The Word Thief

The **Word Thief** is at it again! This time, the Word Thief has gone through the ancient texts of operating systems and removed some key words, thus potentially confusing many generations of students! What a tricky one that Word Thief is.

No one has been able to capture the Word Thief in a photo. This picture is of the Hamburglar. He steals hamburgers, or at least used to, before he was tried and convicted.

Your job is simple: to fill in words that the Word Thief has removed, thus restoring to all the ability to learn. Future generations will thank you for your work on this important text! Future generations may also curse you, because they still have to study Operating Systems due to your diligence.

Strangely, the Word Thief focused on the Persistence chapters of OSTEP. Who knows what drives the motivations of this curious character.

Good _____!

This exam has **16** pages and **60** questions. Each question has **exactly one** answer.

Disks

The Word Thief really did a number on that ***Hard Drives*** *chapter. There is so much missing material! Can you help fill in the blanks from this critical chapter?*

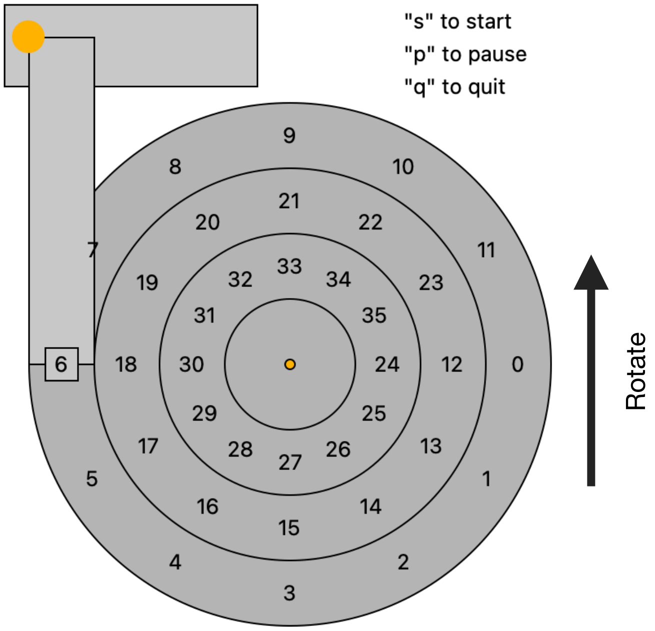
1. “The drive consists of a large number of sectors (512-byte blocks), each of which can be read or ______.”
(a) deleted
(b) [✓ written]
(c) read from
(d) erased
(e) truncated

2. “We start with a _____, a circular hard surface on which data is stored persistently by inducing magnetic changes to it.”
(a) [✓ platter]
(b) surface
(c) track
(d) sector
(e) block

3. “Data is encoded on each surface in concentric circles of sectors; we call one such concentric circle a ______.”
(a) platter
(b) surface
(c) [✓ track]
(d) sector
(e) block

4. “Another reality is that outer tracks tend to have more sectors than inner tracks, which is a result of geometry; there is simply more room out there. These disks are often referred to as ______ disk drives, …”
(a) wicked smart
(b) extended
(c) regular
[✓ (d) multi-zoned]
(e) shingled

5. *In the next four (5-8) questions, the Word Thief has hacked the **disk simulator**! What a clever hacker, this Word Thief is. What's missing? Well, unsurprisingly, it's the solutions!*



Given the disk to the right (which rotates counter-clockwise), and assuming **FIFO** disk scheduling, what request would be serviced **last**, assuming the requests are: **13, 4, 35, 34, 5**

(a) 13
(b) 4
(c) 35
(d) 34
(e) 5

13、4、35、34、5

6. Now assume a **SSTF** (shortest seek first) disk scheduler, and a VERY FAST (essentially infinitely fast) seek; which request would be serviced **last**? (assume that when you get to a particular track, the scheduler will read the blocks in the fastest order available on that track)

a) 13
(b) 4
(c) 35
(d) 34
(e) 5

4、5、13、34、35

7. Now assume a **SATF** (shortest access first) scheduler, and a VERY FAST (essentially infinitely fast) seek; which request will be serviced **last**?

a) 13
(b) 4
(c) 35
(d) 34
(e) 5

34、35、13、4、5

8. Assuming a VERY FAST seek (essentially infinitely fast), how would you order the disk schedulers for those five requests, from fastest to slowest?

