# **CSSE2310: 2015 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.c**

### 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.

**Q1**

A) Show the first line of the file ferret

cat ferret | head -n 1

B) Show the directories which will be searched for commands to run.

echo $PATH

C) Show the first column of the file ferret (columns are separated by commas).

cat ferret | cut -d ‘,’ -f 1

D) Show all instances of vim currently running on the system.

ps -e | grep "vim" | tr -s ' ' | cut -d ' ' -f 5

# Or

pgrep -l vim

E) Count how many files in the current directory have names containing at least 5 characters

Let count=0

For file in $(ls)

Do

if [${#file} -ge 5]

((count++))

Fi

Done

Echo $count

Alternative: ls -ad ?????\* | grep -v ^d | wc -l

**Q2**

A) An integer variable which will not store negative values.

unsigned int foo;

B) A pointer to an arbitrary type.

void \*foo; ~~(or maybe void \*\*foo??)~~

C) An array of 12 true/false values.

(assuming #include stdbool)

bool foo[12];

D) A pointer to a function which takes an array of strings and returns a string.

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

E) A pointer to a function which takes two parameters (of the same type as foo from Part D) and returns a function pointer (also the same type as from Part D).

char \*(\*(\*foo)(char \*(\*foo) (char \*[]), char \*(\*foo) (char \*[])))(char \*[])

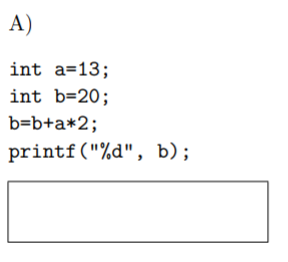
Alternative:

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

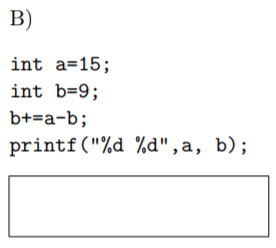
(char\*) (\* (\*foo)(bar, bar))(char\*\*);

**Q3**

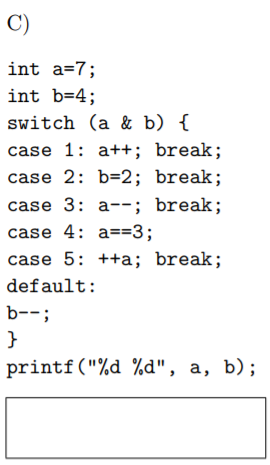
**What is the output from the following statements?**

