THE UNIVERSITY OF QUEENSLAND
AUSTRALIA

This exam paper must not be removed from the venue

Venue ____________________

Seat Number ________

Student Number |__|__|__|__|__|__|__|__|

Family Name ______________________

First Name ______________________

School of Information Technology and Electrical Engineering

EXAMINATION

Semester Two Final Examinations, 2018

CSSE2310/CSSE7231 Computer Systems Principles & Programming

This paper is for St Lucia Campus students.

Examination Duration: 180 minutes

Reading Time: 10 minutes

Exam Conditions:

This is a Central Examination

This is a Closed Book Examination - specified materials permitted

During reading time - write only on the rough paper provided

This examination paper will be released to the Library

Materials Permitted In The Exam Venue:

(No electronic aids are permitted e.g. laptops, phones)

Calculators - Casio FX82 series or UQ approved (labelled)

One A4 sheet of handwritten or typed notes double sided is permitted

Materials To Be Supplied To Students:

None

Instructions To Students:

Additional exam materials (eg. answer booklets, rough paper) will be provided upon request.

Attempt all 11 questions. Answer on the question paper. Notes pages to be collected with question paper. Answer booklets can be used for additional answer space.

100 total marks. Some useful data is provided on page 27. All numbers are assumed to be written in base 10 unless otherwise specified.

If you received an approved extension, as per the course ECP, the relevant assignment will link to the specific exam questions as per below.
Assignment 1: Q3, Q9A, Q9D totalling 16 marks
Assignment 3: Q8, Q9B, Q9C totalling 19 marks
Assignment 4: Q6E, Q10, Q11 totalling 19 marks

For Examiner Use Only

Question	Mark

Total ________

Question 1) Write C code to declare `foo` as: [**5 marks (1 each)**]

A) A non-negative integer.

```
unsigned int foo;
```

B) An array of 18 long integers.

```
long foo[18];
```

C) A data type that holds either a string called `str` or a long integer called `length`.

```
union foo{
        char* str;
        long length;
};
```

D) A pointer to a function which takes as parameters an integer and a double, and returns nothing.

```
void (*foo)(int, double)
```

E) A pointer to a function which takes an array of strings as a parameter, and returns a pointer to a function which takes a constant string and returns an int.

```
int (*(*foo)(char*[]))(const char*)

or
typedef int (*)(const char*) bar;
bar (*foo)(char*[])
```

Question 2) Write shell commands to do the following: [**10 marks**]

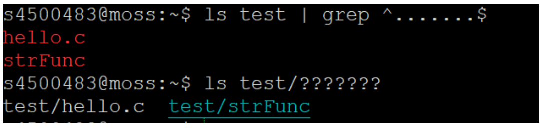
A) Show the names and permission information for all files in the **parent** directory. [1 mark]

cd ..
ls -la

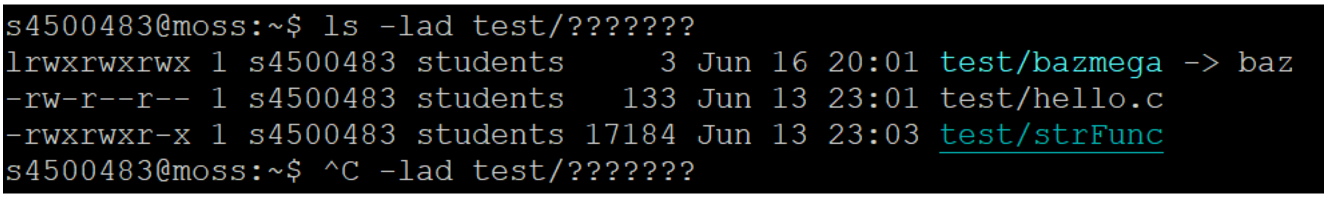
OR just
ls -la ..

