HW #2. Sort (Binary Input)

(Due: 3/22/2015)

The problem to be solved in this homework is exactly the same as HW1. You are given a set of integer sequences, and your program has to sort each of the sequences in ascending order. The only difference here is that HW2 uses binary formats for input.txt and output.txt. Specifically, the data in input.txt and output.txt are serialized in the MessagePack[[1](#_ENREF_1)] binary format. You may notice that it is actually easier to handle the input/output data with the MessagePack deserialization / serialization APIs. You may also notice that processing of the binary data is more efficient than processing of the text data as in HW1.

input.txt

{N:an **integer-type object** indicating the number of sequences in the input file},

{SEQINPUT\_1 : an **array-type object** corresponding to the first sequence of integers to be sorted},

{SEQINPUT\_2: an **array-type object** corresponding to the second sequence of integers to be sorted },

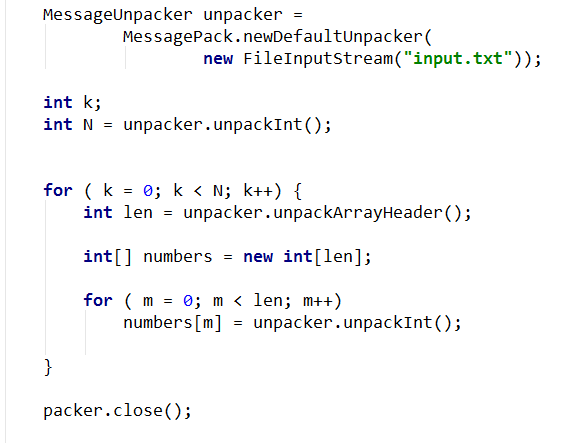
…

{SEQINPUT\_N: an **array-type object** corresponding to the Nth sequence of integers to be sorted }

Each object is stored in binary. The objects are placed back-to-back in the file.

Please refer to [[2](#_ENREF_2)] for the object types in MessagePack

Following is the corresponding deserialization code in Java



output.txt

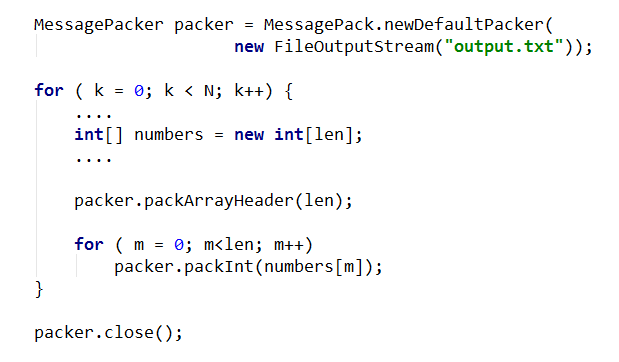
{SEQOUTPUT\_1: an array object containing the integer numbers of SEQINPUT\_1 sorted in ascending order},

{SEQOUTPUT\_2: an array object containing the integer numbers of SEQINPUT\_2 sorted in ascending order},

…

{SEQOUTPUT\_N: an array object containing the integer numbers of SEQINPUT\_N sorted in ascending order},

Following is the corresponding serialization code in Java



Reference

[1] Msgpack.org. *MesssagePack*. Available: <http://msgpack.org/>

[2] *MessagePack object types*. Available: https://github.com/msgpack/msgpack/blob/master/spec.md#types