****

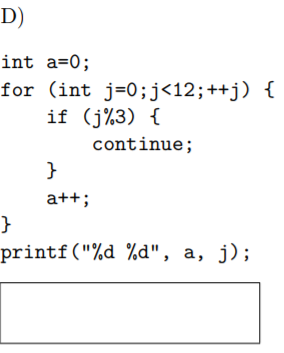
46



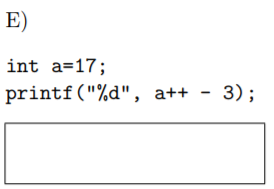
15 15



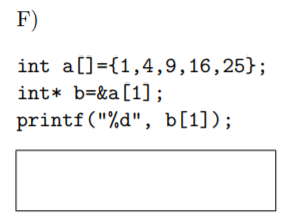
8 4



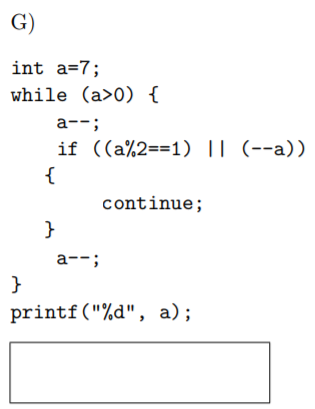
Error, j is only scoped in the loop



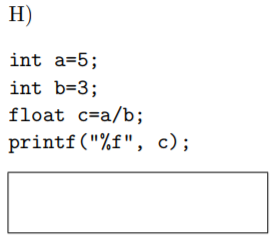
14



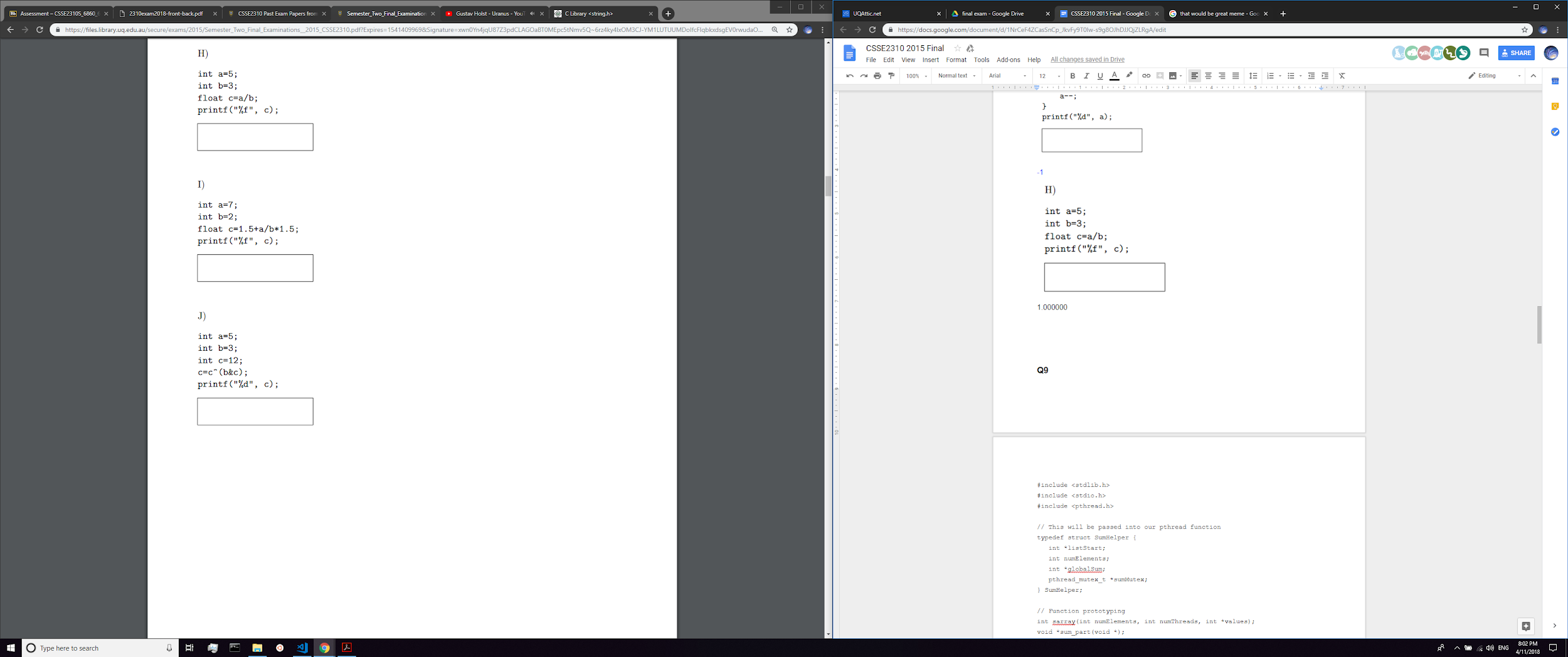
9



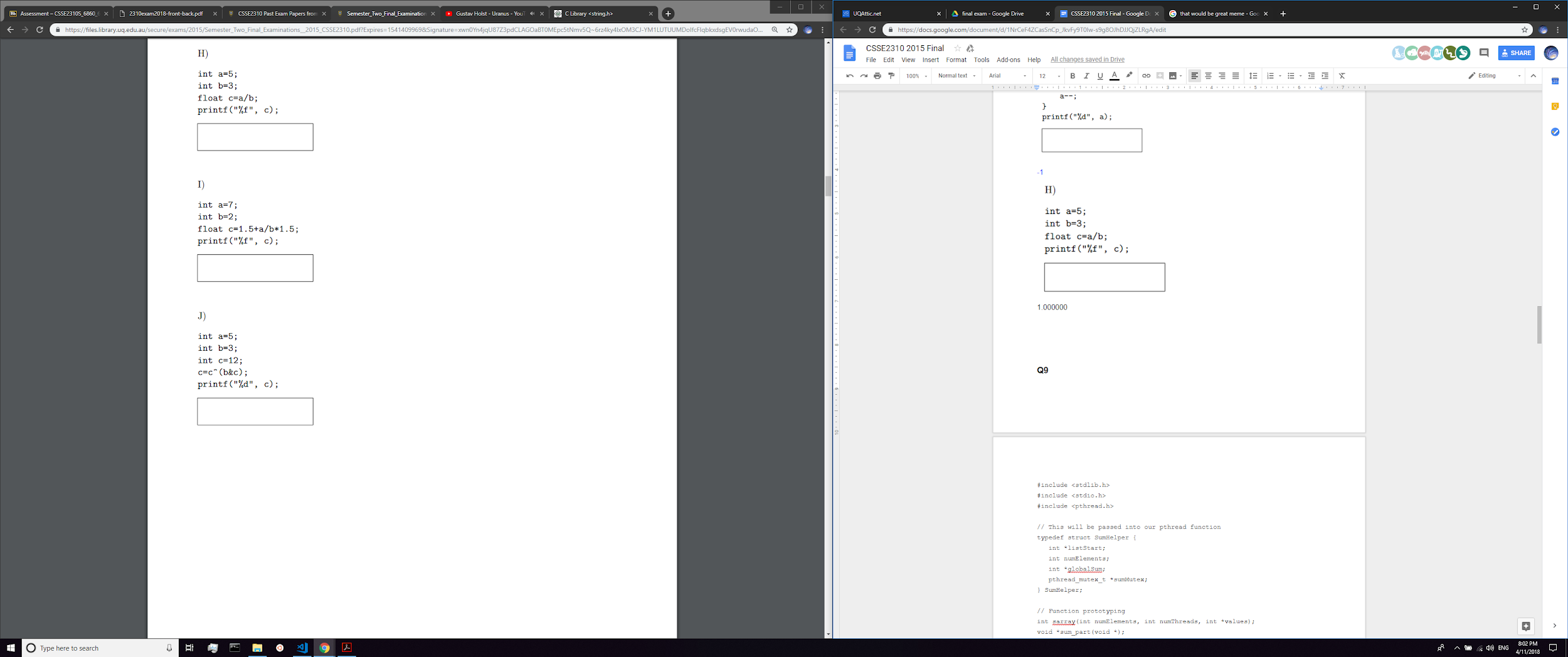
-1



1.000000



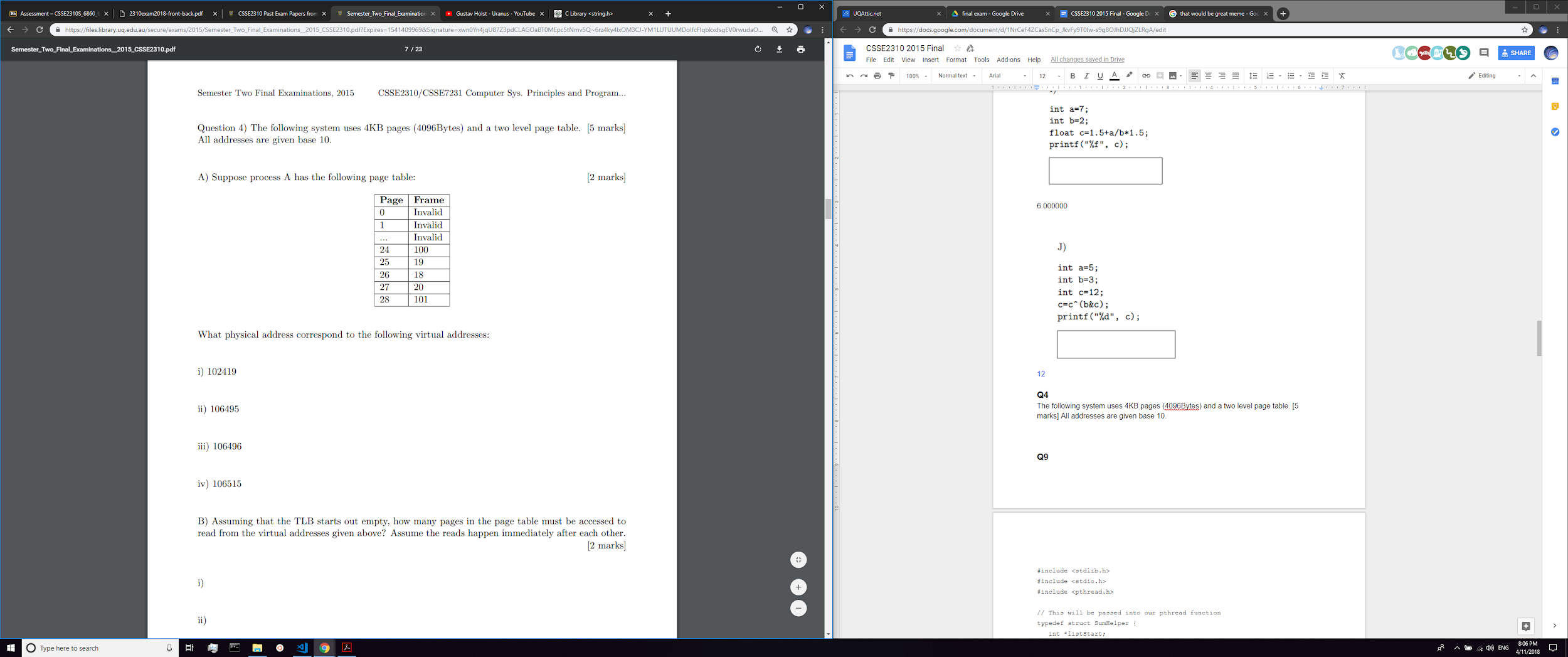
6.000000



12

**Q4**

The following system uses 4KB pages (4096Bytes) and a two level page table. [5 marks] All addresses are given base 10.



i) 102419

