1. Explain why it is possible for an operating system to load only a couple of pages of a program rather than the entire program?

Because programs are set up in a logical sequence where the first page contains the first instructions of the program while the last page has the last instructions of the program. This gives the OS the ability to sequentially load a program ‘page-by-page’.

1. Describe the purpose of the **Job Table** (JT), **Page Map Table** (PMT) and **Memory Map Table** (MMT).

* **Job Table (JT):** This tables holds two values per active job, size and memory location where the Page Map Table is stored.
* **Page Map Table (PMT):** Stores valuable information about pages.
* **Memory Map Table (MMT):** Stores a single entry for each page frame and shows its location plus free or busy status.

1. Explain how the **Job Table** (JT), **Page Map Table** (PMT), and **Memory Map Table** (MMT) are each used to manage a program stored in memory.

The Operating System relies on all three to implement the algorithm. Page Map Tables are used to determine if a page being requested is already in memory, been modified, if the table has been referenced recently and store the page frame number. The memory map table stores the address location of each byte. Job tables store pages of their own program. When certain parts of a program are called upon, the page map table checks if the page is in memory, then grabs the information from the prospective bytes in the memory map table and loads them.

1. What are some advantages and disadvantages of selecting a page frame size of **1 MB**? List at least one (1) advantage and disadvantage.

Programs would have a much lower chance of thrashing, now that there is ample space for code. However, a major disadvantage I see here is the possibility of a large amount of wasted space should internal fragmentation occur. What if the last portion of the program is literally only 2 bytes large and still needs its’ own page? That is a lot of wasted space.

1. What are some advantages and disadvantages of selecting a page frame size of **100 bytes**? List at least one (1) advantage and disadvantage.

The advantage here is that internal fragmentation won’t waste as much memory, but now thrashing could be taxing on system resources / hardware. Plenty of functions in programs can be quite lengthy and I can imagine how many page swaps would be required to deal with this.

1. Answer the following questions about a program containing **512 bytes** and an operating system using a page frame size of **200 bytes**.
   1. How many page frames are necessary to store the entire program in memory?

512 / 200 = 2 with a remainder (or in this case, displacement) of 112 bytes. Therefore, it would take 3 page frames to store the entire program.

* 1. How many bytes of **external** fragmentation exists when the entire program is stored in memory?

There is no external fragmentation when this program is stored, it is internal.

* 1. How many bytes of **internal** fragmentation exists in page frame containing the first page of the program?

No internal fragmentation exists in the first page.

* 1. How many bytes of **internal** fragmentation exists in page frame containing the last page of the program?

Each page frame is 200 bytes, and minus the left over displacement of 112 bytes: 200 – 112 = 88

* 1. On which page would you find the program instruction at byte **256**?
  2. What is the displacement within the page of the program instruction at byte **256**?

1. Answer the following questions about **thrashing**.
   1. What is thrashing?

Thrashing is when pages are constantly being shifted back and forth between main memory and secondary storage. Portions of a program that rely on each other being separated in different pages can cause this problem if there are little frames available and the OS constantly needs to swap out these pages over and over.

* 1. Explain how thrashing improves or degrades system performance.

Thrashing can dramatically decrease system performance, and even harm system hardware if consistent and extreme enough. Useful computations can be degraded up to a factor of 100.

* 1. Provide at least two (2) causes of thrashing.
* Functions in a program may depend on code within another page that is not yet loaded.
* Too many jobs competing for limited resources.
* Pages be removed and called upon in memory frequently.

1. Refer to this page request sequence A,C,B,D,A,C,E,A,C,B,D,E to answer the following questions.
   1. Perform a page trace analysis using three (3) page frames and the **First-In First-Out** (FIFO) page replacement algorithm.
   2. Perform a page trace analysis using three (3) page frames and the **Least Recently Used** (LRU) page replacement algorithm.
   3. Which page replacement algorithm performed best for this page request sequence?
   4. Will one page replacement algorithm **always** perform better than the other?
   5. Perform a page trace analysis using four (4) page frames and the **First-In First-Out** (FIFO) page replacement algorithm.
   6. Did increasing the number of available page frames increase or decrease the Success Rate of the **First-In First-Out** (FIFO) page replacement algorithm? Explain the cause of this increase or decrease.
   7. What is the fewest number of page faults that could occur using either the First-In First-Out (FIFO) or Least Recently Used (LRU) page replacement algorithms?
2. Answer the following questions about cache memory.
   1. Which type of memory (cache or main memory) is **faster**?
   2. Which type of memory (cache or main memory) is **more expensive**?
   3. Which type of memory (cache or main memory) is **larger**?
3. Answer the following questions about cache memory using a computer system with the following characteristics: **Total Memory Requests**: 5000, **Memory Requests Found in Cache**: 3500, **Average Main Memory Access Time**: 1000 nsec, and **Average Cache Memory Access Time**: 200 nsec.
   1. What is the **Hit Ratio** for the system?
   2. What is the **Average Memory Access Time** for the system?
   3. Explain why is the **Average Cache Memory Access Time** is a constant value in the **Average Memory Access Time**formula.
   4. Explain why is the **Average Main Memory Access Time** is dependent on the **Hit Ratio** value in the **Average Memory Access Time**formula.
   5. What affect, if any, would doubling the available **cache** memory have on the **Average Memory Access Time**?