Yang Wang 321004893

Machine Problem 5

1. Blocking Disk Class:

Blocking disk class derives read and write operation from simple disk. Furthermore it also maintains a FIFO queue for ready threads.

```
class BlockingDisk:public SimpleDisk{
public:
   BlockingDisk(DISK_ID _disk_id, unsigned int _size);
   void read(unsigned long _block_no,unsigned char* _buf);
   void write(unsigned long _block_no, unsigned char* _buf);
   Thread* get_thread();
private:
   queue* blockq;
};
```

The yield() function in scheduler will check the device queue first to see if the disk is ready. If yes, the scheduler will dequeue a thread from the blocking disk and dispatch the CPU to it so the I/O operations can be issued. Otherwise, the scheduler will dequeue a thread from the ready_queue and dispatch the CPU to the ready thread.

2. File Class

The File Class is defined as following:

```
class File {
   friend class FileSystem;
private:
    static FileSystem   * file_system;
    unsigned int   file_id;
    unsigned int position;
   unsigned int size;
   unsigned int start;
   unsigned int currentpos;
   unsigned int currentblock;
public:
   File(unsigned int _fid);
   unsigned int Read(unsigned int _n, char * _buf);
   unsigned int Write(unsigned int _n, char * _buf);
   void Reset();
   void Rewrite();
   BOOLEAN EoF();

};
```

The file will record the file id, its size, start point and current postion and current block. The files are maintained by linked list of blocks. Each block is 512 bytes and with 4 bytes pointer information. So the actual size of each block is 508 bytes.

Yang Wang 321004893

File::File():
Every file has a unique id, the constructor will first check if the file is already exist. If not, the file system will call CreateFile() to create a new file.

Read(unsigned int, char* buf) will read minimum number of characters among n, current position and 508-current position. The size of n will decrease during each iteration.

Write():
Similarly, it will write data into the file per iteration. It will check whether the current position reaches the end of file. If the EoF return true, the write function will begin to write data to next available block.

EoF(): Check if the current position is same to the pre-defined size.

Reset() will just set the current position and current block to the beginning.

Rewrite() will first do the same thing as reset, but then link the block to the free block list, which file_system class maintains. At last, the file_system will update the file data and store the size.

3 File System Class:

```
class FileSystem {

friend class File;

private:
     SimpleDisk * disk;
     unsigned int size;
    unsigned int freeblocks;
    unsigned int numberof_file;
    unsigned int file_data[512];
    unsigned int file_size[512];
    unsigned int bloksize=512;

public:

   FileSystem();
   BOOLEAN Mount(SimpleDisk * _disk);
   static BOOLEAN Format(SimpleDisk * _disk, unsigned int _size);
   BOOLEAN LookupFile(int _file_id, File * _file);
   BOOLEAN CreateFile(int _file_id);
   BOOLEAN DeleteFile(int _file_id);

};
```

The file system maintain the file system management information. The first block is reserved for these information.

Format(): function will go through each block and link all blocks.

Yang Wang 321004893

Mount() function will first point to the mounted disk and then copy all the information from block 0 into this new block.

LooupFile() function will get the file from the list, which has the same file id. And initialize the file object and return TRUE. Otherwise return FALSE.

DeleteFile() function will first find the file being deleted by loopup the given file id. And then move the block to the free block list. Then refresh the file found with the give file id and set to 0.