(a) SSTF, SATF, FIFO
(b) SSTF, FIFO, SATF
(c) SATF, SSTF, FIFO
(d) SATF, FIFO, SSTF
(e) FIFO, SATF, SSTF

9. *Now, the Word Thief is back to the usual thieving: removing key words or phrases.*

“The three basic components of disk I/O time are ______, _______, and ______.”
(a) transition, position, constitution
(b) move head, wait for head to stop moving, transfer
(c) position, transfer, react
(d) shake, rattle, roll
(e) seek, rotate, transfer

10. “A disk scheduling algorithm that avoids starvation is called ______”
(a) PASS
(b) CRANK
(c) SCAN
(d) CHECK
(e) FAIRPLAY

RAID

*The Word Thief hates redundant arrays of independent drives (**RAID**)! Why, you ask? Simple: Redundancy makes the Word Thief work harder to remove words from the world.*

11. “RAIDs offer a number of advantages over a single disk. One advantage is performance. Using multiple disks in parallel can greatly speed up I/O times. Another benefit is capacity. Large data sets demand large disks. Finally, RAIDs can improve ______”
(a) size
(b) the odds
(c) reliability
(d) latency
(e) distribution

12. “More generally, a mirrored system (with mirroring level of 2) can tolerate 1 disk failure for certain, and up to ______ failures depending on which disks fail.” **[Assume here there are N disks in the system]**
(a) 2
(b) N/4
(c) N/2
(d) N/3
(e) N

13. “Thus, we can conclude that the maximum bandwidth obtained during sequential writing to a 2-way mirrored array is ______” **[Assume here there are N disks, and that a single disk delivers S MB/s of disk bandwidth]**

(a) S MB/s
(b) 2 x S MB/s
(c) N x S MB/s
(d) N x S / 2 MB/s
(e) N x S x S MB/s

14. “For example, in RAID-4, if we had blocks of size 2 bits (yes, this is still quite a bit smaller than a 4KB block, but you get the picture), they might look something like this:

Block0	Block1	Block2	Block3	Parity”
00	10	11	10	_____

(a) 00
(b) 01
(c) 10
(d) 11
(e) None of the above

15. “RAID-4 uses a disk for parity information for every group of disks it is protecting. Thus, our useful capacity for a RAID group is ______.” **[Assume N disks, and B bytes of data per disk]**

(a) N x B
(b) N
(c) B
(d) N x (B - 1)
(e) (N - 1) x B

16. “Random read performance of RAID-5 (as compared to RAID-4) is ______”

(a) a little better
(b) a little worse
(c) a lot better
(d) a lot worse
(e) about the same

17. “RAID Level(s) _____ encounter(s) the ‘small write’ problem.”

(a) 0
(b) 1
(c) 4
(d) 5
(e) 4 and 5

18. “A single write (to a RAID-4) requires _____ read(s) and then ______ write(s) to the underlying disks.” **[assuming subtractive parity]**
(a) 2, 1
(b) 1, 2
(c) 2, 2
(d) 1, 1
(e) None of the above

19. “Assuming that each disk, under a random write workload, delivers R MB/s, a RAID-5 system with N disks will deliver ______ MB/s under a random write workload.”
(a) N x R MB/s
(b) N x R / 2 MB/s
(c) N x R / 4 MB/s
(d) R / 2 MB/s
(e) R MB/s

20. “To conclude, if you strictly want performance and do not care about reliability, ______ is obviously best.”
(a) rebooting
(b) a parity-based approach
(c) mirroring
(d) thinking
(e) striping

Flash-based SSDs

The Word Thief hates fast storage! Thus, the Thief has worked particularly hard to remove key words from our educational materials about ***Flash-based SSDs.*** *I hope you can figure them out, otherwise we’ll be stuck using hard drives forever!*

21. “A Flash bank is organized into a large number of ______, each of which is further sub-divided into pages.”
(a) mega pages
(b) blocks
(c) units
(d) chunks
(e) candy bars

22. “A typical size of a Flash page is ____.”

[✓ (a) 4 KB]

(b) 256 KB

(c) 256 MB

(d) 256 GB

(e) over 1 TB

23. “Once a Flash page is programmed, it _____.”

