

## ECE 2305 – Introduction to C Programming

### Programming Project 07 Histograms

The file named “*numdata.txt*” contains a list of type **double** numbers. Write a C++ program to open the file and read the data into a 1D array. Write the program so that the program determines the number of elements that are required for the array. Use the data in the array to create two output data files. One with a sorted list of numbers from smallest value to largest value. The second output data file shall be a histogram table with the following form.

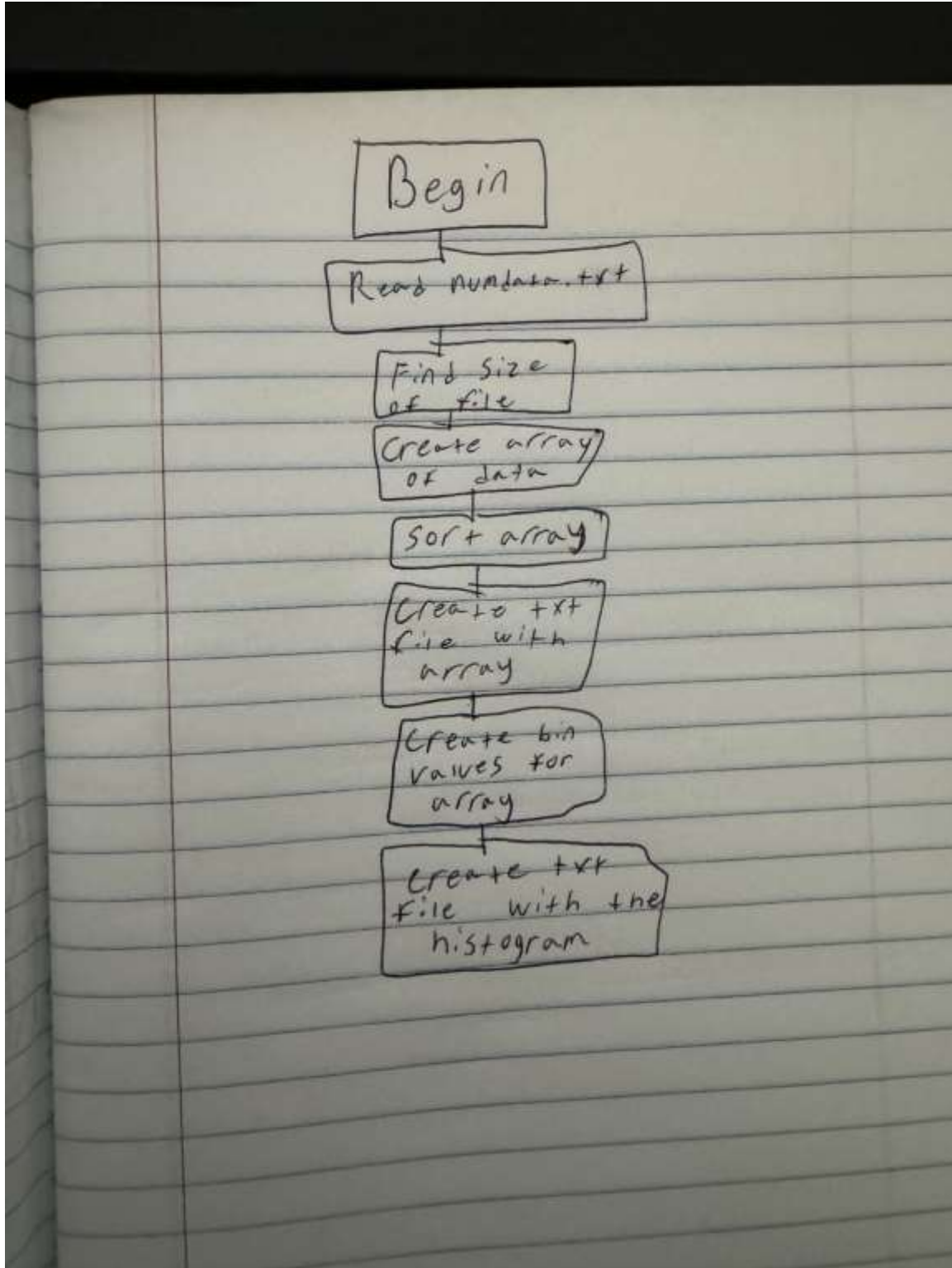
Bin	Number of Values
0-10	$0 \leq \text{value} < 10$
10-20	$10 \leq \text{value} < 20$
20-30	$20 \leq \text{value} < 30$
30-40	$30 \leq \text{value} < 40$
40-50	$40 \leq \text{value} < 50$
50-60	$50 \leq \text{value} < 60$
60-70	$60 \leq \text{value} < 70$
70-80	$70 \leq \text{value} < 80$
80-90	$80 \leq \text{value} < 90$
90-100	$90 \leq \text{value} < 100$

