# **CSSE2310 2018 exam answers**

## [**UQAttic**](http://uqattic.net)

# **Get more out of your study time. Join UQAttic's revision chat.**

#### [**Other exam papers**](https://drive.google.com/drive/folders/1k04K5RHL61k3ae2jJ362MPdN5xy3Bqsp)

### Please **contribute** to these documents.

### If you're looking for an effective way to familiarise yourself with the course material, you can't go past collaborating with fellow students. We have laboured to put these up, and so at the very least point out where you think we are wrong!

### You'll get more out of the course, you'll do better in the exam, and other students will benefit from your input as well.

### To get editing permissions, simply go to the [chatroom](http://uqattic.net) and provide us with your Google Account address.

### **Style.**

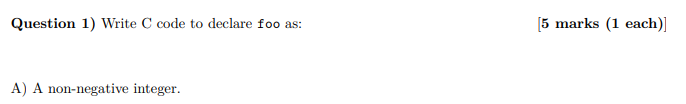
### Type answers in blue beneath each question.

### If you're unsure of your answer, highlight your answer text then hit Ctrl+Alt+M to create a comment beside the text. Once you're satisfied with the answer, click the "Resolve" button on the comment.

### If you want some extra explanation from someone else on their answer, highlight the other person's answer and repeat the procedure above.

### **Communicate.**

### Head over to [uqattic.net](http://uqattic.net/) and click "Chat Now!". You'll find a chatroom full of students just like you. Talk about a revision document (like this one) or swap prep tips. If you have your own IRC client, point it to irc.uqattic.net, port 6667, channel #attic.



* 1. unsigned int foo; uint\_t foo? uint\_t would be wrong - int is arch dependent



* 1. long int foo[18];

long foo[18];

* 1. union Foo { // (We don’t use a struct because “either” 厉害啊)

char\* str;

long length;

}; Unions non-assessable? Do we need to worry about this one?

I thought Joel literally said that unions are non-examinable..

Lecture 3.2 slide 22 says no exam question on unions. Doesn’t mean you can’t use them or that previous exams didn’t include them.



* 1. void (\*foo) (int, double);
  2. int \*((\*foo) (char\*\*)) (const char\*) //<- don't think this is right +1

int (\*(\*foo)(char[]))(const char) // <-don’t think this is right, coz array of strings is 2D array of characters, it should be char[][], char\*\*

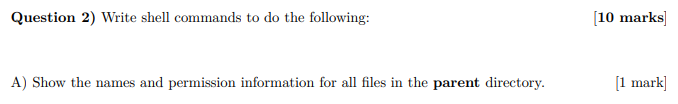
OR int (\*(\*foo)(char\*[]))(const char\*) ([https://](https://cdecl.org)[https://www.microchip.com/webdoc/+avrassembler/avrassembler.wb\_STD\_Z.html](https://www.microchip.com/webdoc/avrassembler/avrassembler.wb_STD_Z.html) [cdecl.org](https://cdecl.org) to confirm)

Looks good to me but for the parameters why did you pass in a char\*\* and const char\*? Arent you suppose to pass in an array of strings and a constant string (hence char[] and const char)? --char[] is an array of chars, char\*\* is an array of char pointers or strings (string == char\*) Gotcha!

Could this also be correct? int (\*(\*foo)(char \*[]))(const char \*) //+2

typedef int (\*)(const char\*) bar;

bar (\*foo)(char\*[]);



1. ls -l ..

ls -la ..

1. ls ./verbose | grep ‘^p.$’ ???? This is not right.. Apparently the internet doesnt know the answer to this one. How surprising… “ls | grep ???????” somewhat works

ls verbose/??????? Thoughts on using ? wildcard?)

ls ./verbose | grep ^.......$ seems to work, and only uses parameters found on the cheat sheet. +4

ls ./verbose | grep -w ‘.......’

ls ./verbose | grep -w -o ‘\w\{7\}’

Or grep ??????? ./verbose | wc -l

ls verbose/??????? (includes the directory in the name tho)

find verbose -maxdepth 1 -name "???????" // this includes extensions though

ls -1 verbose | AWK ‘NR{if (length($1) >= 7) {print $1}}’

cd verbose ; ls -a ???????



1. echo $USER >> goose



1. pgrep -u$USER banzai | wc -l

pgrep -u $U ai | wc -l (-x is needed for pgrep or it will match things like banzai.xxx)j

ps -u $USER | grep banzai | wc -l

ps| grep banzai | wc -l (Current user processes are selected by default) <- no, if we are running multiple terminals with banzai this is not the case

ps -u $USER | grep “banzai” -c (-c for grep counts the lines)



1. svn mv utils.h utils.c (I don’t think this will rename the file on disk) could do svn mv utils.h utils.c & mv utils.h and utils.c (?)

Possibly svn move utils.h utils.c & svn update

Tried it out and the first one seems to rename it on the disk as well

svn mv utils.h utils.c; svn commit

svn rm utils.h

mv utils.h utils.c

svn add utils.c

svn commit

(You must remove utils.h from the svn and then add utils.c, otherwise the svn will still track utils.h (svn status will show utils.h ? instead of A after renaming) and will not change the tracking to utils.c, svn works by filename. Learnt this the hard way in a4 lol)



1. cut -f1 ids | sort | uniq > uids (dont think we need sort as its not asked for) (yes you do need sort, uniq only removes adjacent duplicates).



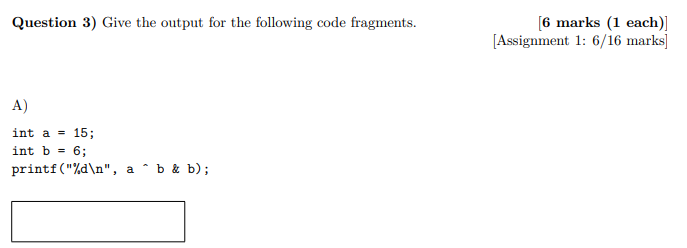
1. mkdir ../backup && cp -r . ../backup && for f in \*;   
   do mv ../backup/$f ../backup/$f.bak; done

Shorter solution: (cp and rename can be done in one command)

`mkdir ../backup && for f in \*; do cp $f ../backup/$f.bak; done

cp \* ./backup/\*.bak (don’t think this works)

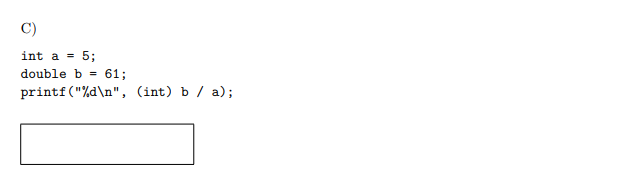
Mkdir ../backup; for i in $(ls);do cp $i ../backup/$i.bak;done



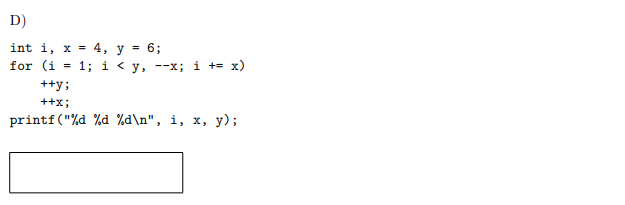
1. ~~6 -~~ I think the answer is 9, as the bitwise AND comes before XOR - yeah, you’re right woops <- last page cheat sheet has roughly correct ordering (cast and increment/decrement are out of place)



1. 1 (how did you get this? +1) verified on moss. Technically 0 is false and any other value is true iirc? Can’t find spec definition so could be implementation dependent? I believed they used 2’s complement by flipping bits all from 0 to 1s resulting 1111 1110 then + 1 This becomes 1111 1111 then & 1 = 1 ? Is it not just true && true? Only 0 is false, so 1 and -1 == true, and as %d is used, since true == 1, 1 is printed...



1. 12 [+1]



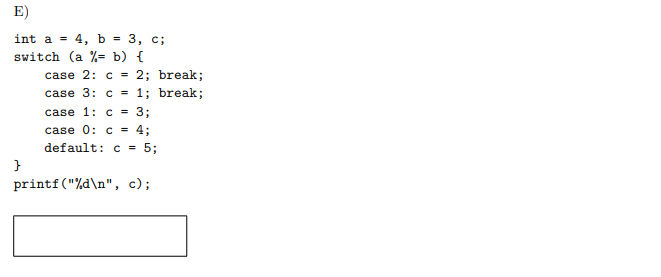
1. 7 1 9 (is this not an error?) why would it be? There( are no brackets ({}) for the if statement) i copied and pasted it and ran it and there were no issues. The if statement? Like within the for loop? That works out to i < --x because it’ll take the right side of the comma as the comparison. The condition for the for loop is just --x, it takes just the right side of the comma. Because there is no {} around the for loop it will only run through what’s directly below it. I.E. ++y is done multiple times but ++x is done after the for loop is finis1 hed. There are no “{ }” for the for loop.

