---- TEAM ----

>> Team name.

Team 14

>> Fill in the names, email addresses and contributions of your team members.

Taeyoon Kim tykimseoul@kaist.ac.kr 50

Sungmin Choi smc9601@kaist.ac.kr 50

contribution1 + contribution2 = 100

>> Specify how many tokens your team will use.

INDEXED AND EXTENSIBLE FILES

==========

---- DATA STRUCTURES ----

A1: Copy here the declaration of each new or changed `struct' or `struct' member, global or static variable, `typedef', or enumeration. Identify the purpose of each in 25 words or less.

A2: What is the maximum size of a file supported by your inode structure? Show your work.

---- SYNCHRONIZATION ----

A3: Explain how your code avoids a race if two processes attempt to extend a file at the same time.

A4: Suppose processes A and B both have file F open, both positioned at end-of-file. If A reads and B writes F at the same time, A may read all, part, or none of what B writes. However, A may not read data other than what B writes, e.g. if B writes nonzero data, A is not allowed to see all zeros. Explain how your code avoids this race.

A5: Explain how your synchronization design provides "fairness". File access is "fair" if readers cannot indefinitely block writers or vice versa. That is, many processes reading from a file cannot prevent forever another process from writing the file, and many processes writing to a file cannot prevent another process forever from reading the file.

---- RATIONALE ----

A6: Is your inode structure a multilevel index? If so, why did you choose this particular combination of direct, indirect, and doubly indirect blocks? If not, why did you choose an alternative inode structure, and what advantages and disadvantages does your structure have, compared to a multilevel index?

SUBDIRECTORIES

==========

---- DATA STRUCTURES ----

B1: Copy here the declaration of each new or changed `struct' or `struct' member, global or static variable, `typedef', or enumeration. Identify the purpose of each in 25 words or less.

---- ALGORITHMS ----

B2: Describe your code for traversing a user-specified path. How do traversals of absolute and relative paths differ?

---- SYNCHRONIZATION ----

B4: How do you prevent races on directory entries? For example, only one of two simultaneous attempts to remove a single file should succeed, as should only one of two simultaneous attempts to create a file with the same name, and so on.

B5: Does your implementation allow a directory to be removed if it is open by a process or if it is in use as a process's current working directory? If so, what happens to that process's future file system operations? If not, how do you prevent it?

---- RATIONALE ----

B6: Explain why you chose to represent the current directory of a process the way you did.

BUFFER CACHE

==========

---- DATA STRUCTURES ----

C1: Copy here the declaration of each new or changed `struct' or `struct' member, global or static variable, `typedef', or enumeration. Identify the purpose of each in 25 words or less.

---- ALGORITHMS ----

C2: Describe how your cache replacement algorithm chooses a cache block to evict.

C3: Describe your implementation of write-behind.

C4: Describe your implementation of read-ahead.

---- SYNCHRONIZATION ----

C5: When one process is actively reading or writing data in a buffer cache block, how are other processes prevented from evicting that block?

C6: During the eviction of a block from the cache, how are other processes prevented from attempting to access the block?

---- RATIONALE ----

C7: Describe a file workload likely to benefit from buffer caching, and workloads likely to benefit from read-ahead and write-behind.

SURVEY QUESTIONS

==========

Answering these questions is optional, but it will help us improve the course in future quarters. Feel free to tell us anything you want--these questions are just to spur your thoughts. You may also choose to respond anonymously in the course evaluations at the end of the quarter.

In your opinion, was this assignment, or any one of the three problems in it, too easy or too hard? Did it take too long or too little time?

Did you find that working on a particular part of the assignment gave you greater insight into some aspect of OS design?

Is there some particular fact or hint we should give students in future quarters to help them solve the problems? Conversely, did you find any of our guidance to be misleading?

Do you have any suggestions for the TAs to more effectively assist students in future quarters?

Any other comments?