Document the program with:

A. A brief description of the purpose of the program and the structure of the program.

The purpose of the program is to read a list of values, sort them, and create a histogram table with them. To do so, the main function begins by reading the file, then finds how many values there are. It then creates an array with those values and sorts them. Next it creates a text file with the sorted array, then a text file with the histogram file of the array.

B. A flowchart to describe the structure of the program.



### C. The code listing.

```
1  ECE 2305-Programming Project 7-Kaleb Badgett
2  #include <iostream>
3  #include <fstream>
4  using namespace std;
5
6  int main()
7
8      ifstream fin("c:numdata.txt", ios::in); //Read file
9      double dataIn;
10     int size = 0;
11     bool fail = 0;
12
13     while (fail == 0)
14     {
15         fin >> dataIn;
16         fail = fin.fail();
17         if (fail == 0) size = size + 1; //Find soze of file
18     }
19     fin.clear(); //Reset file values
20     fin.seekg(0, ios::beg); //Reset file values
21
22     double* data;
23     data = new double[size];
24     for (int n = 0; n < size; n++) fin >> data[n]; //Create array with file values
25
26     double t = 0;
27     for (int m = size ; m > 0; m--)
28     {
29         for (int n = 0; n < m; n++)
30         {
31             if (data[n] < data[n - 1])
32             {
33                 t = data[n - 1];
