**Matthew Austin**

**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)?**
   1. logical address is an address generated by the compiler relative to the start of a process’s address space
   2. a real address in memory
2. 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.
   1. Internal fragmentation is wasted space in each allocated block because of rounding up.
   2. External fragmentation is free space spots that are created in memory or disk space
3. 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)**
      1. *contiguous memory allocation:* Yes, The breaking up of free  
         memory into small chunks via partitioning, which can  
         mean a request for a larger partition later may fail  
         due to lack of contiguous memory.
      2. *pure segmentation:* Yes, Some space is invariably lost between the segments
      3. *pure paging:* No, Does not suffer in external fragmentation: Processes are allocated in page granularity and if a page is not completely utilized, it results in internal fragmentation and a corresponding wastage of space
   2. Internal fragmentation **(6 points)**
      1. contiguous memory allocation: No, paging uses constant-size blocks of memory, and thus minimizes external fragmentation at the expense of internal, if the memory allocated is less than a page
      2. pure segmentation: No, in a pure segmentation system, some space is invariably lost between the segments This is not due to internal fragmentation.
      3. pure paging: Yes, due to pure paging, Processes are allocated in page granularity and if a page is not completely utilized, it results in internal fragmentation and a corresponding wastage of space.
   3. Ability to share code across processes **(6 points)**
      1. contiguous memory allocation: Easy to implement
      2. pure segmentation: Able to share between two different processes.
      3. pure paging: Able to share between two different processes.
4. **Consider the following segment table:**

**Segment Base Length/Limit**

0 219 600

1 2300 14

2 90 100

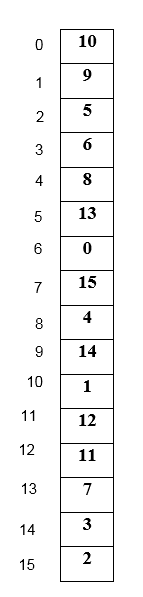
3 1327 580

4 1952 96

What are the physical addresses for the following logical addresses?

**Segment# offset**

1. 0, 430 **(5 points)** 
   1. 430 < 600 thus, 430 + 219 = **649**
2. 1, 10 **(5 points)**
   1. 10 < 14 thus, 10 + 2300 = **2310**
3. 2, 200 **(5 points)** 
   1. 200 <! 100 thus, **Error/Illegal**
4. 3, 400 **(5 points)**
   1. 400 < 580 thus 3 + 1327 = **1330**
5. Please explain what is swapping **(5 points)?** What are the advantages and disadvantages of swapping? **(5 points)** 
   1. Swapping means to roll out a process to disk, releasing all the memory it holds. When process becomes active again, the OS must reload it in memory
   2. The advantages of swapping is:
      1. higher degree of multiprogramming, dynamic relocation.
      2. greater memory utilization.
      3. priority based scheduling.
      4. less wastage of CPU time.
      5. higher performance.
   3. A limitation or disadvantage is that if the power goes off you will lose all the information.
6. Explain why mobile operating systems such as iOS and Android do not support swapping**? (6 points)**
   1. Android does not support swapping and adopts a strategy similar to that used by iOS. It may terminate a process if insufficient free memory is available. However, before terminating a process, Android writes its application state to flash memory so that it can be quickly restarted.
7. 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**)
      1. logical/virtual page number: 24/32 = **0**
      2. offsets: 24 mod 32 = **24**
   2. 128 **(5 points)**
      1. logical/virtual page number: 128/32 = **4**
      2. offsets: 128 mod 32 = **0**
   3. 267 **(5 points)**
      1. logical/virtual page number: 257/32 = **8**
      2. offsets: = 267 mod 32 = **11**
   4. 500 **(5 points)**
      1. logical/virtual page number: 500/32 =**15**
      2. offsets: 500 mod 32 = **20**



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**)
      1. 2^5 + 2^9 = 14 / 2 = **7 bits**
   2. How many bit for the logical address? (**4 points**)
      1. Number of pages \* page size = 16 \* 32 = 9 / 3 = **3 bits**