(a) can be re-programmed repeatedly (without intervening steps)

(b) is guaranteed to store bits within it, permanently

(c) can never be re-programmed

(d) can be re-programmed, but first must be read

[✓ (e) cannot be re-programmed until the entire block is erased]

24. “The biggest reliability problem Flash chips have is ______.”

(a) head crashes

(b) read/write disturbance

(c) cracking

[✓ (d) wear out]

(e) burn out

25. “The process of _______ ensures that dead pages can be reclaimed for subsequent writes.”

(a) wear leveling

(b) read programming

[✓ (c) garbage collection]

(d) input reduction

(e) write amplification

The Write Thief also hacked the OSTEP SSD simulator! What a sneak. Now, the simulator does some weird stuff. Like, for example, not give any answers! Can you help fix this problem?

The initial state of the Flash SSD in the simulator is represented like this:

```
Block 0          1          2          3          4          5          6
Page  0000000000 1111111111 2222222222 3333333333 4444444444 5555555555 6666666666
      0123456789 0123456789 0123456789 0123456789 0123456789 0123456789 0123456789
State iiiiiiiiii iiiiiiiiii iiiiiiiiii iiiiiiiiii iiiiiiiiii iiiiiiiiii iiiiiiiiii
Data
```

This shows the first seven blocks (with ten pages per block) of a Flash-based SSD, with each block in an INVALID state (marked ‘i’).

After five writes, the state of the Flash SSD looks like this, with erased pages marked ‘E’, and VALID (programmed) pages marked ‘v’, and contents shown along the ‘Data’ line:

```
Block 0          1          2          3          4          5          6
Page  0000000000 1111111111 2222222222 3333333333 4444444444 5555555555 6666666666
      0123456789 0123456789 0123456789 0123456789 0123456789 0123456789 0123456789
State vvvvvEEEEE iiiiiiiiii iiiiiiiiii iiiiiiiiii iiiiiiiiii iiiiiiiiii iiiiiiiiii
Data  qiUKz
Live  +++++
```

26. If erases take 1000 microseconds, and page programming takes 40 microseconds, how long did the entire sequence of five writes take to complete?
(a) 1000 microseconds
(b) 1100 microseconds
(c) 1200 microseconds
(d) 40000 microseconds
(e) 1 millisecond

27. Given the state and performance assumptions of the system above, and assuming a log-structured FTL, how long will the next write take to complete?
(a) 10 microseconds
(b) 40 microseconds
(c) 1000 microseconds
(d) 1040 microseconds
(e) 2 milliseconds

28. After the five writes above took place, assume the FTL has the following contents:

```
12->4   14->3   29->2   37->0   39->1
```

What data value will be returned if the user issues a **read** to block 29?

(a) q

(b) i

(c) U

(d) K

(e) z

29. After the first five writes (as shown above), assume the next five writes are to blocks 12, 20, 30, 39, and 50. After these writes, how many pages in the SSD will be *live*?

(a) 7

(b) 8

(c) 9

(d) 10

(e) 11

30. Assuming the same times (erase of a block takes 1000 microseconds, program of a page takes 40 microseconds), what is the average cost per write for the first 10 writes?

(a) 100 microseconds

(b) 120 microseconds

(c) 140 microseconds

(d) 200 microseconds

(e) 1040 microseconds

File System Implementation

No one likes ***implementing file systems*** *(like the very simple file system, VSFS) more than systems students. And no one likes removing key words from educational materials than the Word Thief. These questions are all about that very simple file system…*

31. "The ______ is the generic name that is used in many file systems to describe the structure that holds the metadata for a given file, such as its length, permissions, and the location of its constituent blocks."

(a) superblock

(b) inode

(c) data block

(d) directory

(e) journal

32. “Thus, an inode has a fixed number of direct pointers (12), and a single indirect pointer. If a file grows large enough, an indirect block is allocated (from the data-block region of the disk), and the inode’s slot for an indirect pointer is set to point to it. Assuming each slot can point to a 4-KB block, and that disk addresses are 4 bytes, the file can grow to be ______.

(a) 4096 KB
(b) 4100 KB
(c) 4104 KB
(d) 4044 KB
(e) 4144 KB

33. “Let’s examine an example with twelve direct pointers, as well as both a single and a double indirect block. Assuming a block size of 4 KB, and 4-byte disk addresses, such a file system can have a max file size of ______ (approximately).”

