# C++ Programming: Judge Assignment 2 (JA2)

The following tasks should be submitted to the SoftUni Judge system, which will be open starting Saturday, 22 April 2017, 10:00 (in the morning) and will close on Sunday, 30 April 2017, 23:59. You will be provided with a link to the “contest” (where you will submit the assignment) later.

Solutions for each task will be submitted in the form of compressed archive (.zip) files, containing .h and .cpp files. Depending on the task, some .h and/or .cpp files will be available in the Judge system and your code will be compiled alongside them (so that either your files can use them, or the other files will use the files you submitted). The files available for a task we will call a “solution skeleton”. Detailed instructions on what solution skeletons you are provided with, as well as instructions for submitting your code, are given in each task.

Please be mindful of the strict input and output requirements for each task, as well as any additional requirements on running time, used memory, etc., as the tasks are evaluated automatically and not following the requirements strictly may result in your program’s output being evaluated as incorrect, even if the program’s logic is mostly correct.

You can use C++03 and C++11 features in your code.

Unless explicitly stated, any integer input fits into int and any floating-point input can be stored in double.

NOTE: the tasks here are NOT ordered by difficulty level.

**NOTE: memory and time restriction for these tasks may be altered slightly up to Friday 21 April 23:59 (we’re still testing out the feature for multiple file submission for C++ projects). Please check the task descriptions in the Judge system (when it opens on 22 April, 10:00) for the final memory and time restrictions.**

## Task 1 – List (JA2-Task-1-List)

You are given a List.h file containing the declarations for a List class representing a linked list, and a main.cpp file, which defines a main() function and uses the List class to merge several sorted lists from the standard input into a single sorted list printed on the standard output.

* Create a List.cpp file which contains the implementation of the List class
* The files should successfully compile together (e.g. in a Code::Blocks project with MinGW)
* The resulting program should correctly merge sorted lists read from the console into a sorted list, which should be printed on the console (the main.cpp file does this if you implement List.cpp correctly)
* Submit a .zip file containing the List.cpp file and nothing else

How you choose to implement the linked list is up to you, but you should make sure all the public methods of the List class work correctly, as they are used by main.cpp. The declarations in List.h should be mostly self-explanatory, but if you are unsure what a method should do – just see how main.cpp uses it and make sure you implement it so that the program works correctly.

You are NOT allowed to modify main.cpp or List.h.

The task this program solves is merging multiple sorted (ascending) lists of integer numbers into a single sorted (ascending) list of integer numbers. For example, the lists 1 17, and -3 6 25 42 should be merged into the following list: -3 1 6 17 25 42.

### Input

One or more lines, each of which containing from 1 to 1000 integers, separated by single spaces. The final line will not contain numbers and will only contain the string "end"

### Output

A single line, containing the numbers of the merges sorted list, in ascending order, separated by single spaces

### Restrictions

The total number of elements entered in the input will NOT exceed 100000

The number of elements per input list (line) will NOT exceed 1000

The total running time of your program should be no more than 0.5s

The total memory allowed for use by your program is 5MB

### Example I/O

|  |  |
| --- | --- |
| Example Input | Expected Output |
| 1 17  -3 6 25 42  end | -3 1 6 17 25 42 |
| 4  5 6  1 2 3  end | 1 2 3 4 5 6 |
| 1 3  2  end | 1 2 3 |