102419 = 25\*4096 + 19

(use 25 to map to 19)

⇒ 19\*4096 + 19 = 77843

ii) 106495

81919

iii) 106496

73728

iv) 106515

73747

B) Assuming that the TLB starts out empty, how many pages in the page table must be accessed to read from the virtual addresses given above? Assume the reads happen immediately after each other.

i)

3

ii)

1

^Should this be 0 if a TLB access doesn't count as a page table access? Awaiting confirmation on piazza

iii)

3

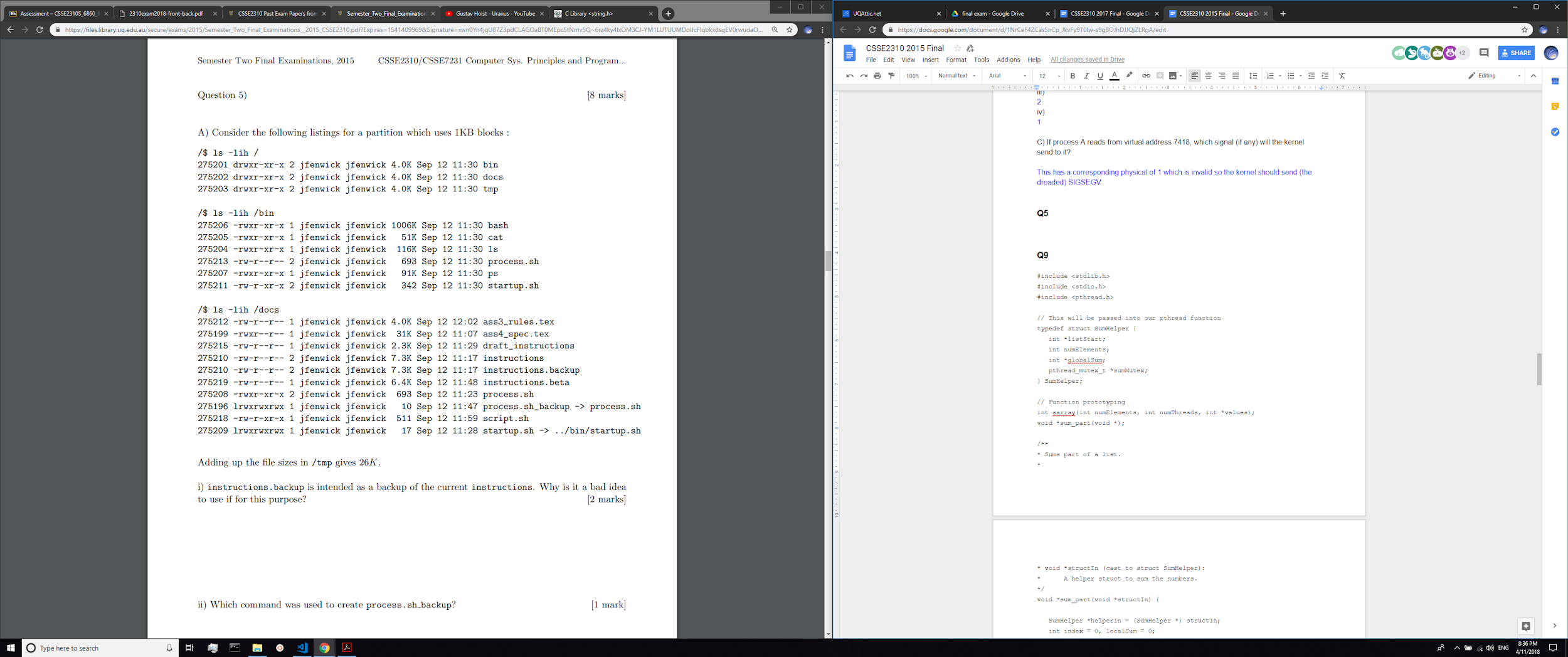
iv)

1

C) If process A reads from virtual address 7418, which signal (if any) will the kernel send to it?