(a) ~4 KB
(b) ~1 MB
(c) ~4 MB
(d) ~1 GB
(e) ~4 GB

34. “In VSFS (as in many file systems), directories have a simple organization; a directory basically just contains a list of (______, ______) pairs.”

(a) directory name, file attribute
(b) file name, inode number
(c) file name, parent location
(d) inode number, file type
(e) inode type, file directory

35. “… free space management is important for all file systems. In VSFS, we have two simple ______ for this task.”

(a) free lists
(b) management teams
(c) hamburgers
(d) bitmaps
(e) inodes

36. “In this example, let us first assume that you want to simply open a file **`/foo/bar`**, read it, and then close it. In doing so, the file system will read ______ inodes.”

(a) 0
(b) 1
(c) 2
(d) 3
(e) 4

37. “Once **`/foo/bar`** is opened, assume a process appends a data block to it three times. The following blocks will be written to during each append: ______, ______, and ______. **[assume no buffering of writes]**

(a) the file’s inode, data bitmap, and the data block itself
(b) the directory data block, the data block, and the superblock
(c) the inode bitmap, the directory data block, and the inode
(d) the journal, the data block itself, and the directory
(e) the department, the chair, the entire university

38. "Write buffering (as it is sometimes called) has a number of performance benefits. They are ______, ______, and ______.

(a) skipping, tricking, and faking out
(b) batching, scheduling, and avoiding writes altogether
(c) batching, scheduling, and smoothing writes out
(d) batching, avoiding writes, and spacing writes out over time
(e) anticipating, logging, and batch amortizing

39. *The Word Thief even hacked the VSFS homework! Now it won’t provide solutions. Fill in the missing solution below. Remember an inode in the simulator contains the following three things:* `[filetype address-of-data reference-count]`

```
Initial State:
inode bitmap  10000000
inodes        [d a:0 r:2][][][][][][][]
data bitmap   10000000
data          [(.,0) (..,0)][][][][][][][]

______ // this is the operation that took place, what is it?

Final State:
inode bitmap 11000000
inodes        [d a:0 r:2][f a:-1 r:1][][][][][][]
data bitmap   10000000
data          [(.,0) (..,0) (m,1)][][][][][][][]
```

(a) creat(“/m”)
(b) mkdir(“/m”)
(c) unlink(“/m”)
(d) append a block to root directory
(e) append a block to root inode

40. Continuing from the above (Q39), one more operation took place, leading to this final state:

```
Final State:
inode bitmap  11000000
inodes       [d a:0 r:2][f a:-1 r:2][][][][][][]
data bitmap   10000000
data         [(.,0) (..,0) (m,1) (o,1)][][][][][][][]
```

What operation was it?
(a) mkdir(“/o”)
(b) unlink(“/m”)
(c) read(“/m”)
(d) link("/m", "/o");
(e) creat(“/o”)

Journaling

The Write Thief actually likes to journal, taking time all day long to write down some thoughts. What are those thoughts about, you ask? Thieving, of course! And here, the Write Thief is removing knowledge about ***File System Journaling!***

41. In discussing crash scenarios during a single data-block append to a file, one crash scenario occurs when “just the **data block** is written to disk. In this case, ______”
(a) the data is on disk, but it can never be read
(b) the data is on disk, and can be read after recovery, but it is garbage
(c) the data is on disk, but the inode bitmap says it is free
(d) the data is on disk, and it can be easily read after recovery
(e) the data never reaches the disk

42. In that same scenario (single block append), another case arises when “just the updated **inode** is written to disk. In this case, ______”
(a) the data is on disk, but it can never be read
(b) the data may seemingly be read after recovery, but it is garbage
(c) the data bitmap and inode bitmap don’t agree
(d) the data is on disk, and can be easily read after recovery
(e) the inode cannot be read

43. In that same scenario (single block append), "just the updated **inode** is written to disk. Sometimes we refer to this as a ______"

[✓ (a) file system inconsistency (the inode and data bitmap disagree)]
(b) file system inconsistency (the data block and inode disagree)
(c) file system inconsistency (the directory and inode disagree)
(d) file system inconsistency (the data bitmap and inode bitmap disagree)
(e) file system confusion

44. "What we'd like to do ideally is move the file system from one consistent state (e.g., before the file got appended to) to another ______ (e.g., after the relevant blocks have been written to disk)"

(a) computationally
(b) passionately
(c) logically
(d) [✓ atomically]
(e) sequentially

45. "Early file systems took a simple approach to crash consistency. Basically, they decided to let inconsistencies happen and then fix them later (when rebooting). A classic example of this lazy approach is found in a tool that does this: `fsck`. … However, `fsck` has a big and perhaps more fundamental problem: it is too ______

(a) [✓ slow]
(b) complicated
(c) redundant
(d) incoherent
(e) fast

46. "The basic journaling protocol includes the following three phases: journal write, journal commit, and ______ (where the contents of the update are written to their final on-disk locations)."

