java code as an example, demonstrate how it is possible to implement seemingly polymorphic arrays in Java. [6]