Problem 1 (Paging: Memory structure and size) 13 pts

1- Explain the functionality of page table pointer and page table length. (2 Pts)

2- Explain how the TLB works. Explain what happens in case of a hit/miss. Is a miss equal a page fault? (5 Pts)

1- Keeping the page table in main memory could be very cumbersome. Explain three strategies to optimize the storage of the page table in memory. (Optimization goals are storage size and access time) (6 Pts)

Problem 2 (Paging: Memory structure and size) 15 pts

A certain computer provides its users with a virtual memory space of $2^{32}$. The computer has $2^{18}$ bytes of physical memory. Virtual memory is implemented by paging, and the page size is 4096 bytes.

Assuming that each page table entry includes a valid/invalid bit an 2 bits for access right management

1. Show how the physical address and the virtual address are partitioned. Which part is used to address the page, the frame and the data? (6 Pts)

2. How many entries are in the page table? (3 Pts)

A user process generates the virtual address 0xA9BB00. Explain how the system establishes the corresponding physical location. Distinguish between software and hardware operations. (5 Pts)

1- Assume the computer uses a software-loaded TLB

2- Assume a two-level page table structure

Problem 3 (Demand paging) 6 pts

Assume we have a demand-paged memory. The page table is held in registers. It takes 8 milliseconds to service a page fault if an empty page is available or the replaced page is not modified, and 20 milliseconds if the replaced page is modified. Memory access time is 100 nanoseconds.

Assume that the page to be replaced is modified 70 percent of the time. What is the maximum acceptable page-fault rate for an effective access time of no more than 200 nanoseconds?
**Problem 4 (Page replacement) 16 pts**

A page-replacement algorithm should minimize the number of page faults. We can achieve this minimization by distributing heavily used pages evenly over all of memory, rather than having them compete for a small number of page frames. We can associate with each page frame a counter of the number of pages associated with that frame. Then, to replace a page, we can search for the page frame with the smallest counter.

1- Define a page-replacement algorithm using this basic idea. Specifically address these problems: *(6 Pts)*

   a. What the initial value of the counters is
   b. When counters are increased
   c. When counters are decreased
   d. How the page to be replaced is selected

2- How many page faults occur for your algorithm for the following reference string, for four page frames? *(5 Pts)*

   1, 2, 3, 4, 5, 3, 4, 1, 6, 7, 8, 7, 8, 9, 7, 8, 9, 5, 4, 5, 4, 2.

3- What is the minimum number of page faults for an optimal page replacement strategy for the reference string in part b with four page frames? *(5 Pts)*

**Problem 5 (Working set) 8 pts**

Consider the parameter \( \Delta \) used to define the working-set window in the working-set model. What is the effect of setting \( \Delta \) to a small value on the page fault frequency and the number of active (non-suspended) processes currently executing in the system? What is the effect when \( \Delta \) is set to a very high value?