B) Show the names of all files from the `verbose` directory with names exactly 7 characters long. [1 mark]

ls -lad test/???????



```
s4500483@moss:~$ ls test | grep ^.......$
hello.c
strFunc
s4500483@moss:~$ ls test/???????
test/hello.c  test/strFunc
```



```
s4500483@moss:~$ ls -lad test/???????
lrwxrwxrwx 1 s4500483 students     3 Jun 16 20:01 test/bazmega -> baz
-rw-r--r-- 1 s4500483 students   133 Jun 13 23:01 test/hello.c
-rwxrwxr-x 1 s4500483 students 17184 Jun 13 23:03 test/strFunc
s4500483@moss:~$ ^C -lad test/???????
```

C) Append your username to the file `goose` [1 mark]

echo $USER >> goose

D) Count the number of processes **you** are running with the name `banzai` [2 marks]

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

or
ps -u $USER | grep bash -c

E) The current svn working directory contains a file `utils.h`. Rename the file on disk and in svn to `utils.c` [1 marks]

```
mv utils.h utils.c
svn rm utils.h
svn add utils.c
svn update
svn commit

or
svn mv utils.h utils.c
svn commit
```

F) There is a file called `ids`, where each line contains an ID number followed by a comment, separated by a single tab character. Store a list of the unique ID numbers from `ids` into a file called `uids`. [2 marks]

```
cut -f1 ids | sort | uniq > uids
```

G) Copy all files in the current directory to a new directory `../backup` and add `.bak` to the end of their name. eg: `A.pdf` would be copied to `../backup/A.pdf.bak` [2 marks]

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

Question 3) Give the output for the following code fragments. [**6 marks (1 each)**]
[Assignment 1: 6/16 marks]

A)

```
int a = 15;
int b = 6;
printf("%d\n", a ^ b & b);
```

B)

```
int a = 1;
int b = -1;
printf("%d\n", a && b);
```

C)

```
int a = 5;
double b = 61;
printf("%d\n", (int) b / a);
```

D)

```
int i, x = 4, y = 6;
for (i = 1; i < y, --x; i += x)
    ++y;
    ++x;
printf("%d %d %d\n", i, x, y);
```

E)

```
int a = 4, b = 3, c;
switch (a %= b) {
    case 2: c = 2; break;
    case 3: c = 1; break;
    case 1: c = 3;
    case 0: c = 4;
    default: c = 5;
}
printf("%d\n", c);
```

F)

```
bool b = false, o = true, w;
if (b && o || (o = true)) {
    w = true;
} else {
    w = false;
}
if (b)
    printf("b");
if (o)
    printf("o");
if (w)
    printf("w");
```

Question 4) Consider the following directory listing. Unless otherwise specified, all commands in this question are executed in this directory. [**6 marks**]

```
ls -ali
total 267200
284179 drwxrwxr-x  4 tman yaks      4096 Oct 13 23:03 .
262177 drwxr-xr-x 37 boot boot      4096 Oct 13 22:54 ..
287449 -rw----r--  2 tman yaks     19183 Oct 13 22:56 bananas
287449 -rw----r--  2 tman yaks     19183 Oct 13 22:56 bpineapple
287569 -rw-rw-r--  1 tman yaks         0 Oct 13 23:02 empty
287447 lrwxrwxrwx  1 tman yaks        17 Oct 13 22:56 flamboyant -> /usr/games/lolcat
287570 -rw-rw-r--  1 tman yaks 273547264 Oct 13 23:02 full
287450 drwxr-xr-x  4 term bots      4096 Oct 13 22:58 include
287448 --wx-w---x  1 tman yaks       194 Oct 13 22:55 shtick.sh
287571 drwxrwxr-x 14 mike yaks      4096 Oct 13 23:03 thonk
```

Groups have the following membership:

Group	Members
yaks	tman, greg, furball
bots	term, wally, atlas, smolpup
others	jim, mike, jan, lking

A) Which files and/or folders can greg read? [1 mark]

. .. empty, full, include, thonk
flamboyant: symbolic link, the permission to
access this file depends on the file it points to

B) How many subdirectories does the thonk folder contain? [1 mark]

14 - 2 = 12
the link count for a directory is 2 plus the number of direct
subdirectories it has.