34                 data[n - 1] = data[n];
35                 data[n] = t;
36             }
37         }
38     } //Bubble sort array smallest to largest.
39
```

```

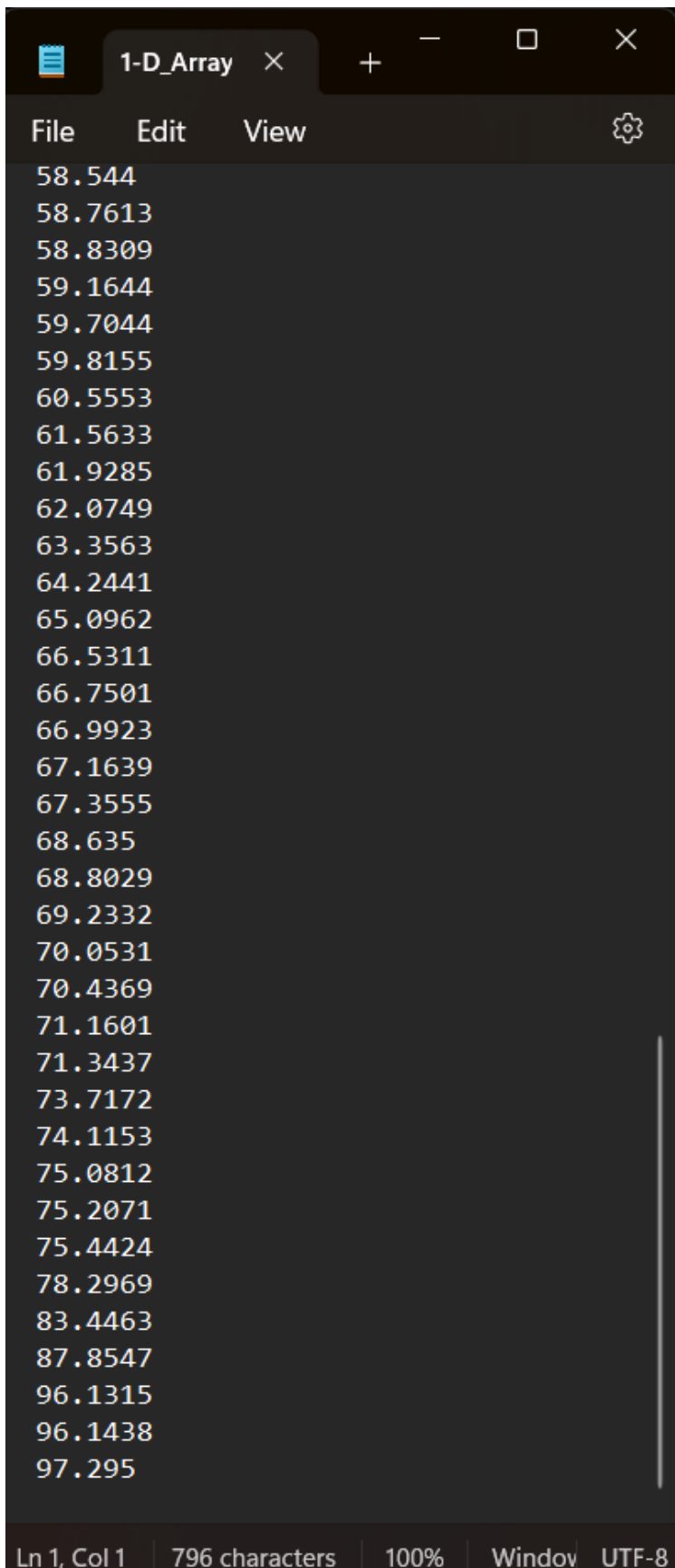
40 //1-D ARRAY OUTPUT FILE
41 ofstream fout("c:1-D_Array.txt", ios::out);
42     fout << "Sorted 1-D Array" << endl;
43     for(int n = 0; n < size; n++)fout << data[n] << endl;
44     fout.close();//Create file with array
45
46
47 //Histogram time
48 int b[10];
49 for(int n = 0; n < 10; n++) b[n] = 0;
50 for(int n = 0; n < size; n++)
51 {
52     if (data[n] < 10) b[0] = b[0] + 1;
53     else if (data[n] < 20) b[1] = b[1] + 1;
54     else if (data[n] < 30) b[2] = b[2] + 1;
55     else if (data[n] < 40) b[3] = b[3] + 1;
56     else if (data[n] < 50) b[4] = b[4] + 1;
57     else if (data[n] < 60) b[5] = b[5] + 1;
58     else if (data[n] < 70) b[6] = b[6] + 1;
59     else if (data[n] < 80) b[7] = b[7] + 1;
60     else if (data[n] < 90) b[8] = b[8] + 1;
61     else b[9] = b[9] + 1;//Find histogram bin values
62 }
63
64 // for output file
65 ofstream dout("c:hist.txt", ios::out);
66
67     dout << "Bins \tNumber of Values" << endl;
68     int lowBins = 0;
69     int highBins = 10;
70     for(int n = 0; n < 10; n++)
71     {
72         dout << lowBins << "-" << highBins << " \t" << b[n] << endl;
73         lowBins = lowBins + 10;
74         highBins = highBins + 10;
75     }
76     dout.close();//Create histogram text file
77
78

