# Midterm #2 for Operating Systems (2020 Fall, Grad Level)
# Question Sheet

1. [U8][12 pts] Consider the following memory workloads:
   a) W1: accessing a small, fixed set of pages repeatedly.
   b) W2: accessing a small set of pages repeatedly. The accessed page set changes from time to time.
   c) W3: Scanning a large set of seqnetial pages.
   d) W4: Accessing all pages randomly.

   Choose a good page replacement algorihtm among LRU, LFU, and Random for each of the following scenarios and explain the rationale of your choices:
   a) W1 + W3
   b) W2
   c) W4
2. [U9][8 pts] You are creating a new file "bar" under an existing directory "foo". However, the new file is not persistent on the completion of the fsync() call (i.e., the file might be lost after a power fail right after the fsync()). Why is that? How to fix this problem?

   ```
   fd=open("/foo/bar", O_CREAT….);
   write(fd,…);
   fsync(fd);
   close(fd);
   ```
3. [U10][12 pts]Modern file systems adopt extents for space allocation. What is an extent? How extent allocation is better than the linked-list allocation method, and what is its potential drawback?
4. [U11][8 pts] Write-ahead logging is a common technique for file system journaling. Answer the following questions based on the desgin of Ext3:
   a) Assume Ext3 operates in *journal* mode. Describe how dirty (modified) user data and metadata are written to the final disk locations in the file system image. Your answer must include the following items: transations, the main memory, the on-disk logging space (jornal), and the file system image.
   b) Consider ***all*** the possible time points of power failure in the proceure your describe above. How a journaling file system recovers from each of the failure time points?
5. [MM1][8 pts] Suppose that you are deciding the page size (large page size 4MB or regular page size 4KB) for programs. What will be the good choice for the

following programs? Why?

a) Bubble sorting on a very large array.

b) Finding a small number of random keys in a very large binary search tree.

6. [MM2][8 pts] Why LRU is vulnerable to seqneital scan? How ARC deals with this problem?

7. [FS1][12 pts] Explain how the following design options from the UNIX Fast File System improves I/O efficiency:

   a) Dividing the disk space into cylinder groups.

   b) Using large allocation units.

   c) Having a few direct pointers (to data blocks) in an inode.

8. [FS2][6 pts] Ext3 is based on write-ahead logging for file system journaling. To avoid the double writing problem of write-ahead logging, in the *ordered* mode, Ext3 journals only file system metadata but not user data. However, when writing a large set of random files, Ext3 in the *ordered* mode still risks poor performance. Explain why.

9. [FS3][8 pts] An important conclusion drew from the NTFS file usage paper is that many files are small and small files have extremely short lifetimes. Based on this conclusion, which one(s) if the following options wil you take when designing a new file system? Why?

   a) Employing byte-addressible, nonvolatile memory to store small files and file system metadata.

   b) Using disk arrays to parallelizing large file writes.

10. [AFS1][6 pts] In log-structured file systems, how the separation of hot data (frequently updated data) from cold data (infrequently updated data) benefits the efficiency of segment cleaning? Explain your answer using an example.

11. [AFS2] [6 pts] Btrfs is based on Copy-On-Write (COW) B-trees. Let a power failure occurs in the middle of a set of disk writes. Show how COW B-trees recover from the power failure.

12. [AFS3] [6 pts] A problem of soft updating is that a set of dirty disk blocks cannot be written to the disk if there is a circular dependency among them. Propose a method to detect such a dependency among dirty blocks.

*Write down your answer in English.

*Provide sufficient details in your answers.