This has a corresponding physical of 1 which is invalid so the kernel should send (the dreaded) SIGSEGV (11)

**Q5**



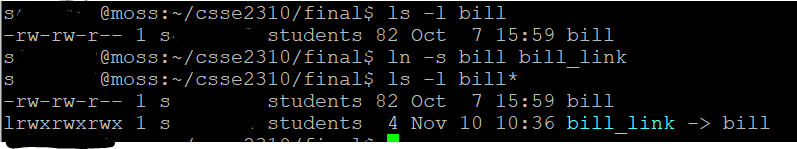
i) instructions.backup is intended as a backup of the current instructions. Why is it a bad idea to use if for this purpose?

They have been hard linked to each other so any corruption in the original will also appear in the back up

Also if the folder is deleted then all references to the file will be deleted?

ii) Which command was used to create process.sh backup?

ln -s process.sh process.sh\_backup **+2**

****

iii) Which files can jfenwick execute in the /docs directory?

Ass4\_spec.tex, process.sh , process.sh\_backup, startup.sh

I disagree --->

*ass4\_spec.tex, process.sh, process.sh\_backup*

* startup.sh can’t be run because if you follow the symlink the file is not executable

iv) Suppose rm -rf /tmp/\* is executed by the administrator (root). How much additional diskspace will be available? Explain your answer.

**Undetermined**. As the block size is 1K, there may be internal fragmentation for some of the files.

E.g. temp has 2 files:

File A (1.5 K)

File B (24.5K)

If we delete both file A & B, we would have freed 28K of space/blocks

Also note that hidden files will not be deleted. Therefore some files will not be removed and the corresponding disk space will not be freed

^^ idk but hopefully it kinda makes sense?

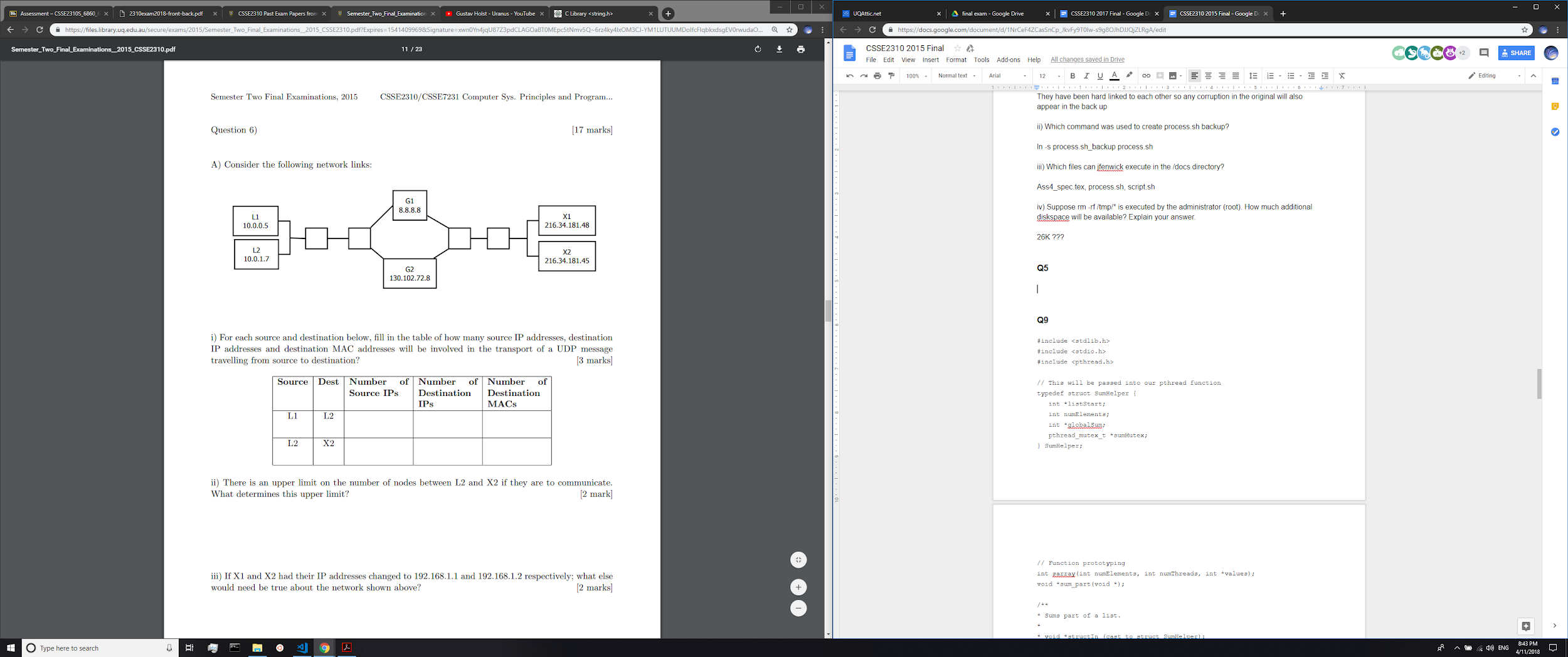
From Joel on piazza:

Not 100% certain but the image might be missing some details.

