Names:

Kyle Thakker

Steven Flynn

Nandan Thakkar

I-labs used in testing:

Man.cs.rutgers.edu

Kill.cs.rutgers.edu

Design.cs.rutgers.edu

Prototype.cs.rutgers.edu

Summary:

We designed a simple memory management named memory.c. We were able to successfully implement a memory manager with only a few limitations (see below). We started by dividing the memory space into an OS region and user region. The user region contains the actual memory pages that are allocated to the user. The OS region is further divided into two separate regions. One region contains all the memory used by the pthread library. The other contains page nodes which are the metadata keeping track of the user pages. Each page node keeps track of the thread, largest free memory, page ID, and offset. The thread is simply the thread number, the largest free memory tracks the size of the largest free block in that page. Similarly, page ID is basically an index of all pages assigned to a certain thread; while offset is the page's offset in memory or the swap file. If offset is negative the memory associated with that page is in the swap file.

Inside a user page, next to each memory block allocated upon request, is a metadata struct called a mem node. These mem nodes contain a char that indicates whether the block is free or not. They also contain the size of the memory block that the mem Node manages.

There are three pages that are used to move pages to temporarily when pages need to be swapped around. There are also 4 pages at the end of the memory array that are for memory that needs to be shared between threads. This memory can be requested with the shalloc function.

Limitations:

- We only allowed the support of about 32 different threads at a time. Anything more than that and we do not guarantee the software will work correctly