CMPE 312 - Operating Systems
Section 01/0101

Homework 01
Deadline: 24.03.2021 23.59

This homework belongs to **CMPE 312.01/0101** Section. If you are not registered this section, please check the questions belongs to your registered section.

RULES

- Every student has to solve the questions belongs to his/her own subsection.
- Codes without documentation (comments) will not be evaluated.
- This work is an individual study. Plagiarism is strictly prohibited, involved students will get zero.

Submission

- Submit your solution as [your_id_cmpe312_hw01_0101].c
- Solution without comments will not be evaluated.
- Late submissions will not be accepted. Submission system will be closed after deadline. Submissions via e-mail will not be accepted.

QUESTIONS

Part I

30 points

1. (10 point) Design a `Stack` structure.
2. (20 point) Define `push` and `pop` functions for your `Stack` implementation.

Part II

70 points

3. (10 point) Create a `Memory` structure owns *number of block*, *block size*, and *stack* attributes. You can add more attributes if it eases your implementations.

   Stack attribute represents memory blocks. *Number of block* attribute represents the length of the *stack*. *Block size* represents the maximum value that each element in the *stack* can get.

4. (40 point) Write an `allocate` function that takes a size parameter. If the given size is bigger than *block size* of the `Memory`, the allocation will be distributed to the different blocks in the *stack* attribute.

   For example, calling `allocate(27)` updates the *stack* as

   $$\texttt{allocate(27)} = [10, 10, 7, 0, 0]$$

   for a `Memory` with *number of block* $= 5$, *block size* $= 10$. The remaining of the elements which don't have maximum value can be sealed until the element is flushed. Therefore, the next allocation can start from next element position after 7 given above.

5. (20 point) Write an `deallocate` function that flushes the last used block.

Note: Ensure that your programs are fully documented, using comments.