02393 C++ Programming Exercises

Assignment 3

To be handed in via CodeJudge — `https://dtu.codejudge.net/02393-e21/assignments`

1 Fun with bags

This assignment is about processing a sequence of commands that query and update a bag of numbers. Your program has to read a sequence of commands from `cin` and provide some reply to `cout`. The commands are:

`add` x: add number x to the bag. Do not produce any output;

`del` x: if x is in the bag, remove it; otherwise, do nothing. Do not produce any output;

`qry` x: if x is in the bag then output `T`, otherwise output `F`;

`quit`: end the program.

The goal of the two exercises below is always the same (reply to the input, assuming an initially empty bag) but the nature of the bag varies.

Fun with bags 1. Here you have to consider that the bag is a set of `int` values between 0 and 1000 (included).
Example:
input: `add 1 add 2 add 1 del 1 qry 1 qry 2 quit`
output: `FT`

Fun with bags 2. Here you have to consider that the bag is a *multiset* of `int` values between 0 and 1000 (included). The main difference with respect to the previous exercise is that now repetitions are allowed. This means, for example, that deleting x just removes one occurrence of x from the multiset.
Example:
input: `add 1 add 2 add 1 del 1 qry 1 qry 2 quit`
output: `TT`

Hints. You can solve both exercises using arrays, maybe reusing ideas from Assignment 2...

If you want, you can also use **C++ containers**. We will see containers later in the course, but you may already try to use them to get a first feeling of how convenient they are with respect to array-based structures. In particular, you could consider these two classes of containers:

- `http://en.cppreference.com/w/cpp/container/set`
- `http://en.cppreference.com/w/cpp/container/multiset`

For every command of the exercise, there is a method/function that does (almost) what you need.

2 Histogram

A *histogram* represents the distribution of a dataset into discrete intervals. Consider for instance the data set given by the integer numbers 100 95 47 88 86 92 75 89 81 70 55 80; suppose we want build a histogram with 11 intervals $[0-10)$, $[10-20)$, ..., $[100-110)$ to be textually represented as follows: *(meaning: there are 0 numbers in the interval* $[0-10)$*, and 5 numbers in* $[80-90)$*, etc.)*

```
0: 0
10: 0
20: 0
30: 0
40: 1
50: 1
60: 0
70: 2
80: 5
90: 2
100: 1
```

Write a program that reads the following values (in this order) from the standard input (cin):

- the number ℓ of intervals (e.g. 11 in the example above)
- the size n of the data set (e.g. 12 in the example above)
- and n non-negative integers

and then outputs the histogram in the above format.

Hints. Let each interval have integer size $k = \lceil \frac{m}{\ell} \rceil$, where m is the maximum number in the data set. That is, interval i should be $[(i-1) \times k \ldots i \times k)$. The function $\lceil \cdot \rceil$ is implemented as function ceil() in the library math.h

As an example, the input for the histogram above (11 intervals, 12 values) is:

```
11 12 100 95 47 88 86 92 75 89 81 70 55 80
```

hence, we have $k = \lceil \frac{100}{11} \rceil = \lceil 9,0909\ldots \rceil = 10$, and thus the i-th interval starts at $(i-1) \times 10$. E.g., the last interval (11-th) starts at $(11-1) \times 10 = 100$. You can see it in the output above.

Another example: with the same data above but interval size $\ell = 7$ we have the input:

```
7 12 100 95 47 88 86 92 75 89 81 70 55 80
```

hence the intervals size is $\lceil \frac{100}{7} \rceil = \lceil 14,2857\ldots \rceil = 15$, and the resulting output is:

```
0: 0
15: 0
30: 0
45: 2
60: 1
75: 6
90: 3
```

Special case. Consider the input: `2 4 8 6 3 1`. Here the maximum number $m = 8$ in the dataset can be divided by the number of intervals $l = 2$. Technically, the second (and last) interval should be $[4, 8)$, thus excluding 8 from the histogram. To solve this problem, we need to check for a special condition: if m can be divided by l, then we include m in the last interval. This way, for the input `2 4 8 6 3 1` we obtain the histogram:

```
0: 2
4: 2
```

rather than:

```
0: 2
4: 1
```