There doesn’t need to be {} for a for loop, it will just go until the first ‘;’ +11

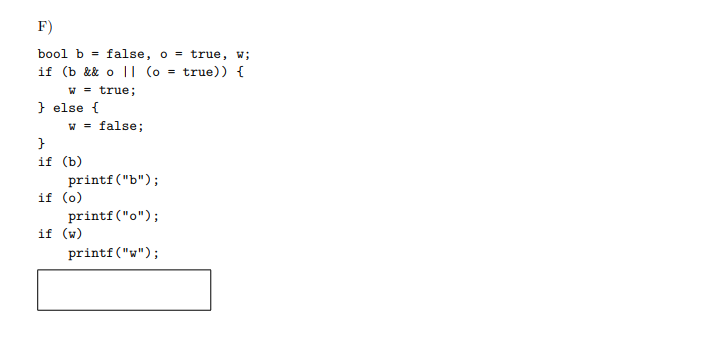
\*Sidenote: for (stuff; stuff; stuff)**;** will not run code on the following line

Y’all really just gonna abandon the style guide and write code like animals? The indentation isn’t even right to do what you’ve done. I will not be constrained by the tyranny of the style guide

Wtf I got



1. 5 code gives 5 when tested



1. ow (Relatable +9) assignment 4 feews, owo what’s thiws, begone - > wdym ass4 was juwst stwing pwocessing with extwa steps wow <- oo wa wa, someones gonna get waid in univewsity UwU [-1000000001]



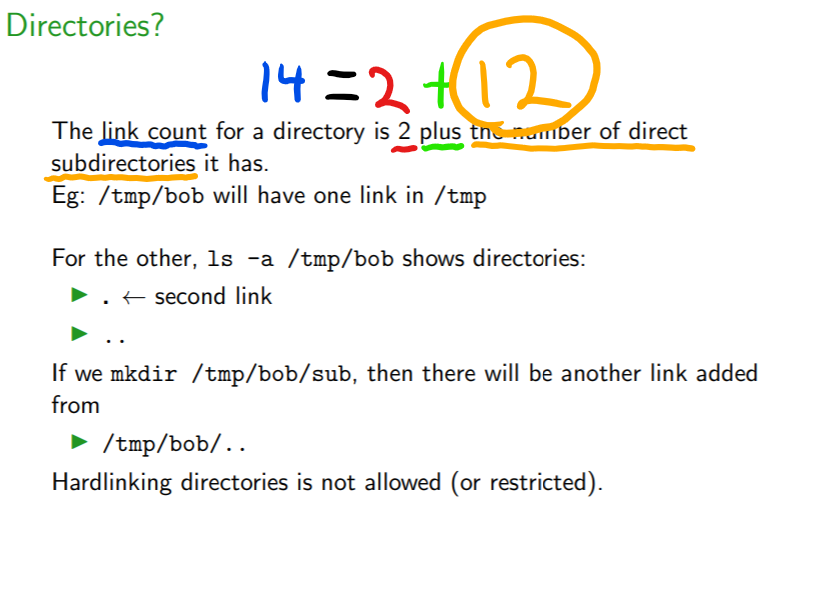
1. .  
   ..  
   empty  
   Flamboyant (how?) (shouldn’t this depend on lolcat’s permissions?, so unknown?) i think from what the tutor said on Piazza that having r and x allows you to read it  
   full  
   Include (how?) greg is not in bots - therefore he’s in the other right, which has read? +12 err

Thonk



1. 12 (? 2+num subdirectories) (No. Links = 2 + num subdirectories. Subdirectories = No. Links - 2) (Total - parent dir - current)/2 = Ans = 6

Why /2? It isn’t /2. Lecture slide below, the answer is 12.





1. (Tentative) depends on block size. E.g. if block size is 4096 bytes then 273547264? Without unstated block size just assume block ends at file end and give 273547264. If block size known and part of last block was empty you would add to it. (How to determine?) So what is the answer for this?? It says “created as a regular user” and others cannot write to the current directory, so would it be nothing? “Regular users” cannot create files?? According to Piazza, there seems to be a few ways to approach the situation. Eg. no block size is known so the max file will be equal to what was deleted. Alternatively, the inode setup isn’t known either so maybe the max size is less than the size of full. The primary thing seems to be that you explain your approach.

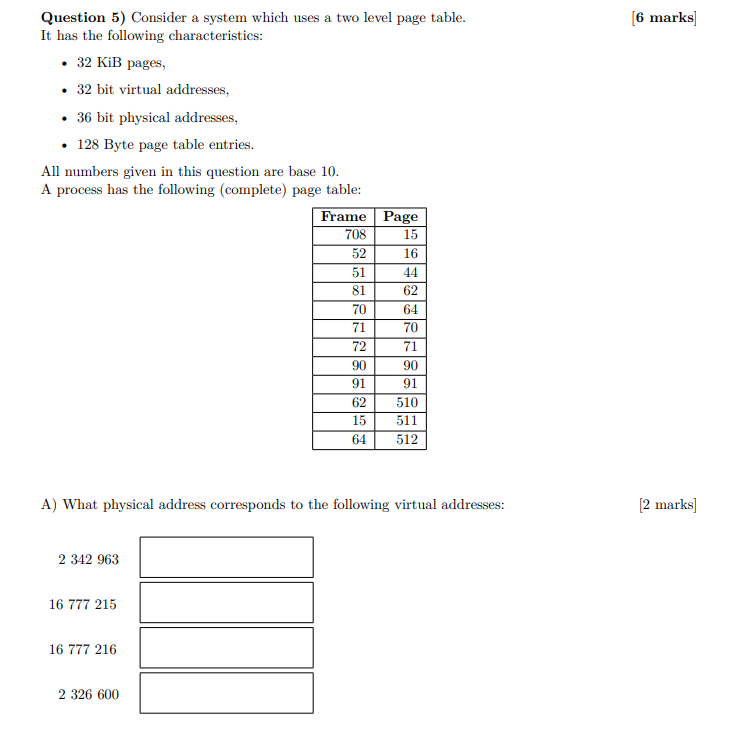


1. tman requires read permissions as well: require chmod u+r shticsk.sh ~~u+x .\~~

(Surely its execution permissions and not read? (chmod g+x shtick.sh))

Tman is the owner, so the “u” option applies . (T1HIS->) For .sh (script or interpreted language files) files, you need to be able to read the file to read what commands to run.  
Tman is in yaks, the group only has write permissions, or does ownership take precedence?

^ doesnt matter because hes the owner - yeah - g only applies to users in the group who are not the owner



Note for this question page and frame have been swapped around. Sick prank joel

1. 1. 2375731 yu
   2. 524287
   3. 2097152 z
   4. 2359368

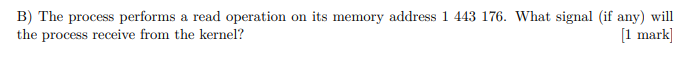
A bit more detail:Y

Page size = 2^15 (32768Bytes) = (1KiB = 2^10 + 32KiB = 2^15)

Page Number = Virtual Memory / page size (use table to get page frame)

Offset = Virtual Memory % page size

Physical memory = Frame(from page table) \* page size + offset



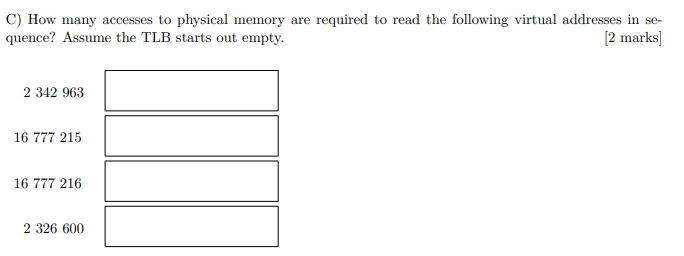
You would get a SIGSEGV signal.

So what’s the answer to this (5b)? -> sigsegv

Can someone explain sigsegv? Is it a physical address at frame 44? Or is it a virtual at pg 44? Cause if its virtual nothing would be received as it exists (+2)

Remember that this is an abstraction, so the process only sees the virtual addresses. Since there’s no frame 44, we win a SIGSEGV, since we’re trying to access something out of bounds (I stand corrected, this is wrong). I thought the frames were physical and the pages were virtual. A process sending a request to the kernel would use the virtual address, the virtual page 44 exists and so as far as I’m aware there shouldn’t be a signal? (slide from revision session belo w) **+9**

Also note that the table is flipped around so frame <- page.