C) Suppose that the above filesystem is full (and no space is reserved), and then rm full is executed. What is the largest file which could then be created on the filesystem (as a regular user)? [2 marks]

273547264 bytes
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.

D) When tman attempts to run ./shtick.sh, it fails. Explain why this is the case, and give a shell command that would fix the problem. [2 marks]

Tman is the owner, so the "u" option applies. For .sh (script or interpreted language files) files, you need to be able to read the file to read what commands to run.

require chmod u+r shticsk.sh

Question 5) Consider a system which uses a two level page table. [**6 marks**]
It has the following characteristics:

- 32 KiB pages,
- 32 bit virtual addresses,
- 36 bit physical addresses,
- 128 Byte page table entries.

pagesize: 32*1024 = 32768
page = vir/32768
offset = vir%32768
phy = frame*32768 + offset

All numbers given in this question are base 10.
A process has the following (complete) page table:

Frame	Page
708	15
52	16
51	44
81	62
70	64
71	70
72	71
90	90
91	91
62	510
15	511
64	512

A) What physical address corresponds to the following virtual addresses: [2 marks]

2 342 963	2375731
16 777 215	524287
16 777 216	2097152
2 326 600	2359368

page = 2342963/32768 = 71.50 = 71
offset = 2342963%32768 = 16435
phy = 72*32768 + 16435 = 2375731

page = 16777215/32768 = 511.99 = 511
offset = 16777215%32768 = 32767
phy = 15*32768 + 32767 = 524287

page = 16777216/32768 = 512 = 512
offset = 16777216%32768 = 0
phy = 64*32768 + 0 = 2097152

page = 2326600/32768 = 71.00219 = 71
offset = 2326600%32768 = 72
phy = 72*32768 + 72 = 2359368

B) The process performs a read operation on its memory address 1 443 176. What signal (if any) will the process receive from the kernel? [1 mark]

page = 1443176/32768 = 44.0422363281 = 44

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

C) How many accesses to physical memory are required to read the following virtual addresses in sequence? Assume the TLB starts out empty. [2 marks]

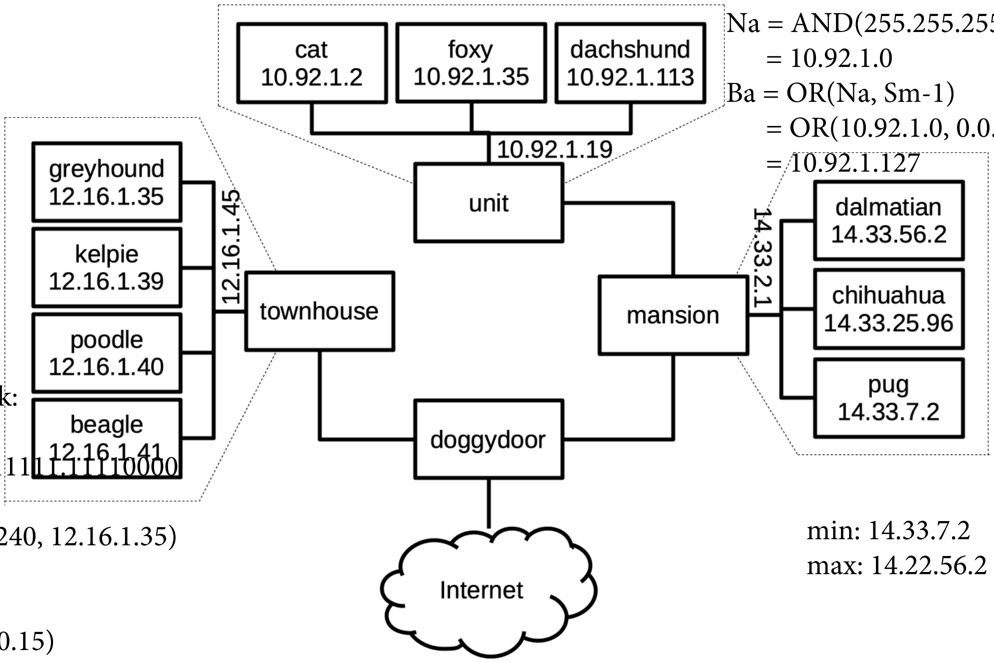
转译后备缓冲器 (Translation lookaside buffer)

