Yang Wang 321004893

Design Report

1 Scheduler

I use a queue structure for implementing the FIFO scheduler. The queue is a singly linked list with a head and tail. For the yield() function, I just dequeue a thread, which is the head. And for add() function, I just enqueue a thread into the data structure.

My data structure looks like the following:

```
struct myQueue{
    int head;
    int tail;
    Thread** Array;
};
```

1.1 The constructor of Scheduler will initialize a queue.

1.2 void Scheduler::yield(): Will dequeue a thread from the QueueRecord.

1.3 void Scheduler::add and void Scheduler::resume are basically the same the only different is that resume() will add the current thread back to the end of queue.

1.4 void Scheduler::terminate will just release the frame, which is allocated to this thread.

2 Thread Class

2.1 static void thread_start(): the interruption is enable by adding code.
2.2 static void thread_shutdown(): the Scheduler::terminate() is called