cs154: Introduction to Computer Systems
Spring 2020

Homework 6
(Assigned May 11)
Due May 18 (Monday) 11:59pm

Submit your work by committing files in the `hw6` sub-directory of your `CNETID-cs154-spr-20` svn repository. The filename should be either `hw6.txt` or `hw6.pdf` in plain ASCII text, or PDF, respectively. PDFs of scanned hand-written pages must not exceed 6 megabytes. Not following directions will result in losing points.

(1) (20 points)

This question concerns internal and external fragmentation in a simple computer with 10MB of memory, using Dynamic Relocation (Base and Bound). Memory is used only with the granularity of megabytes, so for this question we can consider the physical addresses to be 0, 1, ..., 9. Processes P1, P2, and P3 require only 1MB, 2MB, and 3MB, respectively. However, for a new to-be-run process, a contiguous span of 3MB will be allocated in the lowest possible addresses. 1MB of memory at address 5 is already in use (address ranges 0–4 and 6–9 are free). There is no process migration. Consider the question parts below as a sequence (i.e. part **B** assumes that the OS has dealt with the allocation requested for part **A**). Concisely explain your answers; a simple diagram may help. Write "n/a" if there is no meaningful answer.

A. The user wants to run P1. Is there memory for it, and if so, in what address range will P1 be loaded?

B. The user next wants to run P2. Can P2 be loaded, and if so, in what address range?

C. The user next wants to run P3. Can P3 be loaded, and if so, in what address range?

D. After the OS tries to run P1, P2, and P3, what is the internal fragmentation for each of the processes (P1, P2, and P3), and what is the total internal fragmentation?

E. After the OS tries to run P1, P2, and P3, what is the external fragmentation?

(2) (20 points)

Let's travel back to the past where the computer only provides **8-bit addressing and 2 segments**. The segments are code and stack only. The *code* segment's number is "0" and the *stack*'s segment number is "1".

Let's suppose a process P1 is running and the segmentation table of P1 is as follows. (Do not worry about bounds in this question).

Segment	Base	R	W
Stack (1)	0x13	1	1
Code (0)	0x9A	1	0

A. Please convert the following memory accesses. We have provided you a table that helps you convert logical addresses into physical addresses (";" is just a column divider). Access can be read (r) or write (w). In the last column, please put "S" for success and "E" for error.

For the "offset bits" column, please put the actual bits (not hexadecimal numbers). For example, if you believe the offset width is 6 bits and the content is 0x17, then you need to write six bits "010111".

Access & logical address	Segment bit(s)	Offset bit(s)	Base (in hex)	Offset (in hex)	Physical address (in hex)	Success or Error
r 0xF4 =	___	_____ ;	_____ +	_____ =	_____	(____)
w 0x38 =	___	_____ ;	_____ +	_____ =	_____	(____)
r 0x12 =	___	_____ ;	_____ +	_____ =	_____	(____)

B. Given the **architecture** above, what is the largest possible size of a segment? You can answer in any of the following forms: 2^k bytes or X bytes/KB/MB/GB (where X is a decimal number).