The interface for the ready queue is this:

readyqueue();

~readyqueue();

void push(process \* p); // method to add a process to the ready queue

process \* pop(void); // method to get a process from the ready queue and delete it

unsigned int size(void); // method to get the current size of the ready queue

bool empty(void); // method to check if the queue is empty

The class is implemented as a monitor so that only one of its methods can be run at any time. This is achieved by using a pthread mutex with a recursive attribute. The recursive attribute allows one thread to lock the mutex more than once but requires it to unlock it the same number of times before other threads may lock it.

If a process calls pop() when the queue is empty, it blocks on a condition. That condition is signaled by the push(process \*) method.

The blocked queue is likewise a monitor. Its interface is this:

void block(process\* a); // method to block a process

process \* IOFinish(int position); // unblock a process

unsigned int size(); // size of the queue

blockedqueue();

~blockedqueue();

It uses a simple pthread mutex with the default attribute and no conditions.