CSE231 Operating System, Quiz 03 Monsoon 2024
Time allocated: 4pm – 4:20pm

Name	
Roll Number	

Instructions:

- This **is a closed book and closed notes quiz. Please be aware of strict plagiarism policy.**
- For questions requiring justification, please be as concise as possible. 2-3 sentences would be the ideal size of a justification. No extra pages will be provided.

Question 1: What is the actual name of the address that programmers see on an x86 processor? What specific information does this address contain that is necessary for address translation? **[2 marks]**
Answer:

Logical address [+1 marks]
Segment selector [+1 marks]

Question 2: Paging may result in internal fragmentation. True/False. **Justify** using an example. **[2 marks]**
Answer:

True. See Lecture 17, slide 6.
No partial marks. Marks only if correct example (may not be same as in slide).

Question 3: Which of the two segment registers on x86 processor hold the **most** important information related to the program execution? **Justify**, and provide their full name. **[2 marks]**.
Answer:

Code segment (CS) as it contains information related to .text section. [+1 marks]
Stack segment (SS) as it contains information related to the execution stack of the process. [+1 marks]

Question 4: A machine has a 16-bit address space that programmers can use. Each CPU has a dedicated cache of 4KB to store the page table of a running process. The size of an integer or pointer on this machine is 4 bytes. To maximize the number of entries in the page table stored in the cache, how many bits in a 16-bit address should be used to identify the pages uniquely? **Justify**. **[2 marks]**
Answer:
Maximum size of page table to ensure cache can entirely store the page table = 4KB. [+0.75 marks]
Assuming an array-based implementation of page table, each entry will be of 4b. Hence, total number of page table entries that could fit in the cache = 4KB / 4 = 2^10. [+0.75 marks]
Total bits required to uniquely identify these pages = 10. [+0.5 marks]

Question 5: Calculate the total number of page faults that must happen while accessing the heap space as shown in the program. Assume size of integer as 4 bytes and the page size to be 64 bytes. Justify. **[2 marks].**
Answer:

```
int main( ) {
  int* a = (int*) malloc(1024);
  for(int i=0;i<128; i++)
     a[i] = 0;
  return 0;
}
```

Page fault will happen on the pages being accessed. [+1 marks]
Hence, total page faults for accessing 128 integers = (128 * 4) / **64** = **8**
[+1 marks]

Page table size = 2^12
4N = 2^12
N = 1^10