First homework for Architecture.

Variant:

- *Artifact 10: Well of wisdom*
- *Function 10: Straight insertion sort*

Files:

- **main.cpp(1,83KB)**: main function which reads, writes and catches exceptions coming from the container...
- **container.cpp(1,39KB)/.h(644B)**: container with all the functions for containing inputted objects .
- **baseLines.cpp(2,47KB)/.h(768B)**: basic well of wisdom structure with all the functions.
- **aphorism.cpp(925B)/.h(478B)**: aphorism structure with all the functions.
- **proverb.cpp(877B)/.h(432B)**: proverb structure with all the functions.
- **riddle.cpp(883B)/.h(408KB)**: riddle structure with all the functions.

Command line input guide:

1) Write *./task01 -f [inputFileName].txt [outputFileName].txt [sortedOutputFileName].txt* for file input. 2) Write *./task01 -n [number of wells of wisdom] [outputFileName].txt [sortedOutputFileName].txt* for random input generation.

File input guide:

You need to input a couple of matrices according to this template:

1) Input type: 1 for aphorism, 2 for proverb, 3 for riddle. 2) Input text of artifact: line of chars with length 10.000 or less. 3) Input meta text for artifact (author name, country or riddle answer): line of chars with length 10.000 or less.

Tests:

There are 8 tests stored in tests directory, which could be used to understand if the program runs properly and tests could be autogenerated using command line.

Memory:

Stack
main
Init
In / InRnd
Out
Sorting

Heap
"task"
"-r" / "-f"
"input00.txt"
"output00.txt"
"outSorted.txt"

TT

Compoment name	Size (in bytes)
struct ridlle	**80000**
char fstring[10000]	40000

Compoment name	Size (in bytes)
char sstring[10000]	40000
struct aphorism	**80000**
char fstring[10000]	40000
char sstring[10000]	40000
struct proverb	**80000**
char fstring[10000]	40000
char sstring[10000]	40000
struct baseLines	**80004**
enum key	4
union { aphorism a;proverb p;riddle r;}	80000
struct container	**800040008**
enum {max_len = 10000}	4
int len	4
baseLines *cont[max_len]	800040000

Modules:

- Interface modules count: 5
- Implementation modules count: 6