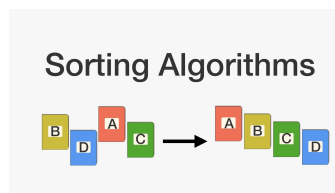**Midterm Exam Part 2 (level-2)** – A Hand-on Exam to implement different algorithmic requirements
50 points
Due (via your git repo) no later than 4:40 p.m., Thursday, 1st March 2021.

Part A (10 points):

The key focus is to test your understanding on algorithmic knowledge.

Sorting Algorithms

1. For simplicity, a starter code using the file named `isa.xml` is provided in the exam repository. The starter code is complete for the example provided below.

   example:

5	4	3	2	1

2. You are required to solve the sorting problem (in ascending order) for the input dataset provided below. Please note that by using the solution we can track the array changes based on i and j indices respectively. Modify and make edits to the `isa.xml` file to include your final solution.

   input dataset:

3	1	5	2	6	4

3. To provide additional support, the Insertion Sort algorithm is provided below for your reference:

Algorithm - Insertion(A)
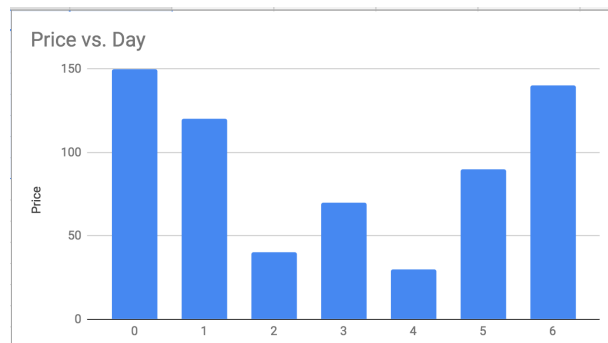
Input: an n-element un-sorted array A of integer values.

Output: an n-element sorted array A of integer values.

```
1: for i = 1 to n do
2:     key ← A[i]
3:     j ← i − 1
4:     while j >= 0 and A[j] > key) do
5:         A[j + 1] ← A[j]
6:         j ← j − 1
7:     end while
8:     A[j + 1] ← key
9: end for
```

PS next page.

Part B (25 points):

The key focus is to test your understanding on algorithmic knowledge.

Price vs. Day

1. For simplicity, a starter code using a file named `tsa.xml` is provided in the exam repository.

   example:

8	6	3	4	2	5	7

2. You are required to solve the stock span (time series) problem (in ascending order) for the input dataset provided below. Modify and make edits to the `tsa.xml` file to include your final solution.

   input dataset:

8	5	3	4	2	6	7	1

3. To provide additional support, the Stock Span series algorithm is provided below for your reference:

```
 1: Initialize an empty stack D
 2: Initialize an array S
 3: for i = 0 to n − 1 do
 4:    h ← 0
 5:    done ← false
 6:    while not (D is empty() or done) do
 7:       if P[i] >= P[D.top()] then
 8:          D.pop()
 9:       else
10:          done ← true
11:       end if
12:    end while
13:    if D is empty() then
14:       h ← −1
15:    else
16:       h ← D.top()
17:    end if
18:    S[i] ← i − h
19:    D.push(i)
20: end for
21: return S
```

PS next page.

Part C (15 points):

The key focus is to test your understanding on how to implement an algorithm.

1. The algorithm to be implemented is provided below:

> **Algorithm:** ExpenseTracker(E)
> **Input:** An n-element array E of expenses each day.
> **Output:** An n-element array A of values such that
> A[i] is the average of elements E[0], E[1], ... , E[i]
>
> 1: Initialize $a, i = 0$
> 2: **for** $(i = 0; i < n; i = i + 1)$ **do**
> 3: $a \leftarrow a + E[i]$
> 4: $A[i] \leftarrow a/i + 1$
> 5: **end for**
> 6: **return A**

2. For simplicity, a starter code using a file named `expense.py` and `Expense.java` is provided in the exam repository.

3. Make edits to the code based on your preference (Python or Java). The starter code is incomplete. Specifically, you are required to implement the compute method. The starter code already generates the input dataset. This input dataset has a series of expenses for an individual. The sample output or the expected output is purposefully left out from this requirement sheet. An underlying task being tested here is to find out what the algorithm does and what does the algorithm produce as an output. Analyze the algorithm provided above.

Submission Details

For this part of the exam, please submit the following to your GitHub repository by using the link shared to you by the Professor:

1. `isa.xml`, `tsa.xml`, and [`expense.py` or `Expense.java`].

2. A document to reflect your experience in this exam in a file named `reflections.txt`.

3. A document with the honor code pledge signed in a file named `honor-code.txt` document.

4. It is highly important, for you to meet the honor code standards provided by the college and to ensure that the submission is completed before the deadline. The honor code policy can be accessed through the course syllabus.