From vague memory:

1. You know about all files on the partition except those in /tmp or in a sub-directory of /tmp
2. You can look at the files you can see and determine how many of them must have another hardlink somewhere under /tmp.
3. From #2 you can work out how much space will be kept from under /tmp vs how much will be freed

**Q6**



i) For each source and destination below, fill in the table of how many source IP addresses, destination IP addresses and destination MAC addresses will be involved in the transport of a UDP message travelling from source to destination?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Source** | **Dest** | **Num Source IPs** | **Num Dest IPs** | **Num Dest MACS** |
| L1 | L2 | 1 | 1 | 1 (2??) |
| L2 | X2 | 2 | 1 | 6 |

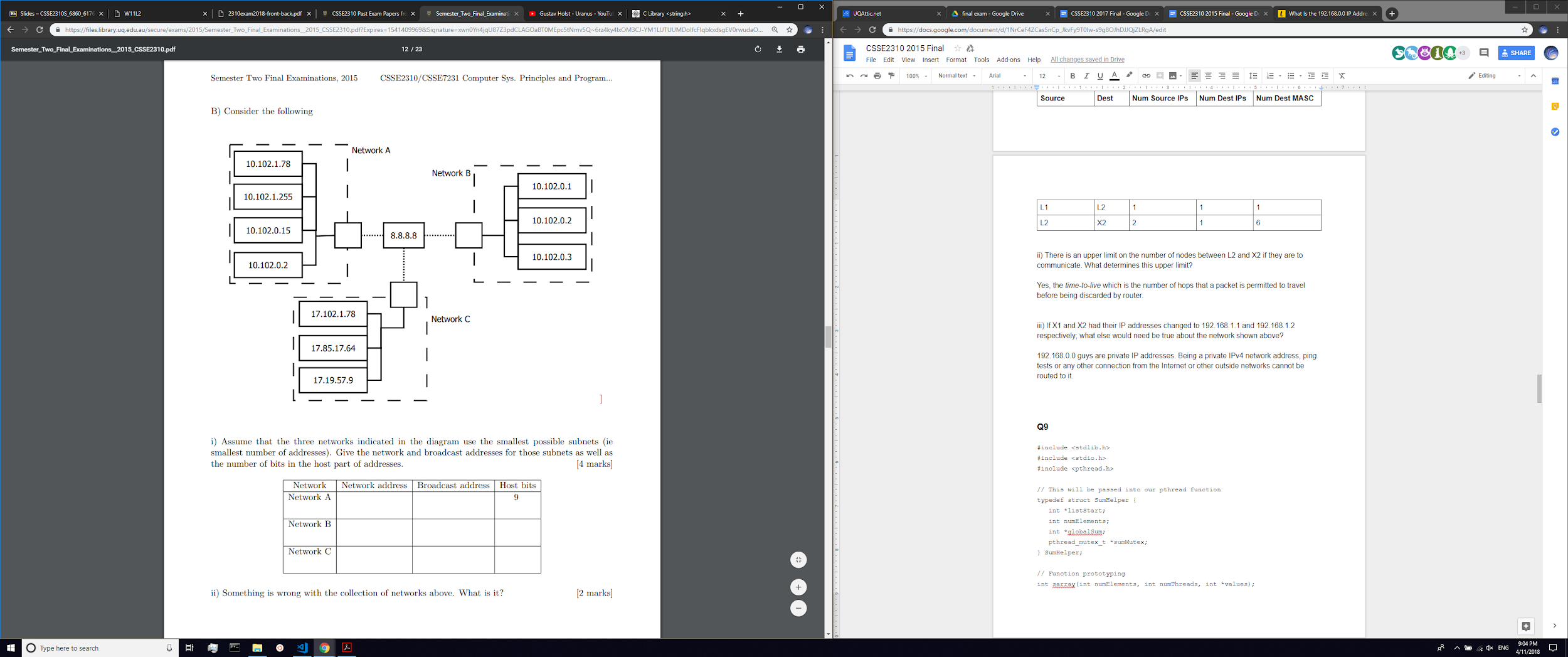
ii) There is an upper limit on the number of nodes between L2 and X2 if they are to communicate. What determines this upper limit?

Yes, the *time-to-live* which is the number of hops that a packet is permitted to travel before being discarded.

iii) If X1 and X2 had their IP addresses changed to 192.168.1.1 and 192.168.1.2 respectively; what else would need be true about the network shown above?

192.168.0.0 guys are private IP addresses. Being a private IPv4 network address, ping tests or any other connection from the Internet or other outside networks cannot be routed to it.

192.168.0.0 are private IP addresses. Therefore there will needed to be an additional router to NAT these addresses if X1 and X2 want to communicate to the internet.



i) Assume that the three networks indicated in the diagram use the smallest possible subnets (ie smallest number of addresses). Give the network and broadcast addresses for those subnets as well as the number of bits in the host part of addresses.

|  |  |  |  |
| --- | --- | --- | --- |
| Network | Network address | Broadcast address | Host bits |
| Network A | 10.102.0.0 | 10.102.1.255 | 9 |
| Network B | 10.102.0.0 | 10.102.0.7 | 3 ~~2?~~ |
| Network C | 17.0.0.0 | 17.127.255.255 | 23 |

ii) Something is wrong with the collection of networks above. What is it?

Yes there are machines in networks A and B whose IP addresses are the same IPs address as the Broadcast address

Just to be specific: on network A: IP address 10.102.1.255 (the second machine down) will be the same as the broadcast address

Nothing is wrong with network B I’m pretty sure +1 -1

If you work out B the same as network A (where the 9 host bits makes you use the broadcast address) then you have the same issue. Also the network address is the same for both A and B which is an issue because these networks are attached to a DNS server which won’t know where to send the packets if someone asks for 10.102.0.0.

