Assignment 5 - Free Response Questions

● Graded

Student

Christian Blake

Total Points

100 / 100 pts

Question 1

Q1

15 / 15 pts

✔ **+ 15 pts** Excellent

+ 14 pts Good

+ 12 pts Mostly Right

+ 9 pts Right Track

+ 7 pts Valiant Effort

+ 0 pts No Answer

Question 2

Q2

15 / 15 pts

✔ **+ 15 pts** Excellent

+ 14 pts Good

+ 12 pts Mostly Right

+ 9 pts Right Track

+ 7 pts Valiant Effort

+ 0 pts No Answer

Question 3

Q3

15 / 15 pts

✔ **+ 15 pts** Excellent

+ 14 pts Good

+ 12 pts Mostly Right

+ 9 pts Right Track

+ 7 pts Valiant Effort

+ 0 pts No Answer

Question 4

Q4

15 / 15 pts

✔ **+ 15 pts** Excellent

+ 14 pts Good

+ 12 pts Mostly Right

+ 9 pts Right Track

+ 7 pts Valiant Effort

+ 0 pts No Answer/No meaningful answer

Question 5

Q5

15 / 15 pts

✔ **+ 15 pts** Correct

+ 7 pts Incorrect

+ 0 pts No Answer

Question 6

Q6

Resolved **25** / 25 pts

✔ **+ 25 pts** Excellent

+ 23 pts Good

+ 20 pts Mostly Right

+ 15 pts Right Track

+ 10 pts Valiant Effort

+ 0 pts No Answer

For b, the correct answer is 1545 data blocks

Regrade Request **Submitted on: Dec 17**

For part b at the bottom I said, "So, 1545 data blocks to contain all the inodes"

I answered correctly.

My mistake, you did answer it correctly and will get full points

Reviewed on: Dec 17

Question assigned to the following page: 1

CS 480 OPERATING SYSTEMS

Assignment 05

Free Response Questions (100 points)

Due: Beginning of the Class, **Nov 28th**

Late Due: Beginning of the Class, **Nov 30th**

You must work on your own.

Six questions – First five each has **15** points, sixth has **25** points.

Turning In to Gradescope

Submit **ONE PDF** file with your answers in the order of the questions. Make sure the file contains your name and Red ID at the beginning!

Make sure to include the **single examinee affidavit** at the beginning of your answer sheet. You must work on your own for this assignment. It would be a red flag if we find submissions from two students are with same incorrect answers in multiple parts. Also as specifically mentioned in the Syllabus, asking questions, and getting answers from online sources are considered plagiarisms.

SINGLE EXAMINEE AFFIDAVIT

"I, the undersigned, promise that this assignment submission is my own work. I recognize that should this not be the case; I will be subject to plagiarism penalties as outlined in the course syllabus."

Student Name: Christian Blake

RED ID: 824904815

Date: 11/23/23

1. (**15** points) Suppose **file1** is first created in directory A:

• A hard link **hardlink1** is created in directory A pointing to **file1**, • A hard link **hardlink2** is created in directory B pointing to **hardlink1**, • A soft link **softlink1** is created in directory C pointing to **hardlink1**.

If **hardlink1** is deleted, what would happen to **hardlink2** and **softlink1**, **and why**? (i.e., are they valid links)

In this case, hardlink 1 would be set kind of as a pointer to file1, then hardlink2 being set to hardlink 1 would ultimately, still be a hardlink to file 1. This means when hardlink 1 is deleted, hardlink 2 will still point to file1 and be valid. On the

Questions assigned to the following page: 2 and 3

other hand the softlink1 would not be valid, as it is set to hardlink 1, which is now non existent and so it is invalid but still exists.

2. (**15** points) A mini file system uses the File Allocation Table for tracking the allocated blocks for files. It contains a directory and **two files A and B**, and **file A ends at block 7**. Entries with no pointers are empty blocks available for use.

Entry	0	1	2	3	4	5	6	7	8	9	1 0
Pointer	4	-1	8	-1	7	-1	2	-1	10	1	5

a. What are the allocated blocks for the two files and the directory:

- file A:

  Start backwards, so 7, 4, 0, File A starts at block 0

- file B Using remaining data, file B should be starting at not 3 because nothing points to it, but instead 5 so we do the same, 5,10, 8, 2, 6 so we start at 6.

- the directory
  Starts at block 3

b. If another file C is added to the directory and uses all the remaining blocks starting from block 9, what would the FAT table become?

Orange highlight above is the changes

3. (**15** points) The CPU commonly communicates with the device registers of the peripheral devices via two ways:
   - Memory mapped I/O – the hardware associates the device registers with memory address space locations and use the typical load and store instructions to access them.
   - Device port-space I/O – use special instructions (e.g., IN/OUT in Intel architectures) that function like load and store except that they read/write against the device port-space I/O addresses.

Thinking about concepts that we have covered in this class, explain how hardware could prevent user programs from accessing the device registers for:

- address-mapped device registers

base limit registers(older way), mmu

The MMU would ultimately do this for address-mapped device registers. Since the device registers are mapped to memory, the MMU maps the virtual addresses to physical addresses. It also is able to set and prevent user

Questions assigned to the following page: 3 and 4

programs from accessing the device registers. By storing device registers in physical memory ranges that user programs can't access it, the MMU prevents user programs from accessing the device registers.

