# Take home coding test

You will be given an input file in XML format representing the following structure:

**Table1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CorrelationID** | **NumberOfTrades** | **Limit** | **Value** | **TradeID** |
| 234 | 3 | 1000 | 100 | 654 |
| 234 | 3 | 1000 | 200 | 135 |
| 222 | 1 | 500 | 600 | 423 |
| 234 | 3 | 1000 | 200 | 652 |
| 200 | 2 | 2000 | 1000 | 645 |

Explanation of fields:

* **CorrelationID:** Group identifier of Trades.  Aggregation will be driven by this field.
* **NumberOfTrades:** Number of trades in the group. 1 to N.
* **Limit:** Upper limit that the group trade should respect.
* **Value:** Actual value of the trade, to be aggregated and compared to the limit.
* **TradeID:** Low level identifier of the atomic trade within the group.  Not to be aggregated.

The purpose of this exercise is to Accept or Reject trades **grouped by CorrelationID based** on a limit.

Take CorrelationID 234 as an example:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CorrelationID** | **NumberOfTrades** | **Limit** | **Value** | **TradeID** |
| 234 | 3 | 1000 | 100 | 654 |
| 234 | 3 | 1000 | 200 | 135 |
| 234 | 3 | 1000 | 200 | 652 |

After CorrelationID aggregation we get:

|  |  |  |  |
| --- | --- | --- | --- |
| **CorrelationID** | **NumberOfTrades** | **Limit** | **Value** |
| 234 | 3 | 1000 | 500 |

Total value of the trade is below the limit (1000 > 500) so this trade will be accepted.  Outputting:

|  |  |  |
| --- | --- | --- |
| **CorrelationID** | **NumberOfTrades** | **State** |
| 234 | 3 | Accepted |

**Assumptions and edge cases:**

* Groups that contain only 1 trade are to be expected (e.g. CorrelationID 222).  This will be a simple case of an inline comparison and output.
* Incomplete Groups need to be outputted as Pending (e.g. CorrelationID 200 where the second trade never arrived).
* ID fields are strings, all other fields are integers (positive and negative).
* Memory consumption and Big O notation are important.  Your solution will be profiled withy several input files.

So, in short, the results of Table1 will look like this:  
  
**Table2**

|  |  |  |
| --- | --- | --- |
| **CorrelationID** | **NumberOfTrades** | **State** |
| 200 | 2 | Pending |
| 222 | 1 | Rejected |
| 234 | 3 | Accepted |

**Deliverable:**

* You will be given an input file (**input.xml**) in xml format representing the structured mentioned above.
* Your solution should output 2 files.  One csv file with the results sorted by CorrelationID (**results.csv**) and a log file (**server.log**) with any exceptions, warnings or messages.
* You are free to architect your solution as you wish.  However your solutions entry point should be the command (the script name is to be taken literally):
  + python bnp-test.py input.xml
  + The results.csv file should have the structure of Table2 in a csv format (e.g. 234,3,Accepted).  **Please include a header row and make sure the results are ordered by CorrelationID (ascending). Sample provided.**
* Your code should be readable and testable.  Feel free to include comments if you want to explain anything.
* Please complete the items above and submit your project compressed in a zip file.