Problem 3 -- Smear program

We're going to take advantage of the `mmap` system call to do something which is kind of awkward with the traditional read/write file I/O interface. I'll call the program `smear` and it will be invoked like this:

```
smear TARGET REPLACEMENT file1 {file 2....}
```

For each of the named files, search for every instance of the string given by the first argument and replace it with the replacement string given by the second argument. **Note:** to simplify the problem, both target and replacement strings must be the same number of characters.

Do this with `mmap`. Again, to simplify the problem, you aren't required to handle cases where the file is so large that it can't be mapped in its entirety into virtual address space (but still remember to check for and properly report any applicable system call errors)

Submit source code, and a screen shot with a small demonstration file showing that the intended search and replace took effect.