1. 2 (why is not 2? That’s two level page tablbye) would it be 2 2 2 1 ? Wouldn’t this be 3 memory accesses? The only reason why the answer might be 2 is because the memory access in level 1 page does not count (as we are only finding physical address accesses, hence the level 2 page finds the physical address (+1 count) and we access it after we find it (+1 count)), is this right? +1  
   2  
   2   
   1 (Could someone explain why the last is 1? Thanks) (Yes pls why is this 1 instead of 0?) its asking for how many accesses to the memory, not how many pages need to be accessed, since the physical address of the page number is already stored in the TLB, you don’t need to access the memory in level 1 and 2 of the page table which finds the physical address of the virtual memory, hence you do not need to access any page table to identify which level 2 page contains the address for page 71. So there is only 1 memory read (which is straight to the physical address directly).

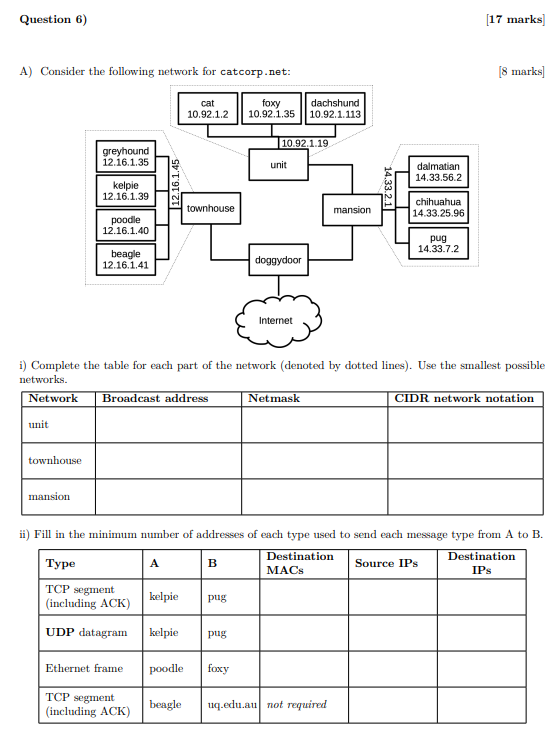
Page tables are stored in RAM so there is one read from memory for each page table level plus one for the memory at the physical address. I think 3,3,3,1 (+8909 0909090) The tutor also confirmed 3,3,3,1 in the review session

That s a lot of damage! +10

What the faaaaaakkk Peter said it was 3221 in the first revision session :(((

Because at the start it says its a two level page table, so it requires 1 extra access. The last one already has the 2 pages accessed so it only needs to access the physical memory and no pages. Virtual -> page level 1 -> page level 2 -> physical [+1]

3 3 3 1



127

| Network | Broadcast (do this last) | Netmask | CIDR |
| --- | --- | --- | --- |
| unit | 10.92.1.127 | 255.255.255.128 | 10.92.1.0/25 |
| townhouse | 12.16.1.47 +14  12.16.1.63 +3 | 255.255.255.240 +12  255.255.255.192 +2  (8+8+8 = 24, 28 - 24 = 4 bits from the left (128+64+32+16 = 240)) | 12.16.1.32/28 +16  12.16.1.0/26 ?? |
| mansion | 14.33.63.255 | 255.255.192.0 | 14.33.0.0/18 |

(Could someone explain townhouse CIDR logic?)

Smallest network -> least host bits jjjjjjjjjjjjj

As I understand in townhouse, bit 28 is the last bit before they differ

Broadcast of Townhouse, which one is correct? Pink or blue? Blue since the first bit to change is the 29th bit

What does the +6, +4 and +8 mean though, is that just ppl voting? Yes.

Why is mansion 18 not 16?

\*shrug\* these I’m not sure i have any clue about

From what I understand, TCP requires a src/dest MAC address, since it needs to go back

UDP don’t (?), ethernet frames maybe? Need to re-read notes, so feel free to change

| 8 | 2 | 2 |
| --- | --- | --- |
| 4 | 1 | 1 |
| 5 | - | - |
| - | 2 | 2 |

Confirmed by a tutor session

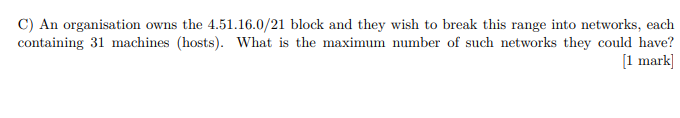
Help why the heck is top left 8? Due to ACK on TCP having to return the message ( yes because of ack return)

There’s a better explanation on piazza but basically we are counting each distinct MAC address used- so one for receiving and one for sending (as this is TCP with ACK, our dest is sending a message back to our src). As there are four hops between kelpie and pug), there’s 8 MAC addresses used.



B/ Physical, possibly link (if diffusederent hardware )

ii/ TCP (Transmission Control Protocol) Remember the whole two generals problem in the lectures? I think the answer would be none here because you can’t guarantee that the ACK is received. +1. Tutor in tutoring session said the answer is TCP ( guy in yellow just got shutdown) +9999999999999999999999999999999999999999999999999999999999999999999999999999010100110100111101010011 \* googolplex



C/ 2^(32-21) = 2^11 total host addresses

floor(2^12 / 31) = 132 Networks

2^(32-21) = 2^11(+1)

*Number of host addresses = 2^(32-21) = 2048*

*Max no. of networks = floor(2^11 / 31) = 66 Networks*

*Didn’t Joel say we need to take into account the 2 unusable addresses? So wouldn’t it be (2^11/33)?? Could be wrong, I just thought w e had to add 2 somewhere in the calc??*

*Joel: “So subnet A.B.C.D / x has 2^(32-x) -2 usable host addresses”*

*i.e. 2^(32-21)-2 = 2046, Max no. of networks floor(2046/31) = 66 Networks. Each network needs its own Broadcast address and Network address. I believe the below answer is correct.*

Thought it was machines + 2 so 31+2 = 33 and since 33 does not fit in 2^5 anymore you must use the next one up so 2^6

Why do you +1 to 31 here?

Then you 32 - 21 = 11 host bits

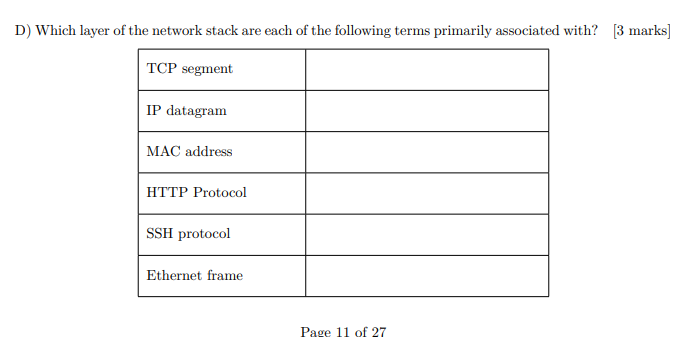
2^11 / 2^6 = 32 max networks **(+2] (TUTOR VERIFIED)**

*Number of network addresses = 2^(32-21) = 2048*

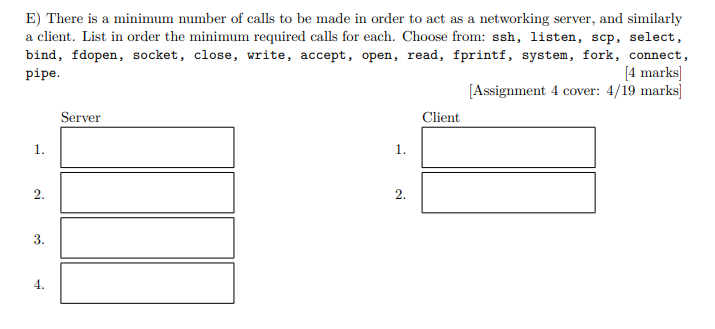
*Number of host addresses = 2048 - 2 = 2046*

*Each subnetwork should hold 31 machines, including broadcast and network addresses this is 33. Requires 6 bits per subnetwork.*

*Number of subnetworks =floor( 2046 / (2^6)) = 31 networks. Explanation?*



D/ Transport, Network, Link, Application, Application, Link[+1]



E/ socket(), bind(), listen(), accept() - anyone who wrote fprintf() I wish you luck in your future endeavours [+420]

socket(), connect() [+69] nice [+6]

7/ 2^(12 - 4) = 2^8 Pointers/Block

We add 1 due to 0 indexing

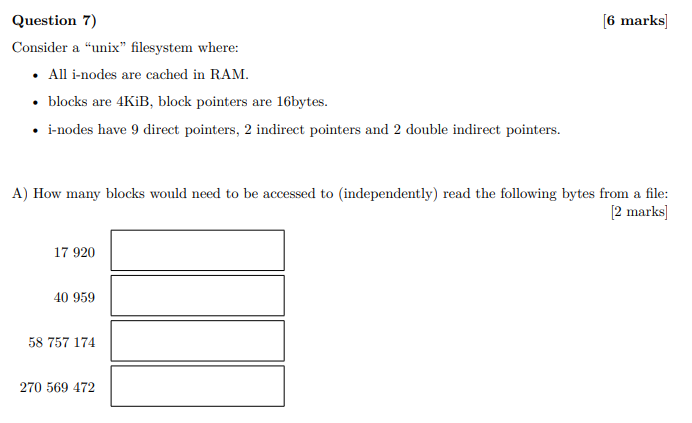
17920 + 1 = 4 \* 2^12 + … => 1 read

40 959 + 1 = 10 \* 2^12 + 0 => 2 reads, first byte of first indirect (Explain why first byte? 9x4Kib = 36864, so wouldnt this be the first byte of first indirect??)