C) A student’s server has crashed and they quickly restart it, however the server is unable to use the port they previously used. What have they forgotten to do?

Forgot to use the setsockopt() function with SO\_REUSEADDR as an option

D) Which layers of the network stack do the following belong to?

|  |  |
| --- | --- |
| **Term** | **Layer** |
| Chrome and Safari | Application (Layer 5) |
| SCP | Application (Layer 5) |
| MAC | Link (2) / Network Layer (Layer 3) |
| Ports | Transport (Layer 4) |

**Q7**

Consider a “unix” filesystem where: [6 marks]

• All i-nodes are cached in RAM

• i-nodes have 10-direct pointers, 2 indirect pointers and 3 double indirect pointers.

• Blocks are 8KB

• Block pointers are 16Bytes

• blocks are numbered from 0

A) What is the maximum possible file size for this file system?

10/2/3

Blk ⇒ 2^3 \* 2^10 = 2^13 Bytes

Blk Ptr ⇒ 2^4

Num Ptrs in Blk = 2^13 / 2^4 = 2^9

Max = 10 \* (2^13) + 2 \* (2^9 \* 2^13) + 3 \* (2^9 \* 2^9 \* 2^13)

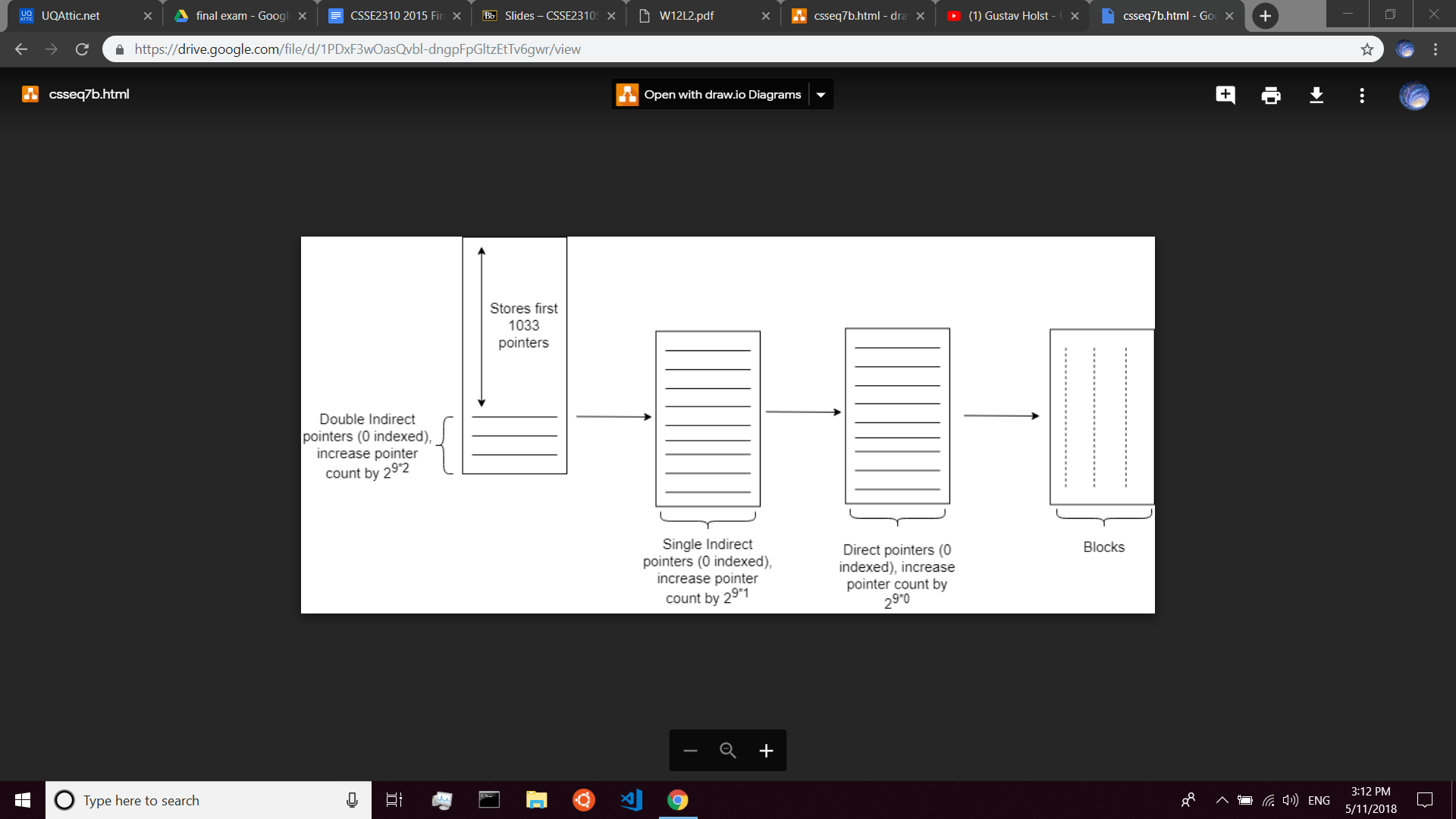
= 6 450 921 472 B

= 6 299 728 KiB

B) How many blocks (in total) must be accessed to read the following blocks from a file: 0, 1, 2, 4012, 8009

Link to picture

https://drive.google.com/file/d/1PDxF3wOasQvbl-dngpFpGltzEtTv6gwr/view?usp=sharing



Block 0 ⇒ Direct block 0

Block 1 ⇒ Direct block 1

Block 2 ⇒ Direct block 2

Block 4012

We want to find:

in the equation

Where represent respective the double indirect, single indirect and direct pointer count. We want to use the largest (non-negative) values for . Clearly since increases the pointer count by . So

Largest value for is

Meaning

Thus

Double Indirect [0] -> Single Indirect [5] -> Direct [418] ⇒ Block 4012

Similar reasoning

Double Indirect [0] -> Single Indirect [13] -> Direct [350] ⇒ Block 8009

Total amount of blocks accessed = 1+1+1+3+2 = 8

C) What is the number of the first block in a large file which needs to use the second double indirect pointer?