```

D. Two output *.txt* data files.

The image displays two side-by-side code editor windows. The left window contains a list of 40 floating-point numbers, starting with 1-D Array and ending with 40.318. The right window, titled '1-D\_Array', contains a list of 20 floating-point numbers, starting with 42.3074 and ending with 62.0749. Both windows have a menu bar with 'File', 'Edit', and 'View' options, and a status bar at the bottom indicating 'Ln 1, Col 1', '796 characters', '100%', 'Window', and 'UTF-8'.

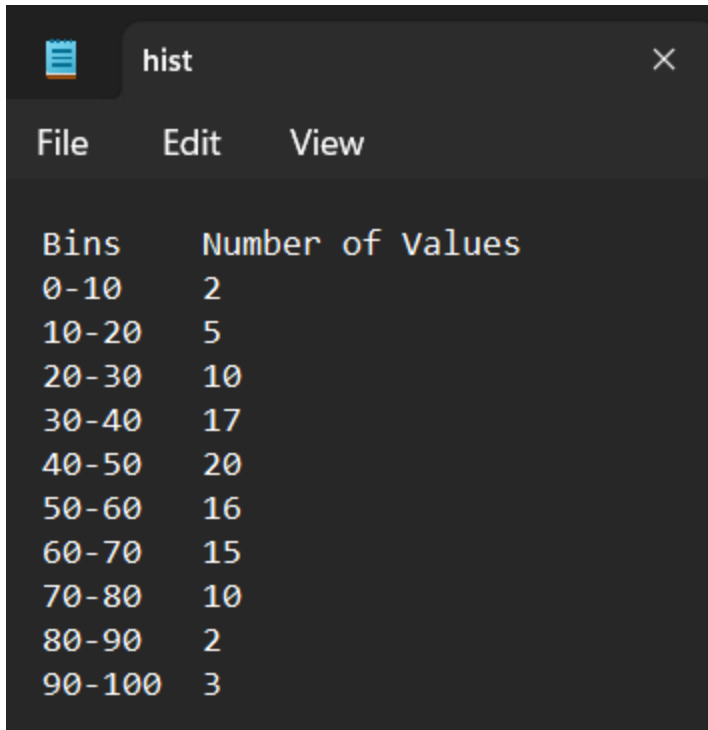
Index	Value
1	1-D Array
2	6.314
3	9.9037
4	11.7867
5	12.355
6	15.1307
7	15.9809
8	18.6846
9	24.3178
10	26.034
11	26.1869
12	26.4744
13	26.5111
14	27.5337
15	28.6472
16	28.9464
17	29.3236
18	29.5748
19	30.1163
20	31.0618
21	31.6929
22	31.7845
23	32.2325
24	33.0614
25	33.1053
26	33.1489
27	33.6985
28	35.158
29	36.2232
30	36.4715
31	37.5937
32	38.5754
33	38.8976
34	39.0651
35	39.3668
36	40.318
37	42.3074
38	42.7359
39	43.2281
40	43.3786
41	43.4411
42	43.9099
43	45.0508
44	46.1802
45	46.8569
46	48.2937
47	48.8423
48	49.0546
49	49.457
50	49.4615
51	49.9078
52	49.9175
53	49.9718
54	51.6078
55	52.0162
56	52.7128
57	54.4748
58	54.9835
59	55.0491
60	55.8461
61	57.79
62	58.2062
63	58.4486
64	58.544
65	58.7613
66	58.8309
67	59.1644
68	59.7044
69	59.8155
70	60.5553
71	61.5633
72	61.9285
73	62.0749



The image shows a code editor window with a dark theme. The title bar at the top contains a file icon, the text '1-D\_Array', and standard window controls (close, maximize, zoom in, zoom out). Below the title bar is a menu bar with 'File', 'Edit', and 'View' options, followed by a settings gear icon. The main text area contains a list of 30 numerical values, each on a new line. The values are: 58.544, 58.7613, 58.8309, 59.1644, 59.7044, 59.8155, 60.5553, 61.5633, 61.9285, 62.0749, 63.3563, 64.2441, 65.0962, 66.5311, 66.7501, 66.9923, 67.1639, 67.3555, 68.635, 68.8029, 69.2332, 70.0531, 70.4369, 71.1601, 71.3437, 73.7172, 74.1153, 75.0812, 75.2071, 75.4424, 78.2969, 83.4463, 87.8547, 96.1315, 96.1438, and 97.295. At the bottom of the window is a status bar with the following information: 'Ln 1, Col 1', '796 characters', '100%', 'Window', and 'UTF-8'.

```
58.544
58.7613
58.8309
59.1644
59.7044
59.8155
60.5553
61.5633
61.9285
62.0749
63.3563
64.2441
65.0962
66.5311
66.7501
66.9923
67.1639
67.3555
68.635
68.8029
69.2332
70.0531
70.4369
71.1601
71.3437
73.7172
74.1153
75.0812
75.2071
75.4424
78.2969
83.4463
87.8547
96.1315
96.1438
97.295
```

Ln 1, Col 1 | 796 characters | 100% | Window | UTF-8



A terminal window titled "hist" with a close button (X) in the top right corner. The window contains a table with two columns: "Bins" and "Number of Values". The table lists 11 bins from 0-10 to 90-100 with their corresponding frequency values.

Bins	Number of Values
0-10	2
10-20	5
20-30	10
30-40	17
40-50	20
50-60	16
60-70	15
70-80	10
80-90	2
90-100	3

Upload the PDF and the TXT files to blackboard.