This homework is due in class on Monday, November 21. However, since I won't be grading it until Sunday, it can also be emailed to me no later than noon on the following Sunday, November 27. Some of the questions do not have definite, unambiguous answers. In all cases, justify and explain your answers! As usual, you can work together on the problems, but everyone should write up their own solutions, in their own words.

1. A computer with a 32-bit virtual address space uses a two-level page table. Virtual addresses are split into an 8-bit first level page table and an 11-bit second level page table. **(a)** What is the page size, and how many pages are there in a virtual address space? **(b)** How many entries are there in the first level page table, and how many entries are there in a second level page table? **(c)** Describe how to find the physical address corresponding the following virtual address:

```
0010 1110 1001 1101 0001 0100 0111 1010
```

2. (*Based on Chapter 8, Exercise 10.*) Consider a computer architecture with multi-level paging, a page size of 4KB, and 64-bit physical and virtual address spaces. Assume that a page table must fit into a single page of memory (that is, its size is 4KB or less). In that case, the location of a page table can be given as a page number. Explain why the size of a single page table entry, rounded up to the nearest power of two, must be at least 64 bits. Given that a page table entry occupies 64 bits, what is the maximum number of page table entries in a page? Given the requirement that each page table fits into a page, how many levels would be required in the multi-level paging to completely map the 64-bit address space?

3. (*Based on Chapter 8, Exercise 4.*) Discuss the advantages of a memory architecture that uses both segmentation and paging over one that uses only segmentation. Then discuss the advantages of a memory architecture that uses both segmentation and paging over one that uses only paging.

4. Memory caches are often designed to load blocks of data from memory rather than the data from single addresses. For example, a level 3 cache might load data in 64-byte blocks. And swapping loads data in page-sized chunks.

(a) Explain what this design decision has to do with spatial locality.

(b) Now, consider the following two code segments for adding up the values in a very large 2D array of *double*. (Assume that it's a real 2D array stored in a single block of memory in row-major order; that is, the first row, followed by the second row, followed by the third row, etc.).

```
double sum = 0.0;
for (i = 0; i < 4096; i++)
  for (j = 0; j < 4096; j++)
    sum += A[i][j]
```

```
double sum = 0.0;
for (j = 0; j < 4096; j++)
  for (i = 0; i < 4096; i++)
    sum += A[i][j]
```

Assume that the code segments are run on a computer with a 128KB cache that loads data in 64-byte blocks. Explain why the two code segments might have very different performance on that computer.

(c) For the code segment with the poorer performance, which cache replacement algorithm would give better performance, RANDOM or LRU? (Why?) How, approximately, would the two algorithms compare in this case? (Why?)