**Lab #2: Nanotable (0)   
CS1010 AY2017/8 Semester 1   
Date of release: 4 September 2017, Monday, 8am.   
Submission deadline: 13 September 2017, Wednesday, 5pm.   
School of Computing, National University of Singapore**

**0 Introduction**

**0.0 Lab guidelines**

**Important:** Please read [Lab Guidelines](http://www.comp.nus.edu.sg/~cs1010/labs/2017s1/labguide.html) before you continue.

This lab requires you to do 3 simple exercises, which altogether lead to a simple table system.   
  
You may assume that the input data are according to specification, and hence there is no need to do input data validation, unless otherwise stated.   
The maximum number of submissions for each exercise is **15**.

Please do not use features such as **arrays**, **pointers** and **recursion** which are not yet covered in class. This exercise will be extended in a future lab whereby these features may be added.   
  
In general, for all lab exercises, do not use features not covered yet. When in doubt, please raise them on IVLE forum or with your DL.

If you have any questions on the task statements, you may post your queries **on the relevant IVLE discussion forum**. However, do **not** post your programs (partial or complete) on the forum before the deadline!

**0.1 Nanotable**

We are now in the age of [BIG DATA](https://www.forbes.com/sites/bradpeters/2012/07/12/the-age-of-big-data/#1711ad0d4f66). Huge volumes of data are stored, transferred and analyzed, for which lots of tools are developed. [Google Bigtable](https://en.wikipedia.org/wiki/Bigtable) is one of most famous data storage systems. In this class, we are going to build something similar, yet much much smaller. It is called Nanotable, which is a data storage system for exam records. It allows users to insert exam scores and do some basic analysis. In this lab, as a first step, we are going to implement some fundamental features. In subsequent labs, we will enhance this Nanotable by adding more functionalities.

**1 System features**

This is the first step towards a table system, which we shall name as **Nanotable0**. Nanotable0 has the following features:

* Parse commands and take actions accordingly. This is the top-level interactive feature of the system which will be implemented in a similar style as a Read-Eval-Print-Loop (What is [REPL](https://en.wikipedia.org/wiki/Read%E2%80%93eval%E2%80%93print_loop)?). Currently, there are only four valid commands: **help**, **exit**, **sum** and **ave**. If **help**, **sum** or **ave** is typed in, some actions will be triggered and the system will wait for the next command after the execution of the actions. If **exit** is typed in, the system will exit. **exit** is the only command to quit the system. If some invalid commands are typed in, it will be rejected and the loop will wait for the next input.
* When the input is "help", the help information is printed. Then the system waits for input again.
* When the input is "exit", the loop breaks and the system exits.
* When the input is "sum", the system asks for the input of "number of integers" N and then asks for input of these N integers and prints the sum of these N integers. Then the system waits for input again.
* When the input is "ave", it is similar to the case of "sum" but the printed result will be the average of these N integers, not the sum.
* Note that the symbol N is used above. However, in writng programs, you should give more descriptive names for variables. Also remember that by convention, constants are named using capital letters so N would normally mean a constant.

**Code skeleton**

You are provided a [skeleton code](http://www.comp.nus.edu.sg/~cs1010/labs/2017s1/lab2/skeleton/nanotable0_skeleton.c). You may develop your system based on that or you may ignore the skeleton code and write everything from scratch yourself.

In this skeleton code, we have implemented some utilities for you:

* check\_command() - input is some command and output is an integer indicating which command it is. Details are provided in the comments in the skeleton code.
* print\_help() - print the help information.

Things you need to do:

* Fix the system so that it loops until the user enters the "exit" command.
* Implement simple\_sum() and simple\_average() as required.

**Sample input & output**

Sample input and outputs are provided below.

* [Inputs](http://www.comp.nus.edu.sg/~cs1010/labs/2017s1/lab2/testdata_for_students/input)
* [Outputs](http://www.comp.nus.edu.sg/~cs1010/labs/2017s1/lab2/testdata_for_students/output)

**2 Command parser**

The command parser is in procedure parse\_command(), which is partially implemented. It is now only taking one input command and will exit after that. You need to modify it so that it works in a REPL style (What is [REPL](https://en.wikipedia.org/wiki/Read%E2%80%93eval%E2%80%93print_loop)?). The action to the command help has been implemented in the procedure print\_help().

**Sample runs**

Sample run using interactive input (user's input shown in blue)   
Sample run #1:

Welcome to Nanotable!

Type "help" for more information...

Waiting for command...

help

Commands available: sum, ave, help, exit

Waiting for command...

help

Commands available: sum, ave, help, exit

Waiting for command...

exit

See you!

Sample run #2:

Welcome to Nanotable!

Type "help" for more information...

Waiting for command...

help

Commands available: sum, ave, help, exit

Waiting for command...

help

Commands available: sum, ave, help, exit

Waiting for command...

test

No such command: test, please input command again!

Waiting for command...

haha

No such command: haha, please input command again!

Waiting for command...

exit

See you!

**3 Sum calculator**

When the input command is "sum", the command parser will callsimple\_sum(), which you need to implement. It first asks users to type in the number N of the coming integers and then takes the followingN integers and print its sum.

**Sample runs**

Sample run #1:

Welcome to Nanotable!

Type "help" for more information...

Waiting for command...

sum

Please indicate the number of integers:

5

Please input the 1st number:

1

Please input the 2nd number:

2

Please input the 3rd number:

3

Please input the 4th number:

4

Please input the 5th number:

5

sum is 15

Waiting for command...

exit

See you!

Please be careful that in the line Please input the 1st number:, you need to output the correct ordinal form of the number. For example, 1-->1st, 2-->2nd, 21-->21st, etc. The input number range from 1 to 1000 and each integer range from 1 to 100.

**4 Average calculator**

Similar to 2. You also need to be careful to the print of ordinal number. The calculated result will be rounded to integer. The rounding is demonstrated by examples: 1.1 --> 1, 1.2-->1, 1.3-->1, 1.4-->1, 1.5-->2, 1.6-->2, etc.

**Sample runs**

Sample run #1:

Welcome to Nanotable!

Type "help" for more information...

Waiting for command...

ave

Please indicate the number of integers:

3

Please input the 1st number:

1

Please input the 2nd number:

2

Please input the 3rd number:

5

average is 3

Waiting for command...

ave

Please indicate the number of integers:

3

Please input the 1st number:

1

Please input the 2nd number:

2

Please input the 3rd number:

4

average is 2

Waiting for command...

exit

See you!

**5 Deadline**

The deadline for submitting all programs is **13 September 2017, Wednesday, 5pm**. Late submission will NOT be accepted.

*Last updated: 12 September 2017*