I got 263178.[+3]

The first double indirect pointer contains pointers to a total of block #s:

1034 to 263177 (where 1034 + 512^2 - 1 = 263177)

The second double indirect pointer contains pointers to a total of block #s:

263178 to … doesn’t matter

Would it be 263179 since 263178 is the last pointer for the first indirect???

Nope, 263178 is the first pointer for the second double indirect and 263177 is the last pointer for the first double indirect.

**Q8**

Consider the following code which process A executes:

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

pid\_t me=getpid();

for (int i=0;i<4;++i) {

fork();

}

// line X

int s;

if (getpid()==me) {

while(wait(&s)>=0) {}

}

exit(0);

}

A) When process A reaches line X, how **many** children does A have?

Question asks how many children… not how many new processes. A has 4 children

B) How many processes are created overall?

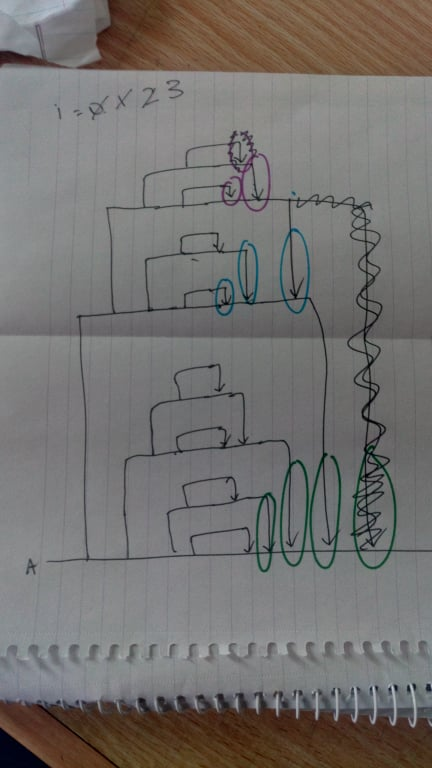
A is not created. It is 15 new processes: 24 - 1

C) Draw a fork/process diagram showing the lifetimes of the processes above?

Indeterminate? (Since we’re using wait instead of waitpid)

It is worth 4 marks might be a good idea to draw something. Also it is only indeterminate to an extent. coz wait() will wait for finished children

Colour coded circles on which ones can have order interchanged. (didn’t circle all of them)



**Q9: person 1**

#include <stdlib.h>

#include <stdio.h>

#include <pthread.h>

// This will be passed into our pthread function

typedef struct SumHelper {

int \*listStart;

int numElements;

int \*globalSum;

pthread\_mutex\_t \*sumMutex;

} SumHelper;

// Function prototyping

int sarray(int numElements, int numThreads, int \*values);

void \*sum\_part(void \*);

/\*\*

\* Sums part of a list.

\*

\* void \*structIn (cast to struct SumHelper):

\* A helper struct to sum the numbers.

\*/

void \*sum\_part(void \*structIn) {

SumHelper \*helperIn = (SumHelper \*) structIn;

int index = 0, localSum = 0;

for (index = 0; index < helperIn->numElements; index++) {

localSum += helperIn->listStart[index];

}

pthread\_mutex\_lock(helperIn->sumMutex);

\*(helperIn->globalSum) += localSum;

pthread\_mutex\_unlock(helperIn->sumMutex);

free(helperIn);

return NULL;

}

/\*\*

\* Sum the first few elements of the array. The sum is returned as an integer.

\*

\* int numElements:

\* The number of the elements in our list that we would like to sum.

\*

\* int numThreads:

\* The number of threads used to sum the list.

\*

\* int \*values:

\* The list containing the values we want to sum.

\*

\* return:

\* The final value of our sum.

\*/

int sarray(int numElements, int numThreads, int \*values) {

int sum = 0, index = 0, perThread = numElements/numThreads;

pthread\_mutex\_t sumMutex;

pthread\_mutex\_init(&sumMutex, NULL);

pthread\_t \*threadList = (pthread\_t \*) calloc(numThreads, sizeof(pthread\_t));

for (index = 0; index < numThreads; index++) {

SumHelper \*helper = (SumHelper \*) calloc(1, sizeof(SumHelper));

// Specify how many numbers this threads needs to sum.

if ((index + 1) == numThreads) {

helper->numElements = numElements - (index \* perThread);

} else {

helper->numElements = perThread;

}

// Adjust the start of the list for each thread

helper->listStart = values + (index \* perThread);

helper->globalSum = &sum;

helper->sumMutex = &sumMutex;

pthread\_create(threadList + index, NULL, sum\_part, (void \*) helper);

}

// Join all the threads

for (index = 0; index < numThreads; index++) {

pthread\_join(threadList[index], NULL);

}

free(threadList);

return sum;

}

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

int index = 0, \*numbers = (int \*) calloc(100, sizeof(int));

// Populate an array with number from 1 to 100

for (index = 0; index < 100; index++) {

numbers[index] = index + 1;

}

// Only sum the first 50

int sum50 = sarray(50, 3, numbers);

printf("Sum is: %d", sum50);

free(numbers);

return 0;

}

**Q9 Person 2: // tested this and works :)**

struct Job {

int start;

int stop;

int sum;

int max;

int\* v;

pthread\_t tid;