Number of bytes in the single indirects = 9 \* 2^12 + 2 \* 2^8 \* 2^12 = 2,134,016

That last two would require 3 reads (2 for indirects)

B/ (9 + 2 \* 2^8 + 2 \* 2^8\* 2^8) \* 2^12 = Number of pointers \* block size = 539,004,928 Bytes or 526372 KiB



**Q7.**

1. Answers from piazza/revision sesh

1 block - 17,920

2 block - 40 959

3 block - 58 757 174

3 block - 270 569 472

Pls feel free to elaborate on these answers

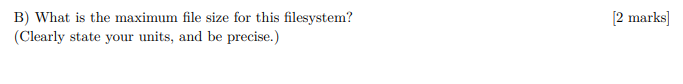
Space that direct pointers take => 9\*4096 = 36,864 bytes

17920 comes under that range, and to access a direct pointer we only read 1 block.

+ (2 \* 256 \* 4096) = 2 134 016Bytes so the 2 indirect pointers have a range from 36,864 (incl) to 2 134 015 bytes

I’m not too sure about the calculation involved to say that 3 blocks need to be read but i’m pretty sure they come under the 2nd indirect pointer range, so if someone could add the calculation bit it’d be super helpful.

For the range of the 2nd indirect pointers: 9 \* 4096 + (2 \* 256 \* 4096) + (2 \* 256^2 \* 4096) = 539 004 928 bytes. Range = 2 134 016 (incl) to 539 004 927 bytes (counting bytes starts at 0).

(B)

I think that the fourth one would be 4 reads 66057.06 at least 1 + 1 + 2 (for the extra read because not coverage with the double indirect pointer) [+3]

Working:

Pointers per block: 4096/16 = 256

Total number of blocks on the system: 9 + (2\*256) + (2\*256^2) = 131593

Size = no. of pointers \* block size = 131593 \* 4096 = 539004928 bytes 539004928/ 2^10 = 526,372 KiB

4 KiB x (9 + 256 + 256 + 256^2 + 256^2) = 526,372 KiB [+9] ?

There are two indirect pointers not 1

So which one is right here? Why do we have to divide by 2^10?



C/ ((2^8)^3 - (2^8)^2) \* 4 = (triple indirect entries - double indirect entries) \* blockSize = 66,846,720 kiB [+3]

Probably a much more straightforward way would be:

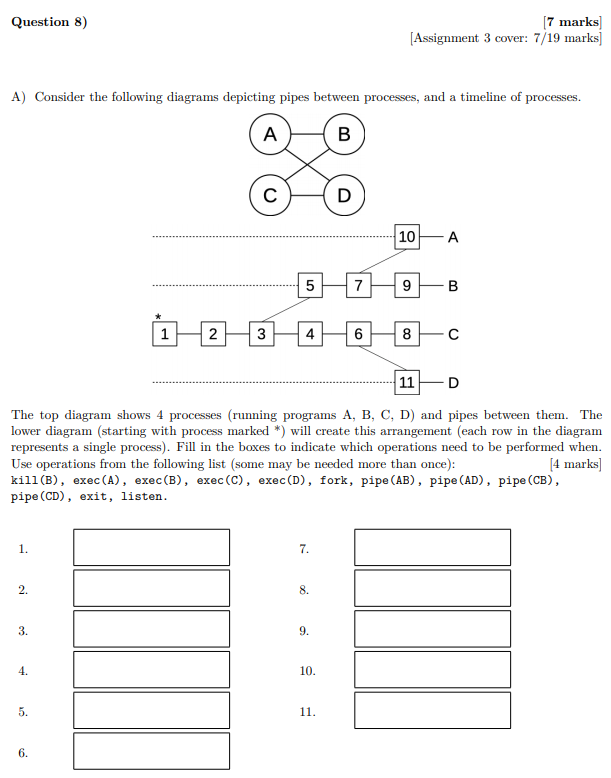
New modified quantities (9 direct, 2 single indirect, 1 double indirect, 1 triple indirect)

Max size = [(9 + 2\*2^8 + 2^8 \* 2^8 + 2^8 \* 2^8 \* 2^8)] \* 4 = 6737309 KiB

Difference = 67373092 - 526372 = 66846720 KiB [+3]

**Hey can someone explain where 526372 comes from??**

526372 is the max file size from the previous section (B), which still have the original quantities (9 direct, 2 single indirect, 2 double indirect).

8/

A/ First two are interchangable

~~pipe(CB), pipe(CD), fork, <empty>, pipe(AB), fork, fork, exec(C), exec(B), exec(A), exec(D)~~

Does empty can be pipe(AD)? No - pipes need to be created before the fork so both pograms have access to the pipe. Since <empty> comes after the fork of B (which forks A), there’s no way for A to inherit pipe(AD) from its parent. See below.

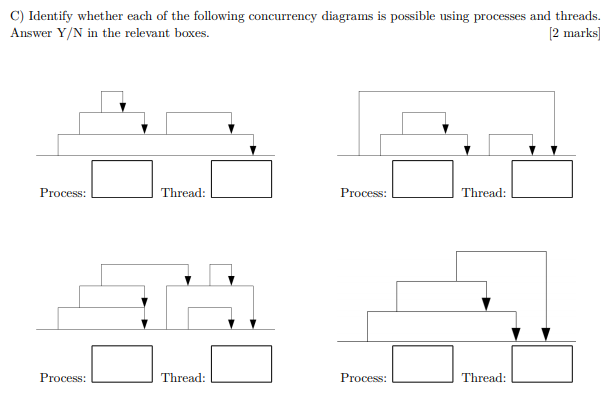
1. pipe(CB),
2. pipe(AD),
3. fork(),
4. pipe(CD),
5. pipe(AB),
6. fork(),
7. fork(),
8. exec(C),
9. exec(B),
10. exec(A),
11. exec(D)[+9][Home](https://www.youtube.com/)

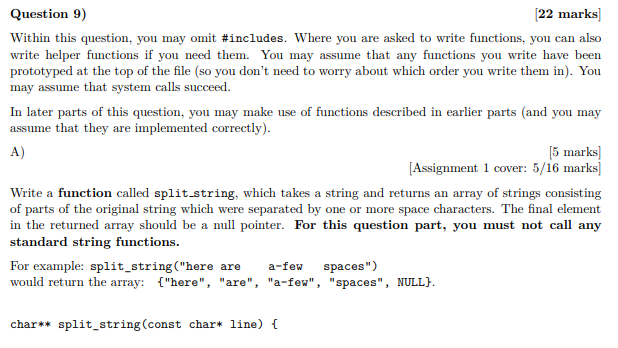
Why is the pipe(AD) before the fork? Won’t be inherit this and B is not supposed to be connected to D? The pipes do not need to be open from B to D but A cannot connect to D without this being set up here.

B/ kill -INT 517

kill(517, 1); OR kill -1 517 OR kill -SIGINT 517 (tested, works) [+2]

kill -2 517 (+2) use “kill -l” to list the signals and see what number you need. SIGINT is 2.

C/ Y, Y; Y, Y; N, Y; N I d be Y[+3] because init process reaps child +1 (tutor confirmed thish in revision session) , Y

9/ (The best part of the exam)[-420] Keep in mind there are many ways to answer these questions (rip markers). In general, keep an eye out for first/final cases in loops, adding 1 to char\* lengths if you want a string for the NULL terminator. I didn’t see anything saying good style is required (and it would be crazy if it were), but remember that some semblance of style makes the code easier to understand (and mark, especially for variable names and indentation).