Some research:
In Tanenbaum pg 341, "with memory mapped I/O, no special protection mechanism is need to keep user processes from performing I/O. All the operating system has to do isd refrain from putting that portion of the address space containing the control registers in any user's virtual address space. Even better yet, if each device has its control registers on a different page of the address space, the operating system can give a user control over specific devices but not others by simply including the desired pages in it's page table"

Cpu access directly. Modern computers have a form of memory word caching. Caching of a device control register could cause this? First reference would be cached, and any subsequent would just map the cache value.

• port-space I/O mapped device registers privileged instructions
Need to use special instructions like IN/OUT. By setting each control register to be assigned an I/O port number(8 or 16 bit integer). The set of all the I/O ports form the I/O port space, which is protected so that ordinary user programs can't access it(only the operating system can). Commands such as IN REG,PORT, or OUT PORT,REG can be used.

4. (**15** points) A file system uses a bitmap to track a disk block free list. The bitmap is implemented as an array of **32**-bit integers. Write C syntax pseudocode (with reasonable comments if necessary) to implement the following function:

/*
* **Given** a bitmap (first argument) stored in an array of words, find, return the
* number of consecutive zeros (free blocks) starting from a block index (second
* argument)
* The block index **starts at 0** from the first bit to the left (most significant) of the
* **first** word, incrementing one at a time to the right, then continuing to the next
* word. E.g., the block index of the first bit of the second word would be 32, etc.
*
* **Return 0** if the bit at the block index is with 1, meaning it is occupied or used.
* **Hint**: to evaluate each bit from the word, use a bit mask and bitwise and. */

#define BITSPERWORD 32
unsigned int countFreeBlocks(int words[], unsigned int blockIndex)
// Find the word index and position in the word

Questions assigned to the following page: 4 and 5

```
        unsigned int wordIndex = blockIndex / BITSPERWORD;
        unsigned int bitLocation = blockIndex % BITSPERWORD;
        //create a count for zeros
        unsigned int count = 0;
        //iterate
        While (words[wordIndex] != 0) {
            //create bitmask at bitLocation
            Unsigned int bitmask = 1 << (BITSPERWORD -1 - bitLocation);

            //check if bit is taken up already
            if ((words[wordIndex] & bitmask) != 0) {
                break;
            }
            //increase count and move to next bit
            count++;
            bitLocation++;

            //move to next word if end of word reached
            if(bitLocation == BITSPERWORD) {
                bitLocation = 0;
                wordIndex++;
            }
        }
        Return  count;
    }
```

Note:

- Error checking is **not** required.
- Assume **int** type has 32 bits.
- Each bit in the bitmap represents one block: 1 is occupied, 0 is free.
- **NO** need to worry about the endianness of storing each integer / word.

5. (**15** points) Is the following file editing operation **idempotent**? Add 2 white spaces at the beginning of a paragraph if the paragraph starts with less than 5 white spaces. Consider all possibilities. **Justify** your answer.

white(0) = " white"   white(white(2)) = "   white"   white(white(white(4))) = "   white" not idempotent

white(2) = "   white"   white(white(4)) = "   white" not idempotent

white(3) = "   white" white(white(5)) = "   white"   idempotent

Questions assigned to the following page: 5 and 6

white(4) = "   white" white(white(6)) = "   white" idempotent

white(5) = "   white" white(white(5)) = "   white" idempotent

No. If you started at 5 you would have no spaces added which would be idempotent. On the other hand if you started at 3 or 4 you would have 2 white spaces added which would still be idempotent because calling again you would still have the same string output. On the other hand, if you went below 2 it is no longer idempotent. This is because you will repeat the operation of adding 2 white spaces which duplicates the operation and changes the output.

6. (**25** points) A multi-level indexed allocation system has:

- **12** direct blocks,
- **5** single indirect blocks, and
- **3** double indirect blocks.
- Assuming **4KB** (4096B) data block size, each i-node pointer entry uses **8** bytes, and **each i-node must fit into one data block**.

4KB *12 direct= 48KB

1 indirect = 4096/8 = 512 *4 KB = 2,048KB

1 double indirect = 512 * 512 * 4 KB = 1,048,576 KB = 1.0486GB

5 single = 10,240 KB = 10.24 MB

3 double = 3.1457 GB

a) What would be the **largest** file size supported by this system?

Around 3 GB or 3,156,016 KB to be exact (answer may be 3,145,728 if only counting the largest)

4KB *12 direct + ((4096/8)*4)*5 + $3(512^2 * 4)$ = 3,156,016 KB

b) Including the root inode, **how many data blocks** would be needed for holding the inodes to describe the **largest** file?

Root inode = 1 block

5 single indirect = 1 block *5 = 5 data blocks

3 double indirect = 512 single blocks + 1 double inode = 513 *3 = 1539

So, 1545 data blocks to contain all the inodes.

c) Starting from the root inode, which i-node blocks need to be visited to find the data block that has **191,470,852**th byte into the file (a direct access into a spot in a file).

Show your calculation.

Hint: watch the class recording for the examples discussed.

My index

191,470,852 / 4KB = 46,745.81 or 46,745

Direct blocks: 46,745 - 12 = 46,733

Single indirect blocks: 46,733 - 2560 = 44,173

Question assigned to the following page: 6

Double indirect block 1 = 262144 blocks contains 512 single's so divide remaining by 512:

44,173 / 512^2 = 0th double indirect

44,173/512 =86.27539 = 86th single indirect

44,173 -(86 * 512) = 141st block