};

void\* threadfunc(void\* thing) {

struct Job\* job = (struct Job\*) thing;

for (int i = job->start; (i < job->stop && i < job->max); i++) {

job->sum += job->v[i];

}

}

int sarray(int n, int t, int\* values) {

int total = 0, length = (n + t - 1) / t;

struct Job\*\* jobs;

jobs = (struct Job\*\*) malloc(sizeof(struct Job\*) \* t);

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

struct Job\* job = (struct Job\*) malloc(sizeof(struct Job));

job->v = values;

job->start = i \* length;

job->stop = (i + 1) \* length;

job->sum = 0;

job->max = n;

pthread\_create(&job->tid, NULL, threadfunc, (void\*) job);

jobs[i] = job;

}

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

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

total += jobs[i]->sum;

free(jobs[i]);

}

free(jobs);

return total;

}

**Q10**

**A/B**

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

#include <pthread.h>

// This will be passed into our pthread function

typedef struct TopsHelper {

char \*\*linesExtracted;

FILE \*fileIn;

int numLines;

} TopsHelper;

int strcmp\_wrapper(const void \*dataOne, const void \*dataTwo) {

char \*strOne = \*(char \*\*) dataOne;

char \*strTwo = \*(char \*\*) dataTwo;

return strcmp(strOne, strTwo);

}

char \*\*load\_and\_sort(FILE \*fileIn) {

int numLines = 0;

int linePos = 0;

char \*\*stringList = (char \*\*) calloc(numLines + 1, sizeof(char \*));

stringList[0] = (char \*) calloc(80, sizeof(char));

char byteIn = 0;

while ((byteIn = fgetc(fileIn)) != EOF) {

if (byteIn == '\n') {

numLines++;

linePos = 0;

stringList = (char \*\*) realloc(stringList, sizeof(char \*) \*

(numLines + 1));

stringList[numLines] = (char \*) calloc(80, sizeof(char));

} else {

stringList[numLines][linePos] = byteIn;

linePos++;

}

}

// numLines is 0 indexed, which is why we need to +1 it here

qsort(stringList, (numLines + 1), sizeof(char \*), strcmp\_wrapper);

return stringList;

}

void \*get\_first\_n(void \*voidHelper) {

TopsHelper \*helper = (TopsHelper \*) voidHelper;

// First get the number of line in this file

helper->numLines = 0;

char byteIn = 0;

while ((byteIn = fgetc(helper->fileIn)) != EOF) {

if (byteIn == '\n') {

helper->numLines++;

}

}

helper->numLines++;

rewind(helper->fileIn);

// Now use of load and sort function

helper->linesExtracted = load\_and\_sort(helper->fileIn);

return NULL;

}

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

int index = 0, threadNum = 0, numLines = atoi(argv[1]);

// Make a list of tops helpers to pass into our thread creates.

TopsHelper \*helperList = (TopsHelper \*) calloc(argc - 2, sizeof(TopsHelper));

pthread\_t \*threadList = (pthread\_t \*) calloc(argc - 2, sizeof(pthread\_t));

for (index = 0; index < (argc - 2); index++) {

helperList[index].fileIn = fopen(argv[index + 2], "r");

helperList[index].linesExtracted = NULL;

pthread\_create(threadList + index, NULL, get\_first\_n,

(void \*) (helperList + index));

}

// Now join the threads

for (index = 0; index < (argc - 2); index++) {

pthread\_join(threadList[index], NULL);

fclose(helperList[index].fileIn);

}

// Now print our each of the lines collected

for (threadNum = 0; threadNum < (argc - 2); threadNum++) {

// Now print out each of the lines collected by each thread.

for (index = 0; index < ((numLines < helperList[threadNum].numLines) ?

numLines : helperList[threadNum].numLines); index++) {

printf("%s\n", helperList[threadNum].linesExtracted[index]);

}

}

free(helperList);

free(threadList);

exit(0);

}

**C/D**

#include <stdio.h>

#include <unistd.h>

#include <err.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/types.h>

#include <sys/wait.h>

#include <fcntl.h>

/\*\*

\* Enumerates reading and writing for pipes

\*/

enum ReadAndWrite {

READ = 0,

WRITE = 1

};

void ptops(int n, char \*filename) {

// Create our ls string, first find how long the string will be

int length = snprintf(NULL, 0, "-n %d", n);

// Now actually create the string

char \*headArg = calloc(length, sizeof(char));

sprintf(headArg, "-n %d", n);

// Prepare the commands with their respective arguments

char \*sort[] = { "sort", filename, NULL };

char \*head[] = { "head", headArg, NULL };

// Set up a list for our pipe

int conn[2];

pid\_t child\_id;

int status;

if (pipe(conn) == -1)

err(1,"pipe");

// First perform the sort function

if((child\_id = fork()) == 0) {

// We dont need the reading end

close(conn[READ]);

// Child process for ls

if(dup2(conn[WRITE], STDOUT\_FILENO) == -1)

err(1,"dup2 child");

// Close write pipe since it now dupped to STDOUT

close(conn[WRITE]);

execvp(sort[0], sort);

err(1, "exec child");

} else {

// parent process

// Wait for child to stop

close(conn[WRITE]);

waitpid(child\_id, &status, 0);

}

if((child\_id = fork()) == 0) {

// Child process for grep

if(dup2(conn[READ], STDIN\_FILENO) == -1)

err(1,"dup2 child grep");

close(conn[READ]);

execvp(head[0], head);

err(1, "exec child grep");

} else {

// parent process

// Wait for all children to stop

close(conn[READ]);

waitpid(child\_id, &status, 0);

}

}

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

int index = 0;

int numLines = atoi(argv[1]);

for (index = 0; index < (argc - 2); index++) {

ptops(numLines, argv[2 + index]);

}

return 0;

}