Being forced to hand-write code in an exam should be punishable by segfault. +10

char\*\* split\_string(const char\* line) {

int word\_count = 0;

int char\_count = 0;

char \*\*strings = malloc(sizeof(char\*) \* 10);

char \*word = malloc(10);

char \*empty\_string = NULL;

int cont\_space = 0;

for (int i = 0; line[i] != '\0'; i++){

if (line[i] == ' '){

if (!cont\_space){

strings = realloc(strings, sizeof(char\*) \* ++word\_count);

strings[word\_count - 1] = word;

word = calloc(sizeof(char), char\_count + 1);

char\_count = 0;

}

cont\_space = 1;

}else{

cont\_space = 0;

word = realloc(word, sizeof(char) \* ++char\_count);

word[char\_count - 1] = line[i];

}

}

strings = realloc(strings, sizeof(char\*) \* ++word\_count + 1);

strings[word\_count - 1] = word;

strings[word\_count] = empty\_string;

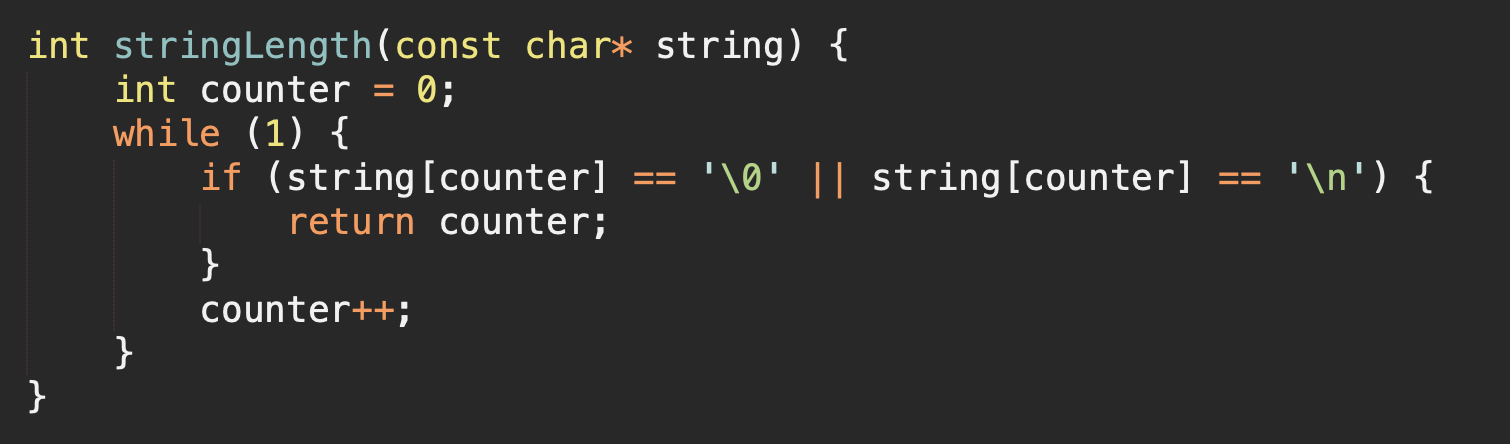
return strings;

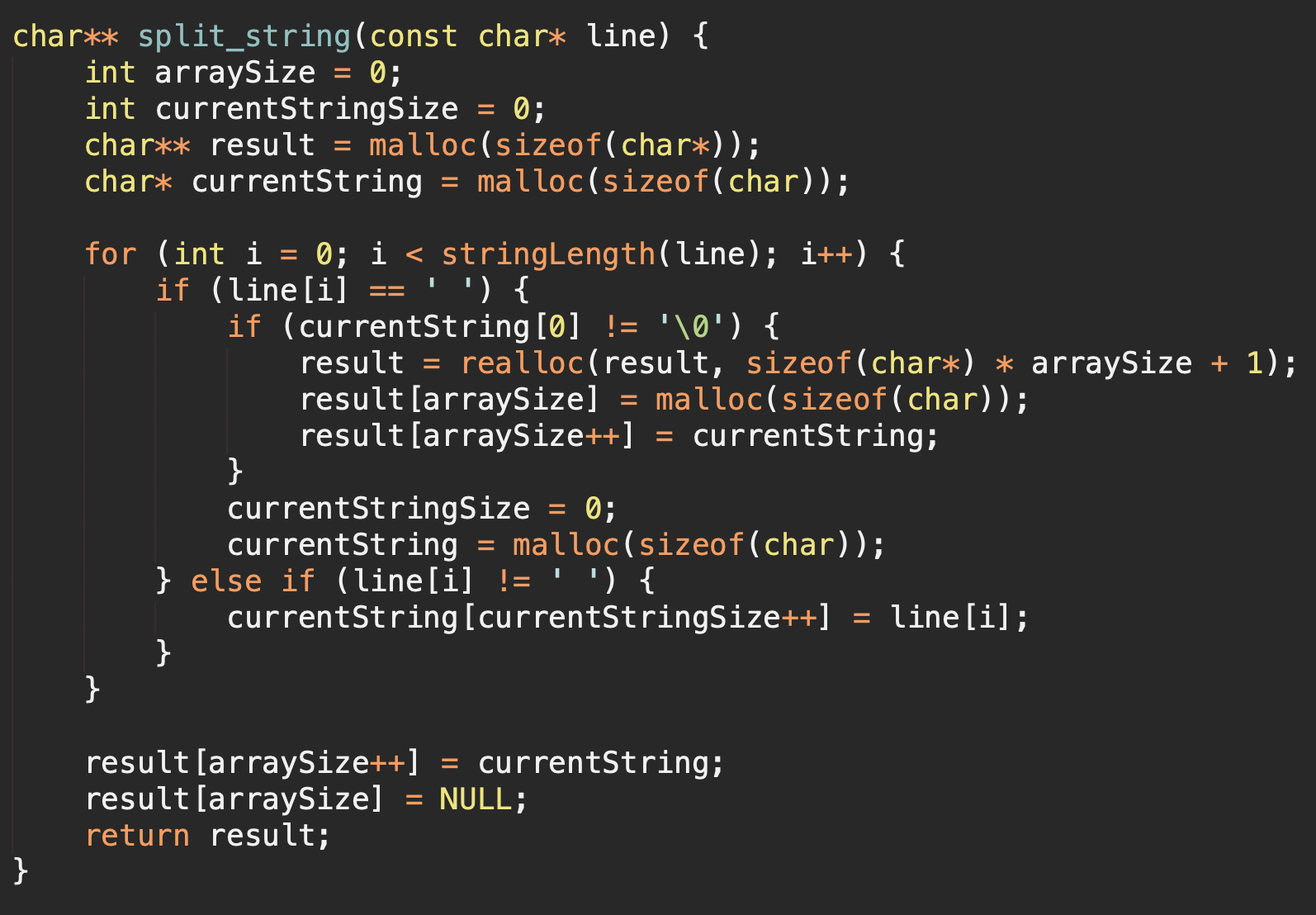
}

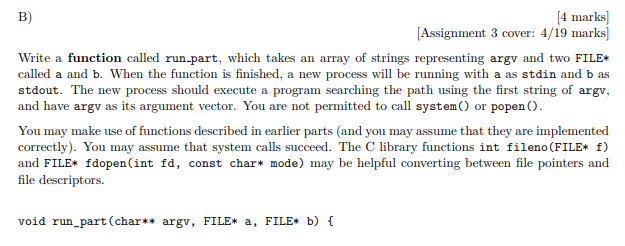
This is gonna result in a string containing just a space at the end of the array (before NULL), if we put in something like “hi j “,i.e, result in {“hi, j, “ “, NULL}

If you mean the code with a black background then it works fine as far as im aware. Tested it out as well.

A)





9B???

Given we don’t know the length of argv, I’m assuming argv already has NULL at the end

argv always has NULL at the end

2

{

If (fork == 0) {

dup2(fileno(a), fileno(stdin));

dup2(fileno(b), fileno(stdout));

fclose(a); // Why are we closing??? The child is closing the duplicates.

fclose(b); // Probably not 100 % needed Tested it and you don’t need the fclose

execvp(argv[0], argv);

}

}

void run\_part(char\*\* argv, FILE\* a, FILE\* b) {

char\*\* spl = split\_string(argv); // No need to split here, argv it self is a string[][]

if(fork()) {  
 dup2(fileno(a), 0); // or fileno(stdin) pretty sure its always 0

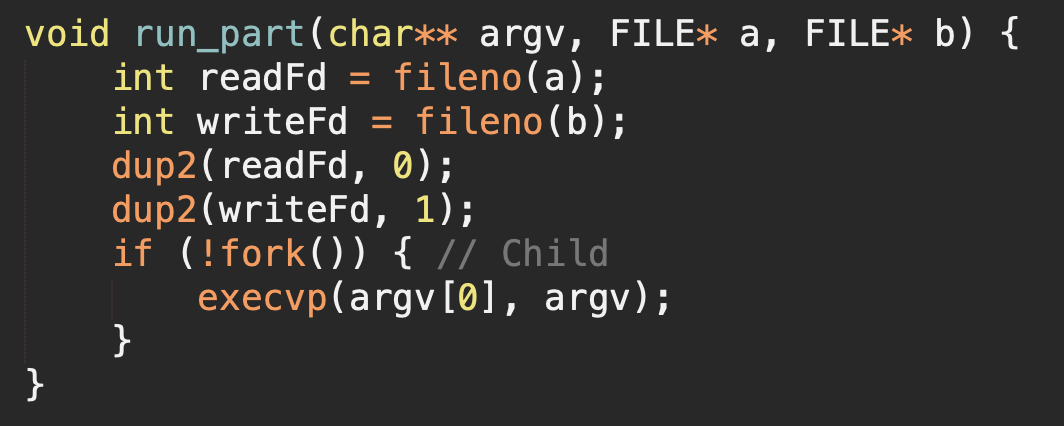
dup2(fileno(b), 1);

