Problem Set 4
Due Sunday February 26, 11:59pm

1.Suppose in xv6 user code, I call a procedure foo, with a large automatic variable. What happens and why? What if the size of the automatic is much larger? What if the code is in the xv6 kernel? What happens and why?

```
void foo() {
  int big[2000];

  big[1500] = 0xBEEF;
}
```

2. Improve your mmap code to allow multiple processes to map the same file at the same time, and to work when a process forks (both child and parent have access to the mmap'ed region. Hint: memory for a mapped file should only be reclaimed when all processes have unmapped it.

3. Add copy on write for processes sharing memory. The UNIX fork/exec mechanism seems inefficient – first, make a copy of the address space, and then throw that copy away during exec. An efficient UNIX fork is possible, however, by supporting copy on write (see sidebar in OSPP p. 381). The fork makes a copy of the address space page table, marking every page as read-only in both the parent and child, and then resumes execution. When either the parent or the child modifies the page, a page fault occurs and the kernel can fill in the missing page.