2 342 963	3	page: 71
16 777 215	3	page: 511
16 777 216	3	page: 512
2 326 600	1	page: 71

Question 6) [17 marks]

min: 10.92.1.2
max: 10.92.1.113
2: 00000010
113: 01110001
length of the subnet mask: 24+1=25
11111111.11111111.11111111.10000000
Sm: 255.255.255.128
Na = AND(255.255.255.128, 10.92.1.2)
= 10.92.1.0
Ba = OR(Na, Sm-1)
= OR(10.92.1.0, 0.0.0.127)
= 10.92.1.127

A) Consider the following network for `catcorp.net`: [8 marks]



min: 12.16.1.35
max: 12.16.1.41
35: 00100011
41: 00101001
length of the subnet mask:
= 24+4=28
11111111.11111111.11111111.11110000
Sm: 255.255.255.240
Na = AND(255.255.255.240, 12.16.1.35)
= 12.16.1.32
Ba = OR(Na, Sm-1)
= OR(12.16.1.32, 0.0.0.15)
= 12.16.1.47

min: 14.33.7.2
max: 14.22.56.2

i) Complete the table for each part of the network (denoted by dotted lines). Use the smallest possible networks.

Network	Broadcast address	Netmask	CIDR network notation
unit	10.92.1.127	255.255.255.128	10.92.1.0/25
townhouse	12.16.1.47	255.255.255.240	12.16.1.32/28
mansion			

ii) Fill in the minimum number of addresses of each type used to send each message type from A to B.

Type	A	B	Destination MACs	Source IPs	Destination IPs
TCP segment (including ACK)	kelpie	pug	8	2	2
UDP datagram	kelpie	pug	4	1	1
Ethernet frame	poodle	foxy	5	0	0
TCP segment (including ACK)	beagle	uq.edu.au	*not required*	2	2

B) [1 mark]

i) Which layer(s) of the network stack are affected by a cable change from copper to a WiFi connection?

physical, possibly link (if different hardware)

ii) Name a reliable transport layer protocol.

TCP

C) An organisation owns the 4.51.16.0/21 block and they wish to break this range into networks, each containing 31 machines (hosts). What is the maximum number of such networks they could have? [1 mark]

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

D) Which layer of the network stack are each of the following terms primarily associated with? [3 marks]

TCP segment	transport
IP datagram	network
MAC address	link
HTTP Protocol	application
SSH protocol	application
Ethernet frame	link

E) There is a minimum number of calls to be made in order to act as a networking server, and similarly a client. List in order the minimum required calls for each. Choose from: `ssh, listen, scp, select, bind, fdopen, socket, close, write, accept, open, read, fprintf, system, fork, connect, pipe`. [4 marks]

[Assignment 4 cover: 4/19 marks]

Server

1. socket
2. bind
3. listen
4. accpet

Client

1. socket
2. connect

Question 7) **[6 marks]**

Consider a “unix” filesystem where:

- All i-nodes are cached in RAM.
- blocks are 4KiB, block pointers are 16bytes.
- i-nodes have 9 direct pointers, 2 indirect pointers and 2 double indirect pointers.

9 * 4096 + (2 * 256 * 4096) + (2 * 256^2 * 4096) = 539 004 928 bytes.

A) How many blocks would need to be accessed to (independently) read the following bytes from a file: [2 marks]

17 920	1
40 959	2
58 757 174	3
270 569 472	3

Space that direct pointers take => 9*4096 = 36,864 bytes
17920 < 36864

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

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) What is the maximum file size for this filesystem? [2 marks]
(Clearly state your units, and be precise.)