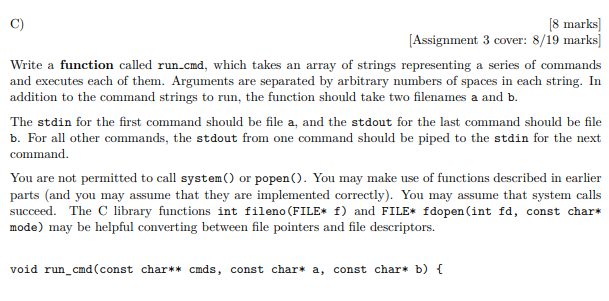
execvp(spl[0], spl);

}

}

B)



****

**C**

C. Here we will assume that argv is terminated by NULL

{

Int pipes[2];

pipe(pipes); // remember that 0 is read, 1 is write

// First program

If (fork() == 0) {

dup2(inFd, fileno(stdin));//stdin for the first command be file inFd

dup2(pipes[1], fileno(stdout));//stdout

close(pipes[1]);

fclose(a); //a is a filename not a file stream?

fclose(b);

Char \*\*parts = split\_string(argv[0]);

execvp(parts[0], parts);

}

close(pipes[1]);

Int inFd = pipes[0], outFd; // these will track the pipes we use

// Should inFd and outFd be declared before the first fork?

// Intermediates

Int i;

For (i = 0; cmds[i + 1] != NULL; i++) {

pipe(pipes);

outFd = pipes[1];

If (fork() == 0) {

dup2(inFd, fileno(stdin));

dup2(outFd, fileno(stdout));

fclose(a);

fclose(b);

close(inFd);

close(outFd);

Char \*\*parts = split\_string(argv[i]);

execvp(parts[0], parts);

} else {

close(outFd);

close(inFd);

inFd = pipes[0];

}

}

// Fork for B

If (fork() == 0) {

dup2(inFd, fileno(stdin));

dup2(fileno(b), fileno(stdout));

close(inFd);

fclose(a);

fclose(b);<http://www.waves.utoronto.ca/prof/svhum/ece357/problems/probset4soln.pdf>

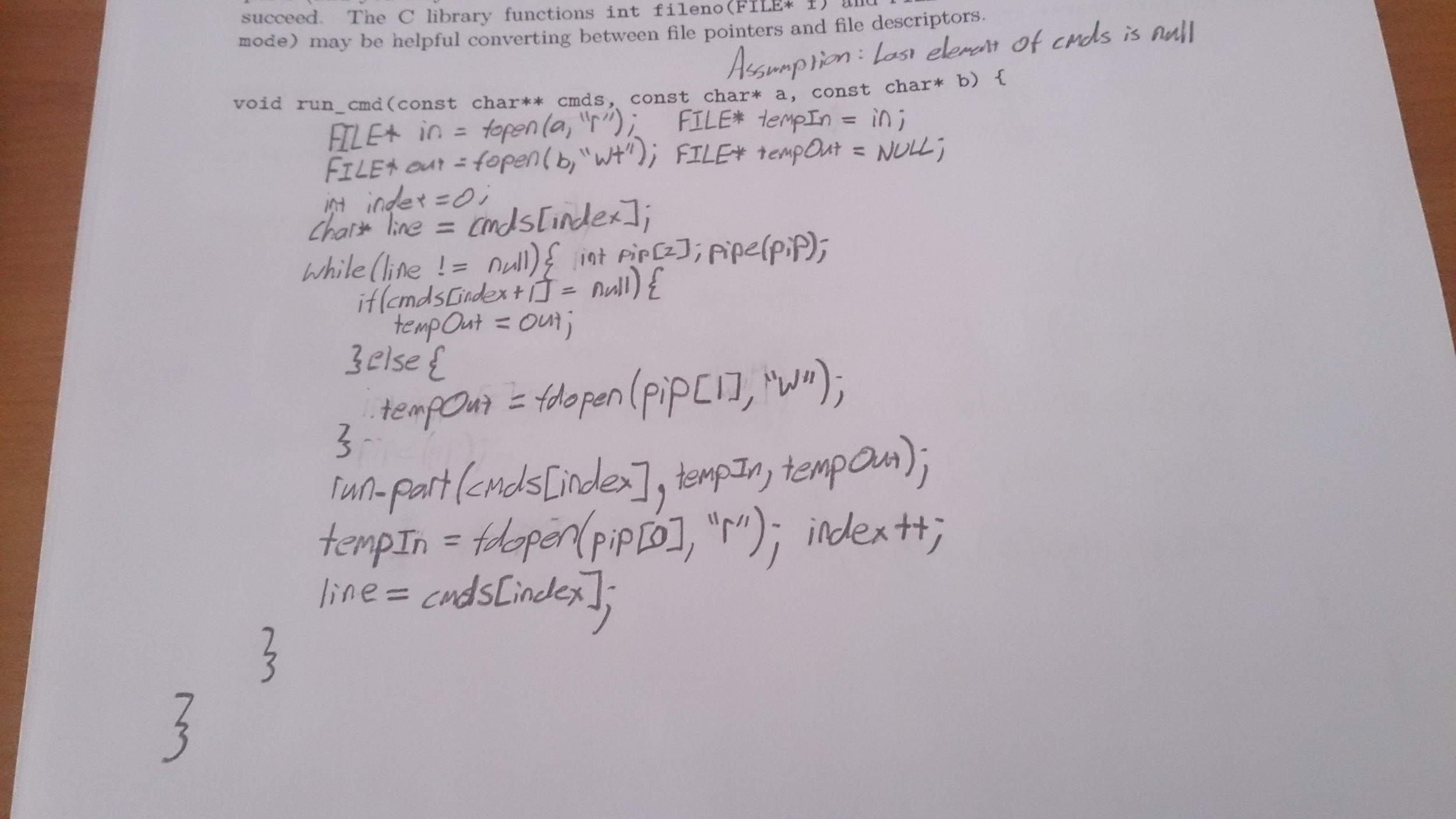
Char \*\*parts = split\_string(argv[i]);

execvp(parts[0], parts);

}

close(inFd);

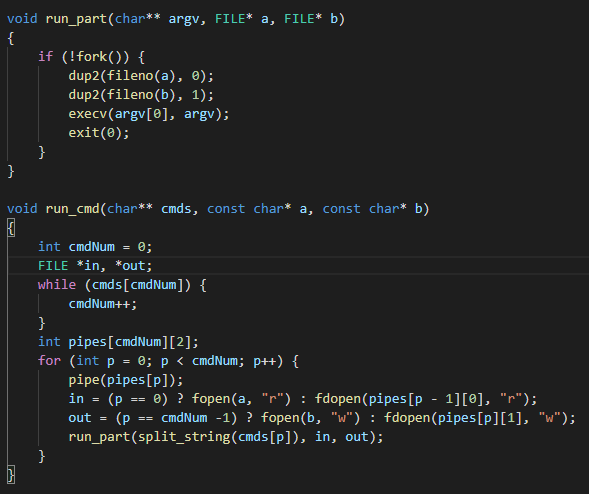
}



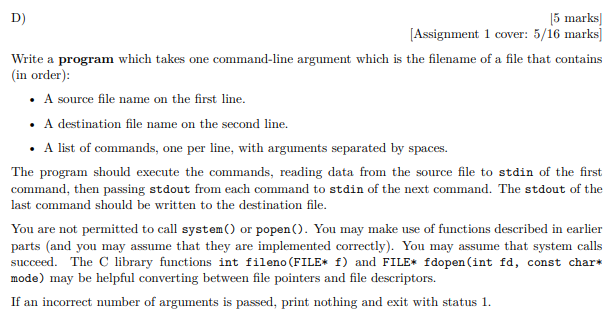
Copied from piazza post 1758 (^)

Has anyone tested this works? It is so confusing

Screwed for question 9 :’’(



Tested and work^

**D**

D. For d-whydoesthisexamneverend :(

Int main(int argc, char\*\* argv) {

If (argc != 2)

Return 1;

}

FILE \*file = fopen(argv[1], “r”);

FILE \*src = fopen(read\_line(file), “r”); // I’ll define this later

FILE \*out = fopen(read\_line(file), “w”);

Int lineCount = 0;

Char \*\*cmds = NULL;

Char \*line;

While ((line = read\_line(src)) != NULL) {

lineCount++;

cmds = realloc(cmds, sizeof(char \*) \* lineCount);

cmds[lineCount - 1] = read\_line(src);

}

run\_cmd(cmds, src, dest);

Return 0;

}

// Here’s an interesting thought: could we use getline() since it’s only banned in assignments?