(a) transaction
(b) full write
(c) journal delete
(d) [✓ checkpoint]
(e) phase out

47. “A simpler (and more common) form of journaling is sometimes called **ordered** journaling (or **metadata** journaling) except that ______ is/are not written to the journal.”
(a) inodes
(b) user data
(c) directory data
(d) bitmaps
(e) information

48. “In ordered (or metadata) journaling, data must be written to disk before _______ in order to ensure that a committed inode does not point to garbage data.”
(a) the checkpoint
(b) freeing space in the journal
(c) the transaction commit block
(d) anything else
(e) sunrise

49. “After rebooting, a journaling file system scans the log and look for transactions that have committed to the disk; these transactions are thus replayed (in order), with the file system again attempting to write out the blocks in the transaction to their final on-disk locations. If a crash happens during replay, _______”
(a) all data is lost
(b) the system may not be able to reboot
(c) the recovery starts over after reboot, but might lose data committed to the journal
(d) the recovery starts over after reboot, and should work correctly
(e) you are out of luck.

50. “Data journaling reduces performance by (roughly) a factor of _______ during sequential writes as compared to ordered (metadata-only) journaling.”
(a) 1.5
(b) 2
(c) 3
(d) 4
(e) 5

NFS

The Write Thief has one last task for you: to fix up the ***NFS (Network File System)*** *chapter. Alas! Will the work never end? It will, actually, after these questions.*

51. “The largest benefit of using a distributed client/server file system such as NFS is ______”
(a) performance
(b) sharing
(c) reliability
(d) code coverage
(e) ease of testing

52. “Servers (seem to) crash (or become unavailable) primarily due to power outages, bugs, and _______”
(a) application demands
(b) clients with little memory
(c) slow disks
(d) bears
(e) network partitions

53. “NFS protocol requests, which contain all relevant information needed to complete the request, are sometimes called _______”
(a) stateless
(b) harmless
(c) connectionless
(d) tasteless
(e) quirky

54. “The NFS file handle consists of three parts: volume identifier, inode number, and ______”
(a) file descriptor
(b) security token
(c) smoke screen indicator
(d) request identifier
(e) generation number

55. “During a file open, many ________ requests will likely be made to the server.”
(a) NFSPROC_RMDIR
(b) NFSPROC_MKDIR
(c) NFSPROC_LOOKUP
(d) NFSPROC_REMOVE
(e) NFSPROC_FLUSH

56. “An operation is called **idempotent** when the effect of performing the operation ______ is equivalent to the effect of performing the operation a single time.”

(a) never

(b) once

(c) silently

(d) many times

(e) in reverse

57. “NFS clients handle network packet losses and server crashes uniformly by using a _______ approach.”

(a) caching

(b) oddly efficient

(c) redundancy-based

(d) timeout/retry

(e) handshake/fistbump

58. “NFS clients use caches to improve performance. However, keeping caches consistent is a great challenge. Two primary subproblems of cache consistency are _______.”

(a) latency/staleness

(b) visibility/correctness

(c) choiceness/visibility

(d) correctness/staleness

(e) staleness/visibility

59. “NFS clients buffer writes in memory for some time before flushing them to the server. However, if a client closes a file, all dirty blocks in the client cache are flushed to the server. This ‘flush on close’ behavior addresses the _______ problem.”

(a) latency

(b) staleness

(c) correctness

(d) visibility

(e) choiceness

60. “NFS servers also have a cache, but must flush writes to disk immediately before returning success to clients. The reason for this is ________.”

(a) performance; this approach is usually faster

(b) correctness; this ensures no writes are lost due to an untimely server crash

(c) choiceness; having more choice is critical in the modern world

(d) caching in both clients and servers adds too much complexity to the protocol

(e) lost to history