**Assignment 4**

**CIS355** – Spring Term 2018

**Point Value**: 100 points

**Assignment Due Date**: **In class Thursday, April 12, 2018**

**Submission Instruction**

The name of the file should be HW4\_YourLastname\_YourFirstname.docx. Please submit the file .docx on Schoology by 11:59pm and a hard copy of the .docx file to the instructor in class.

**Short answers**

1. What is logical address **(4 points)?** What is Physical address **(4 points)?**

* **A logical address is an address generated by the compiler relative to the start of a process’s address space. A physical address is a real address in memory.**

1. What is internal fragmentation **(5 points)?** What is external fragmentation **(5 points)?** Please use your own words to describe them. You can draw figures/diagrams to illustrate them.

* **Internal fragmentation is the wasted space within each allocated block because of rounding up from the actual requested allocation to the allocation granularity. In other words, it exists when internal memory to a partition is wasted.**
* **External fragmentation exists when there is enough memory to fit a process in memory, but the space is not contiguous. It is the various free spaced holes that are generated in either your memory or disk space.**

1. Compare the memory organization schemes of *contiguous memory allocation*, *pure segmentation*, and *pure paging* with respect to the following issues:
   1. External fragmentation **(6 points)**
   2. Internal fragmentation **(6 points)**
   3. Ability to share code across processes **(6 points)**

**a). External fragmentation is contiguous allocation with the fixed size partitions not suffering from the external fragmentation, but contiguous allocation with variable sized partitions. Pure paging does not suffer from external fragmentation, since  
partitions and pages are fixed in size. Segmentation does suffer from external fragmentation.**

**b) Internal fragmentation does not suffer segmentation and variable-sized partitions, since by definition, a segment/partition is exactly as large as it needs to be. However, the contiguous allocation with fixed size partitions and paging both may suffer from internal fragmentation when partitions and pages are not completely filled.**

**c.) Ability to share code across processes contiguous allocation provides no support for the code sharing. In segmentation, as long as the segments of a process do not mix text and data, we can easily share code between processes.**

1. **Consider the following segment table:**

**Segment Base Length/Limit**

0 219 600

1 2300 14

2 90 100

3 1327 580

4 1952 96

What are the physical addresses for the following logical addresses?

**Segment# offset**

1. 0, 430 **(5 points) 219+430 = 649**
2. 1, 10 **(5 points) 2300 + 10 = 2310**
3. 2, 200 **(5 points) 90 + 200 = 290**
4. 3, 400 **(5 points) 1327 + 400 = 1727**
5. Please explain what is swapping **(5 points)?** What are the advantages and disadvantages of swapping? **(5 points)**

* **Swapping is exchanging data between the hard disk and the RAM.**

### Advantages: No internal fragmentation, may save memory if segments are very small and should not be combined into one page, segment tables: only one entry per actual segment as opposed to one per page in VM, average segment size >> average page size, less overhead.

### Disadvantages: external fragmentation, costly memory management algorithms, segmentation: find free memory area big enough, paging: keep list of free pages, any page is ok, segments of unequal size not suited as well for swapping.

1. Explain why mobile operating systems such as iOS and Android do not support swapping**? (6 points)**

* **First is that these mobile devices typically use flash memory with limited capacity and swapping is avoided because of this space constraint. Second, flash memory can support a limited number of write operations before it becomes less reliable. Lastly, there is typically poor throughput between main memory and flash memory.**

1. Assuming a 512 bytes memory and the page size is 32 bytes, CPU can address 1 byte. What are the page numbers and offsets for the following logical address references:
   1. 24 (**5 points**)

**24/32 =0(page number), frame number = 10, 24%32 = 24(offset)**

* 1. 128 **(5 points)**

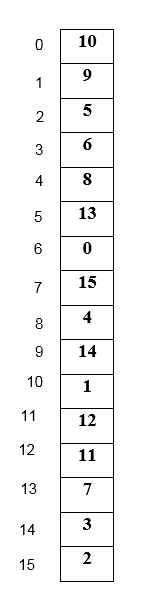
**128/32 =4((page number), frame number = 8, 128%32 = 0(offset)**

* 1. 267 **(5 points)**

**267/32 = 8(page number), frame number = 4, 267%32 = 11(offset)**

* 1. 500 **(5 points)**

**500/32 = 15, frame number = 2, 500%32 = 20(offset)**



1. Assuming a 512 bytes memory and the page size is 32 bytes, CPU can address 1 word (4 bytes):
   1. How many bit for the physical address (**4 points**)

512/32 = 16… 512/4 = 128 bytes

2^x = 128

x = 7 bits

* 1. How many bit for the logical address? (**4 points**)

4 bits(page #) + 3 bits(offset) = 7 bits