// It would be interesting because it does what this function does: to auto-reallocs char\*

// Though unlike this function, it includes \n

Char \*read\_line(FILE \*stream) {

Int charIn, strSize = 1;

Char \*output = NULL;

While ((charIn = fgetc(stream)) != (int)‘\n’ && !feof(stream)) { //Think this should be fgetc [Fixed]

strSize++;

Output = realloc(output, sizeof(char) \* strSize);

Output[size - 2] = (char)charIn;

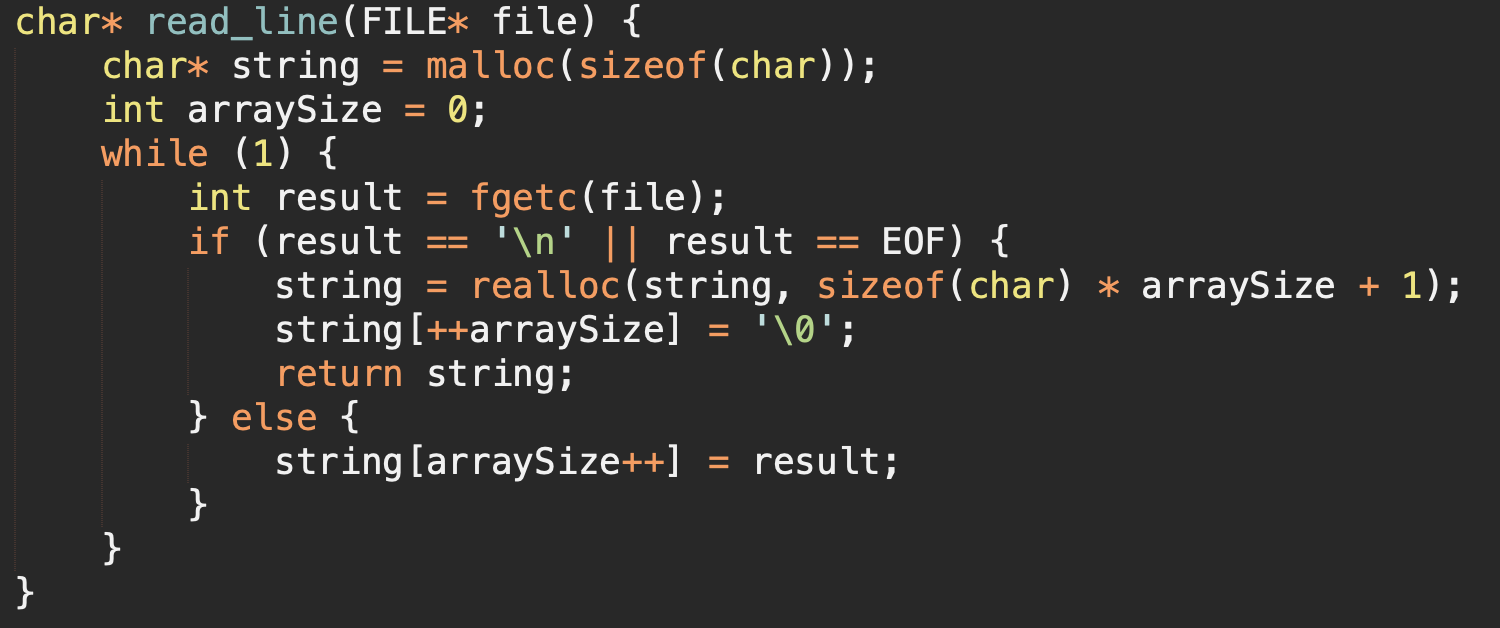
Output[size - 1] = ‘\0’;

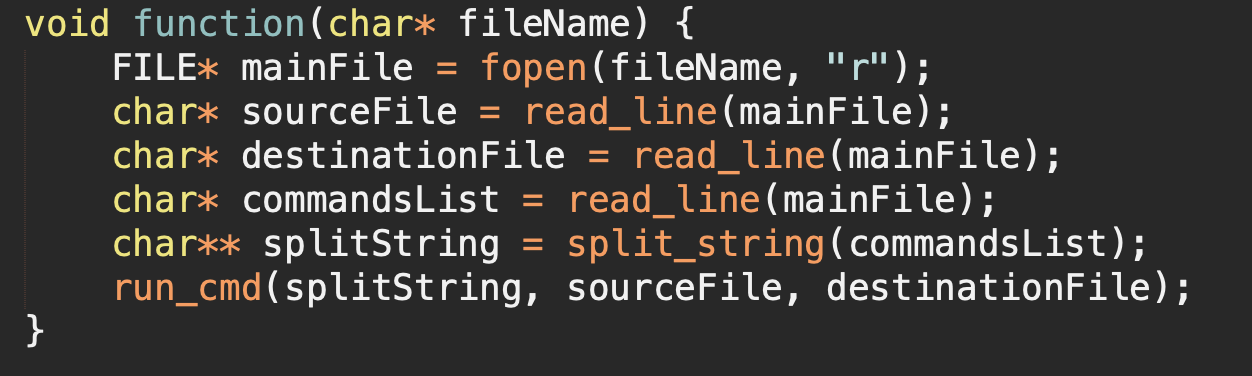
}

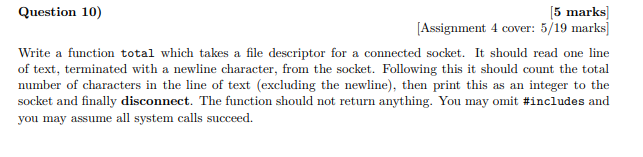
Return output;

}

D)





**10.**

//haven’t validated this by running it, but should be ok

Void total(int fd) {

Char \*buffer;

Int numOfChars = 0;

FILE \*file = fdopen(fd, “w+”);

fgets(buffer, 80, file);

For (int i = 0; buffer[i] != ‘\0’; i++) {

If (buffer[i] != ‘\n’) {

numOfChars ++;

} else {

Break;

}

}

fprintf(file, “%d\n”, numOfChars);

fflush(file);

fclose(file);

close(fd); // disconnect? If you call fclose then you don’t need to call close.

}

// Q10. Here is another try to solve fixed buffer size

// correct it if anything wrong I think should dup() when opening sock a second time, which also requires both to be closed. Socket is double direction, not like pipe (+1)

void total(int sock) {

FILE\* fileR = fdopen(sock, "r");

FILE\* fileW = fdopen(dup(sock), "w");

int total = 0;

char next;

while(next = fgetc(fileR), next != '\n') {

total++;

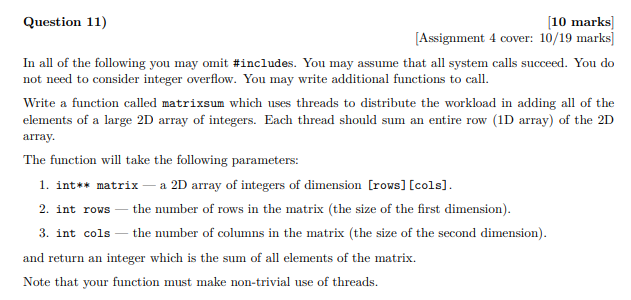
}

fprintf(fileW, "%d", total);

fflush(fileW);

close(sock);

}

11/

typedef struct {

int rowNum;j

int maxCol;

int\* matrixRow;

int\* sum;

pthread\_mutex\_t\* guard;

} Param;

int matrixsum(int\*\* matrix, int rows, int cols) {

pthread\_t tid[rows];

pthread\_mutex\_t guard = PTHREAD\_MUTEX\_INITIALIZER;

int sum = 0;

for (int i = 0; i < rows; i++) {

Param\* param = malloc(sizeof(param));

param->rowNum = i;

param->maxCol = cols;

param->matrixRow = malloc(sizeof(int\*) \* cols);

for (int j = 0; j < cols; j++) {

param->matrixRow[j] = matrix[i][j];

}

param->sum = &sum;

param->guard = &guard;

pthread\_create(&tid[i], NULL, sum\_helper, param);

}

for (int i = 0; i < rows; i++) {

pthread\_join(tid[i], NULL);

}

return sum;

}

void\* sum\_helper(void\* arg) {

Param\* param = (Param\*)arg;

for (int i = 0; i < param->maxCol; i++) {

pthread\_mutex\_lock(param->guard);

\*param->sum += param->matrixRow[i];

pthread\_mutex\_unlock(param->guard);

}

pthread\_exit(0); // I think you could also just return NULL;

}

//Help my peepee, see you later virgins [+8]

