C++ Advanced – Exam 1 (07 Apr 2019)

Write C++ code for solving the tasks on the following pages.

Code should compile under the C++11 standard.

Submit your solutions here: https://judge.softuni.bg/Contests/1441/CPlusPlus-Advanced-Exam-07-Apr-2019

Any code files that are part of the task are provided under the folder **Skeleton**.

Please follow the exact instructions on uploading the solutions for each task.

Task 2 – Bytes Parsing

Your task is to write program, which reads/represents/parses numbers out of contiguous array of bytes in memory into C++ primitive data type numbers. You are given the **`main()`** function, which reads two string values (as whole rows) of memory.

- The first string value indicate your command buffer. The buffer may only contain the letters 's', 'i', 'l' (in any order and in any number of occurrences);
  - 's' – stands for 'short' C++ primitive data type;
  - 'i' – stands for 'integer' C++ primitive data type;
  - 'l' – stands for 'long long' C++ primitive data type;
- The second string value contains your data buffer (as a single row of data).
  Each character of the second string will be in the range **[0, 9]** inclusive;
  After the read from the console an -= '0' operation is performed on each char so the remaining value is the actual integer values from 0 to 9;
  Keep in mind that the command buffer may contain commands, which you have no data for in your data buffer. When you reach such case -> simply ignore the rest of the commands from the command buffer.

You should implement the functions **parseData()** and **printResult()** in another .cpp file. (For example BytesParsing.cpp)

Keep in mind that the Judge system has a 64bit Little-endian architecture so:

- sizeof(short) is 2 bytes;
- sizeof(int) is 4 bytes;
- sizeof(long long) is 8 bytes;

Example:

command - "sil"

bufferData – "10200030000000"

- First parsed number is 'short' and first 2 bytes "[0-1]" are represented as an 'short';
- Second parsed number is 'int' and next 4 bytes from index [2-5] are represented as an 'int';
- Third parsed number is 'long long' and next 8 bytes from index [6-13] are represented as an 'long long';

The result is "1 2 3"

Example 2:

command - “silll”

bufferData – “10200030000”

The result is “1 2 Warning, buffer underflow detected”

Data buffer does not have enough information about all the listed commands. All the parsed number so far are printed first.

As a result of **parseData()** – you should print a status message depending on the received **ErrorCode** in **printResult()** followed by a **newline**.

You should print:

- For successful allocation parsing – all parsed numbers divided by a **whitespace** (the last number should also have a whitespace before the newline);
- For partial parsing (more requested commands than actual data to parse) – all **successfully** parsed numbers so far divided by a **whitespace** followed by “Warning, buffer underflow detected” (the last number should also have a whitespace before the “Warning part”);
- For preventing an empty command buffer or data buffer – “No input provided”;

Your task is to study the code and implement the function so that the code accomplishes the task described.

You should submit a single **`.zip`** file for this task, containing **ONLY** the files you created.

The Judge system has a copy of the other files and will compile them, along with your file, in the same directory.

Restrictions

You are free to implement another function/functions that are used internally by the **parseData ()** and **printResult().**

The command buffer and data buffer size will be in the range **[0, SIZE_T_MAX]** inclusive;

Note: ‘size_t’ and ‘unsigned long long integer’ are the same thing;

Examples

Input	Output
ss 2002	2 512
is 11110	16843009 Warning, buffer underflow detected
sil 900200003000000	9 512 196608