Pointers per block: 4096/16 = 256
Total number of blocks on the system: 9 + (2*256) + (2*256^2) = 131,593
Size = no. of pointers * block size = 131,593 * 4096 = 539,004,928 bytes or
539,004,928/ 2^10 = 526,372 KiB

alt: 4 KiB x (9 + 2*256 + 2*256^2) = 526,372 KiB

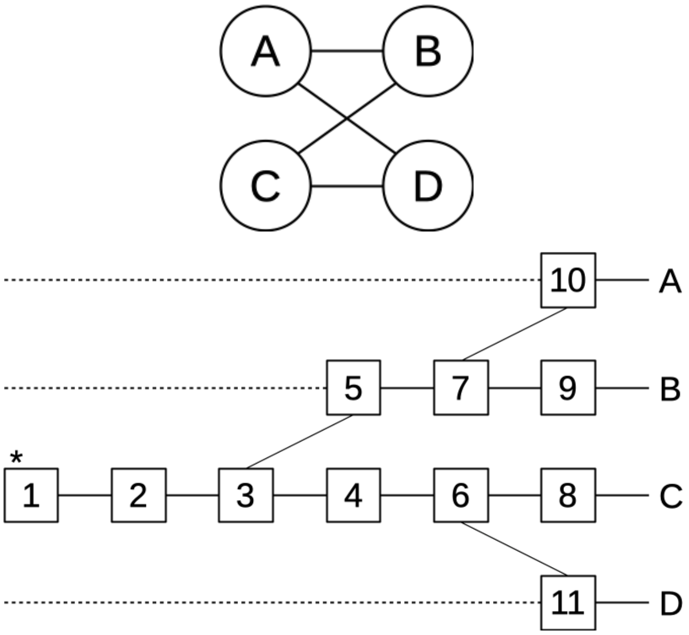
C) What would be the **change** in maximum file size if one double indirect pointer was changed to be a triple indirect pointer? [2 marks]
(Answer precisely in KiB.)

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]

Question 8) **[7 marks]**

[Assignment 3 cover: 7/19 marks]

A) Consider the following diagrams depicting pipes between processes, and a timeline of processes.



The top diagram shows 4 processes (running programs A, B, C, D) and pipes between them. The lower diagram (starting with process marked *) will create this arrangement (each row in the diagram represents a single process). Fill in the boxes to indicate which operations need to be performed when. Use operations from the following list (some may be needed more than once): [4 marks]

`kill(B), exec(A), exec(B), exec(C), exec(D), fork, pipe(AB), pipe(AD), pipe(CB), pipe(CD), exit, listen.`

1. pipe(CB)

2. pipe(AD)

3. fork()

4. pipe(CD)

5. pipe(AB)

6. fork()

7. fork()

8. exec(C)

9. exec(B)

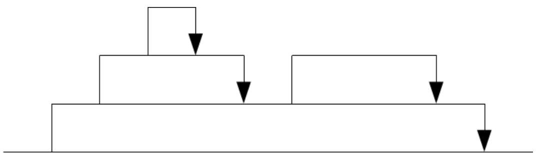
10. exec(A)

11. exec(D)

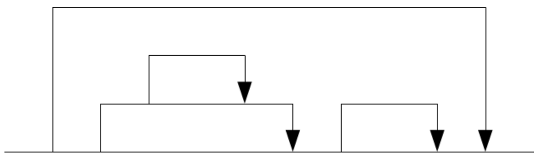
B) Write a shell command or C code to send a SIGINT to a process with pid= 517. [1 mark]

kill(517, 2); OR kill -2 517 OR kill -SIGINT 517

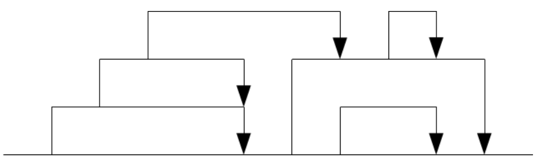
C) Identify whether each of the following concurrency diagrams is possible using processes and threads. Answer Y/N in the relevant boxes. [2 marks]



