ESS201 – C++ Programming
International Institute of Information Technology Bangalore

A3: C++ Classes and Composition
Announcement on Oct 14, 2020 (Wednesday), at 1:30 pm IST
Submission by Oct 30, 2020 (Friday), at 11:59 pm IST, on Domjudge and LMS

Highlight:
Build from A1 and A2:

- Improve the Histogram class
- Introduce a class for 5-number summary to be composed in the DataSequence class
- Computes statistical descriptors
- Writes in std output, the computed results

Details:
Input: The input is formatted into segments, and the segments are separated by -1 as delimiter, and terminating character(s). Examples of segment:

1. For creating a new DataSequence object:
   format: N,<#samples><space><comma-separated-nonnegative-real-values>
   example: N,4 3.4,1.3,2.51,3.24
2. For adding to an existing sequence, indexed starting from 0 from the start of the input:
   a) Adding to a sequence using an incorrect index must be considered as incorrect/invalid input
   format: A,<index of sequence to be added>,<value-to-be-added>
   example: A,0,3.4

A valid input altogether:
N,4 3.4,1.3,2.51,3.24 -1 N,6 3.3,2.3,1.5,2.4,2.6,3.4 -1 A,0,4 -1 A,1,3.4 -1 N,3 2.1,2.3,1.6 -1 A,1,2.3 -1 A,2,3.5 -1

Use STL vector (std::vector) to store DataSequence objects as created in the program. This input should create 3 DataSequence objects and stored in "DSVector":
std::vector<DataSequence> DSVector;
…
DSVector[0] has 5 samples -- [3.4,1.3,2.51,3.24,4]
DSVector[1] has 8 samples -- [3.3,2.3,1.5,2.4,2.6,3.4,3.4,2.3]
DSVector[2] has 4 samples -- [2.1,2.3,1.6,3.5]

Check for invalid inputs and handle incorrect inputs as done in A1 and A2. A DataSequence with 0 samples must not be created, and the segment starting N may be ignored.
N,4 3.4,1.3,2.51,3.24 -1 N,6 3.3,2.3,1.5,2.4,2.6,3.4 -1 A,0,4 -1 A,1,3.4 -1 N,3 2.1,2.3,1.6 -1 A,1,2.3 -1 A,2,3.5 -1 N,0 -1 N,0 -1
is a valid input, that creates a DSVector with 3 DataSequence objects only.

Output: The output must provide segments for each DataSequence object, separated by space.

Each segment must have the comma-separated 5-number summary in comma-separated, without-spaces, with upto 4 decimal places followed by space and followed by the histogram output, for 10 bins, as done in A2, and terminated by -1. Format for each segment:

<minimum>,<first-quartile>,<median>,<third-quartile>,<maximum> <comma-separated-11-bin-values> <comma-separated-10-bin-normalized-frequencies> -1

Final output:
<segment for DSVector[0]> <segment for DSVector[1]> <segment for DSVector[2]> … so on.
e.g. for DSVector[0], the five-number summary is 1.3, 2.51, 3.24, 3.4, 4.0; and for DSVector[1] is 1.5, 2.3, 2.5, 3.35, 3.4
for DSVector[0], the bin-values are 1.3 , 1.57, 1.84, 2.11, 2.38, 2.65, 2.92, 3.19, 3.46, 3.73, 4.
and normalized bin-frequencies or bin-probabilities are 0.2, 0. , 0. , 0. , 0.2, 0. , 0. , 0.4, 0. , 0.2

Sample input and output:
N,4 3.4,1.3,2.51,3.24 -1 N,6 3.3,2.3,1.5,2.4,2.6,3.4 -1 A,0,4 -1 A,1,3.4 -1 N,3 2.1,2.3,1.6 -1 A,1,2.3 -1 A,2,3.5 -1

~~1.3,2.51,3.24,3.4,4.0 1.3,1.57,1.84,2.11,2.38,2.65,2.92,3.19,3.46,3.73,4.0 0.2,0.0,0.0,0.0,0.0,0.0,0.2,0.0,0.0,0.0,0.4,0.0,0.2 -1 1.5,2.3,2.5,3.35,3.4 1.5,1.69,1.88,2.07,2.26,2.45,2.64,2.83,3.02,3.21,3.4 0.125,0.0,0.0,0.0,0.0,0.375,0.125,0.0,0.0,0.0,0.0,0.375 -1 1.6,1.85,2.2,2.9,3.5 1.6,1.79,1.98,2.17,2.36,2.55,2.74,2.93,3.12, 3.31,3.5 0.25,0.0,0.25,0.25,0.0,0.0,0.0,0.0,0.0, 0.0,0.25 -1~~
1.3000,2.5100,3.2400,3.4000,4.0000
1.3000,1.5700,1.8400,2.1100,2.3800,2.6500,2.9200,3.1900,3.4600,3.7300,4.0000
0.2000,0.0000,0.0000,0.0000,0.2000,0.0000,0.0000,0.4000,0.0000,0.2000 -1 1.5000,2.3000,2.5000,3.3500,3.4000
1.5000,1.6900,1.8800,2.0700,2.2600,2.4500,2.6400,2.8300,3.0200,3.2100,3.4000
0.1250,0.0000,0.0000,0.0000,0.3750,0.1250,0.0000,0.0000,0.0000,0.3750 -1 1.6000,1.8500,2.2000,2.9000,3.5000
1.6000,1.7900,1.9800,2.1700,2.3600,2.5500,2.7400,2.9300,3.1200,3.3100,3.5000
0.2500,0.0000,0.2500,0.2500,0.0000,0.0000,0.0000,0.0000,0.0000,0.2500 -1
// Note: The output is to be printed as a single line. For better readability and line overflow, the above is split across lines. //

Resources:
https://en.wikipedia.org/wiki/Five-number_summary

Code – Classes:

1. Create a class for 5-number summary “FiveNumberSummary” with data members for the 5 statistical descriptors. The DataSequence class must have a data member of FiveNumberSummary type. Update the values in the data member of DataSequence, whenever the float pointer of data values in the DataSequence is updated. This is an example of composition.
2. In the Histogram class, store a value for the number of samples N_s used to compute the frequency distribution. Add a data member for the same. The default value for number of bins N_b is 10.
3. In A3, Histogram class can now be changed to be an association with DataSequence class. Update the Histogram class, when the float pointer in DataSequence is updated.
4. The bin-frequencies in the Histogram now has to be normalized. Use the N_s to do the same.
5. Add friend functions for std::ostream for DataSequence, FiveNumberSummary, and Histogram classes, which must be used appropriately. The std::ostream for DataSequence should call the same for FiveNumberSummary and Histogram classes. For your program output, std::cout must be directly called on the DataSequence objects.

Helpful Hints:
For verifying your results, you may use Python libraries.
>>> import numpy as np
>>> a=np.array([3.4,1.3,2.51,3.24,4.0])
>>> ahist=np.histogram(a)

```
>>> abf=np.array([x/float(np.shape(a)[0]) for x in ahist[0])
>>> abv = ahist[1]
>>> afns = [np.min(a), np.percentile(a,25,interpolation='midpoint'), np.median(a), np.percentile(a,
75,interpolation='midpoint'), np.max(a)]
```

afns gives five-number summary, abv gives bin-values, abf gives normalized bin-frequencies.