Alternate solution for Question 11 which leverages the advantages provided by threading. The above solution enables only one thread at any given time to sum the total number due to the mutex lock therefore it is slower then summinmg the matrix traditionally even with multiple cores. This alternate solution sums each row individually, and returns the sum of the row. These sums are then added together. Has been tested and works.

typedef struct {

int\* row;

int len;

} SumArg;

void\* sum\_row(void\* raw) {

SumArg\* arg = (SumArg\*)raw;

int\* row = arg->row;

int len = arg->len;

free(arg);

int\* sum = (int\*)malloc(sizeof(int));

\*sum = 0;

for (int i = 0; i < len; i++) {

\*sum = \*sum + row[i];

}

return sum;

}

int matrixsum(int\*\* matrix, int rows, int cols) {

pthread\_t\* threads = (pthread\_t\*)malloc(sizeof(pthread\_t) \* rows);

int i;

for (i = 0; i < rows; i++) {

SumArg\* arg = (SumArg\*)malloc(sizeof(SumArg));

arg->row = matrix[i];

arg->len = cols;

pthread\_create(&threads[i], NULL, sum\_row, arg);

}

int sum = 0;

for(i = 0; i < rows; i++) {

void\* retVal;

pthread\_join(threads[i], &retVal);

sum += \*((int\*)retVal);

free(retVal);

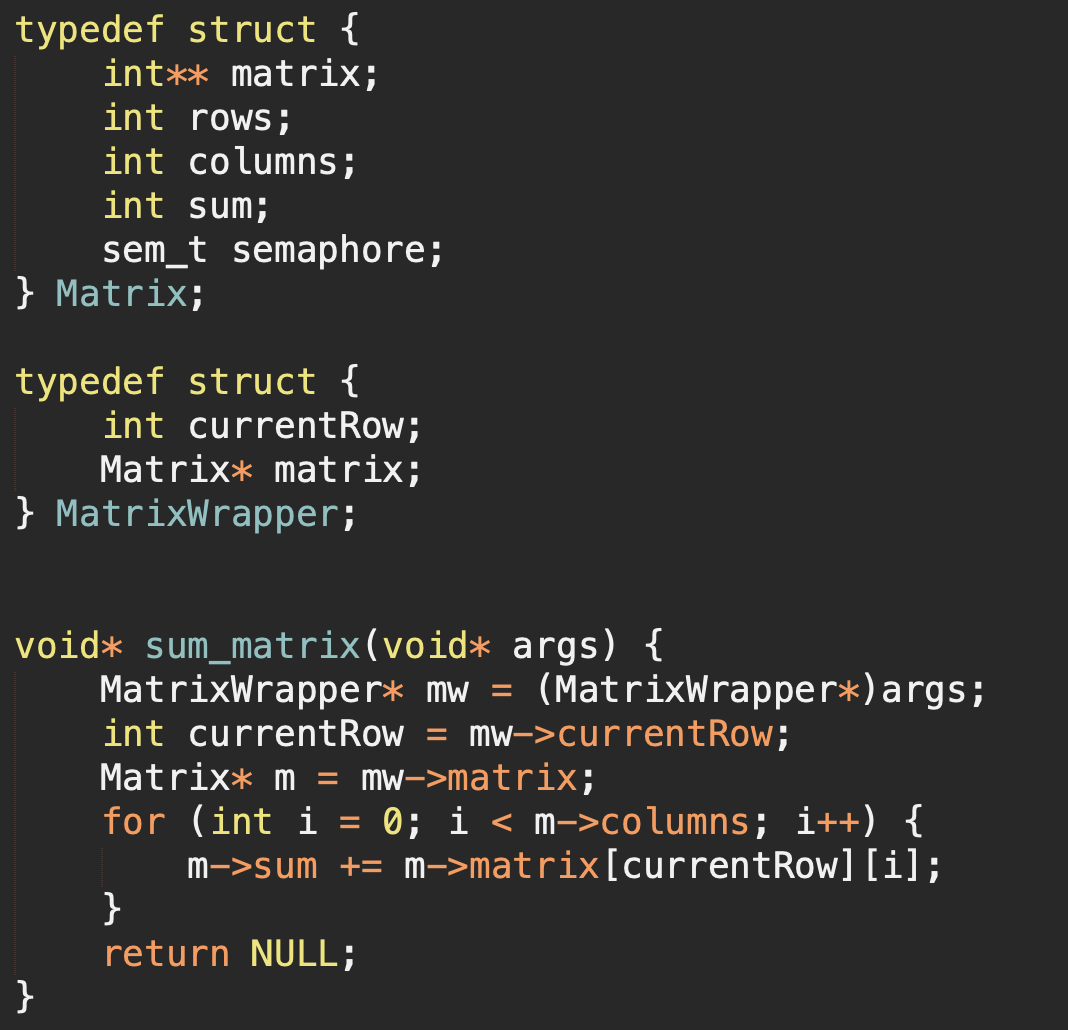
}

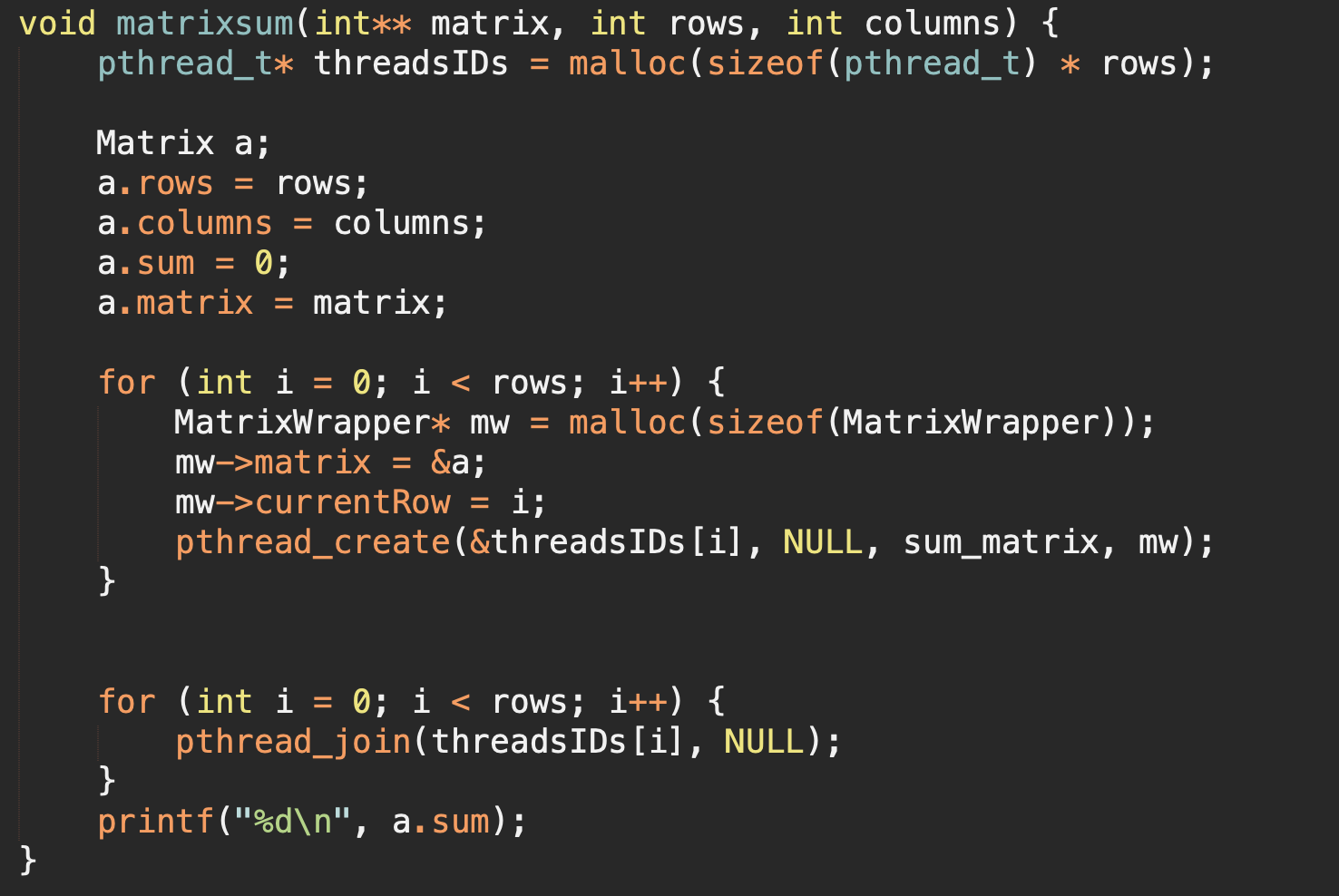
return sum;

}

This is the best one.

Compiled and works





There are no semaphores because MACOS hates semaphores >:(

The semaphores go around the m->sum +=... line which are just sem\_wait and sem\_post respectively

The other solutions were a bit confusing for me, i think this works