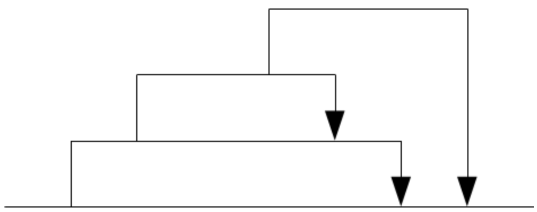
Process: Y Thread: Y



Process: Y Thread: Y



Process: N Thread: Y



Process: Y Thread: Y

Question 9) **[22 marks]**

Within this question, you may omit `#includes`. Where you are asked to write functions, you can also write helper functions if you need them. You may assume that any functions you write have been prototyped at the top of the file (so you don't need to worry about which order you write them in). You may assume that system calls succeed.

In later parts of this question, you may make use of functions described in earlier parts (and you may assume that they are implemented correctly).

A) [5 marks]
[Assignment 1 cover: 5/16 marks]

Write a **function** called `split_string`, which takes a string and returns an array of strings consisting of parts of the original string which were separated by one or more space characters. The final element in the returned array should be a null pointer. **For this question part, you must not call any standard string functions.**

For example: `split_string("here are    a-few   spaces")`
would return the array: `{"here", "are", "a-few", "spaces", NULL}`.

```
char** split_string(const char* line) {
```

B) [4 marks]
[Assignment 3 cover: 4/19 marks]

Write a **function** called `run_part`, which takes an array of strings representing `argv` and two `FILE*` called `a` and `b`. When the function is finished, a new process will be running with `a` as `stdin` and `b` as `stdout`. The new process should execute a program searching the path using the first string of `argv`, and have `argv` as its argument vector. You are not permitted to call `system()` or `popen()`.

You may make use of functions described in earlier parts (and you may assume that they are implemented correctly). You may assume that system calls succeed. The C library functions `int fileno(FILE* f)` and `FILE* fdopen(int fd, const char* mode)` may be helpful converting between file pointers and file descriptors.

```
void run_part(char** argv, FILE* a, FILE* b) {
```

C) [8 marks]
[Assignment 3 cover: 8/19 marks]

Write a **function** called `run_cmd`, which takes an array of strings representing a series of commands and executes each of them. Arguments are separated by arbitrary numbers of spaces in each string. In addition to the command strings to run, the function should take two filenames `a` and `b`.

The `stdin` for the first command should be file `a`, and the `stdout` for the last command should be file `b`. For all other commands, the `stdout` from one command should be piped to the `stdin` for the next command.

You are not permitted to call `system()` or `popen()`. You may make use of functions described in earlier parts (and you may assume that they are implemented correctly). You may assume that system calls succeed. The C library functions `int fileno(FILE* f)` and `FILE* fdopen(int fd, const char* mode)` may be helpful converting between file pointers and file descriptors.

```
void run_cmd(const char** cmds, const char* a, const char* b) {
```

D) [5 marks]
[Assignment 1 cover: 5/16 marks]

Write a **program** which takes one command-line argument which is the filename of a file that contains (in order):

- A source file name on the first line.
- A destination file name on the second line.
- A list of commands, one per line, with arguments separated by spaces.

The program should execute the commands, reading data from the source file to `stdin` of the first command, then passing `stdout` from each command to `stdin` of the next command. The `stdout` of the last command should be written to the destination file.

You are not permitted to call `system()` or `popen()`. You may make use of functions described in earlier parts (and you may assume that they are implemented correctly). You may assume that system calls succeed. The C library functions `int fileno(FILE* f)` and `FILE* fdopen(int fd, const char* mode)` may be helpful converting between file pointers and file descriptors.

If an incorrect number of arguments is passed, print nothing and exit with status 1.

This page intentionally left blank for extra working space.

Question 10) [**5 marks**]
[Assignment 4 cover: 5/19 marks]

Write a function `total` which takes a file descriptor for a connected socket. It should read one line of text, terminated with a newline character, from the socket. Following this it should count the total number of characters in the line of text (excluding the newline), then print this as an integer to the socket and finally **disconnect**. The function should not return anything. You may omit `#includes` and you may assume all system calls succeed.

Question 11) **[10 marks]**
[Assignment 4 cover: 10/19 marks]

In all of the following you may omit `#includes`. You may assume that all system calls succeed. You do not need to consider integer overflow. You may write additional functions to call.

Write a function called `matrixsum` which uses threads to distribute the workload in adding all of the elements of a large 2D array of integers. Each thread should sum an entire row (1D array) of the 2D array.

The function will take the following parameters:

1. `int** matrix` — a 2D array of integers of dimension `[rows][cols]`.
2. `int rows` — the number of rows in the matrix (the size of the first dimension).
3. `int cols` — the number of columns in the matrix (the size of the second dimension).

and return an integer which is the sum of all elements of the matrix.

Note that your function must make non-trivial use of threads.

END OF EXAMINATION

This page left intentionally blank for extra working space.
This page will **not be marked** unless there is a note earlier on the paper pointing to this page.

This page left intentionally blank for extra working space.
This page will **not be marked** unless there is a note earlier on the paper pointing to this page.

This page left intentionally blank for extra working space.
This page will **not be marked** unless there is a note earlier on the paper pointing to this page.

The following content may be helpful in answering the exam questions.
You may detach this sheet.
Do not record answers on this sheet (either side).
This sheet will not be marked.

Example addresses:

broadcast:	130.102.17.255
netmask:	255.255.255.0
CIDR:	130.102.17.0/24
private addresses:	10.0.0.0/8
	172.16.0.0/12
	192.168.0.0/16

KiB=	2^{10} Bytes
MiB =	2^{20} Bytes
GiB =	2^{30} Bytes

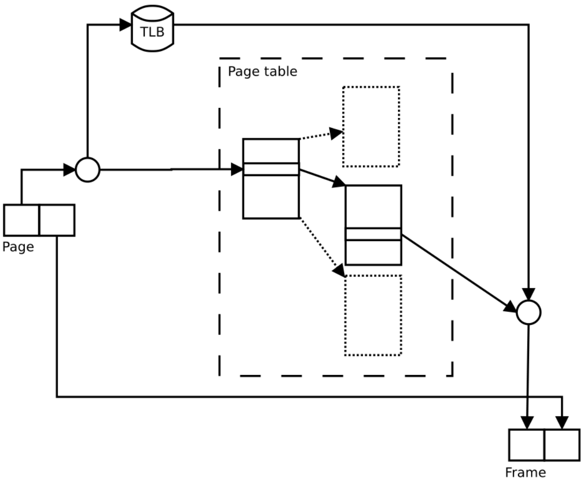
String format specifiers

`%c`	character
`%d`	integer
`%u`	unsigned integer
`%lf`	double (scanf)
`%f`	double (printf)
`%p`	void pointer
`%ld`	long integer

`[]`	array access
`.`	member selection
`->`	follow and select
`++ --`	
`sizeof`	
`~`	bitwise not
`!`	logical not
`+ -`	unary forms
`&`	address of
`*`	follow
`()`	cast
`* / %`	
`+ -`	binary forms
`> < <= >=`	
`== !=`	
`&`	bitwise AND
`^`	bitwise XOR
`|`	bitwise OR
`&&`	logical AND
`||`	logical OR
`? :`	ternary operator
`= += -= ...`	
`,`	comma operator

Commands

```
grep [-v] [ $ ^ . * ]
ls [-ladi]
ps [-ef]
kill [-s]
sort [-r -k]
uniq [-c]
cat
head [-n]
tail [-n]
cut [-f -d]
wc [-l]
diff
svn
chmod
ln [-s]
rm [-rf]
mkdir
rmdir
cp [-r]
mv
vim/nano
less
gcc [-o -c -std -L -l -I -g]
```



Network layers
link
application
onion
physical
network
gooey caramel
transport