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, 2019

CSSE2310/7231 Computer Systems Principles and 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 single 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 questions.
Safety net 1 = Questions 1A-E, 3, 9B-D (19 marks total)
Safety net 3 = Questions 2, 8, 10A-B (19 marks total)
Safety net 4 = Questions 6A-B, 9F-9G (19 marks total)

For Examiner Use Only

Page	Mark
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
22	

Question 1) Write shell commands to do the following: [10 marks (1 each)]

A) Make a new subversion working copy from url `https://files.uq.edu.au/svn/rep`

```
svn checkout https://files.uq.edu.au/svn/rep
```

B) Make an executable program from `start.c` which can be debugged with gdb.

```
gcc -g start.c
```

C) Compile a program from `a.c`, `b.c` and linking the maths (`libm`) library.

```
gcc -lm a.c b.c
```

D) Show the names of all the `.c` files in the current directory.

```
ls *.c
```

E) Copy all the lines which contain "rose" from the file `data` into a file called `matches`.

```
grep rose data > matches
```

F) $ ls -l instr

```
-rw------- 1 me me 1256 Sep 15 11:50 instr
```

Make shell script `instr` runnable by anyone.

```
chmod u+x g+x o+x instr
```

G) A file `x.cols` has columns separated by ':'. Output the second column of `x.cols`.

```
cut -d ':' -f 2 x.cols
```

H) Output the lines of `data` which contain both "chocolate" and "icecream".

```
grep chocolate data | grep icecream
```

I) Output the first three lines (in lexicographic order) of `data` which contain "muffin".

```
grep muffin data | head -n 3 | sort
```

J)

```
UID        PID  PPID  C STIME TTY          TIME CMD
me        2480  1513  0 11:15 pts/2    00:00:00 bash
me        2971  2480  0 12:19 pts/2    00:00:00 args
me        2972  2971  0 12:19 pts/2    00:00:00 vi
me        2973  2480  0 12:19 pts/2    00:00:00 process
me        2974  2480  0 12:19 pts/2    00:00:00 ps -f
```

Remove process id 2972 from the list of running processes.

```
kill -9 2971
```

Question 2) Write C to declare `foo` as . . . : [5 marks (1 each)]

A) A function which returns a floating point value and takes two integers as parameters.

```
float foo(int, int);
```

B) A `true` / `false` value.

```
bool foo;
```

C) An array of small integers.

```
short foo[];
```

D) A positive integer which could change in unexpected ways.

```
volatile unsigned int foo;
```

E) A pointer to a function which returns nothing and takes no parameters.

```
void (*foo)(void)
```

Question 3) Give the output for the following code fragments. [6 marks (1 each)]

A)

```
int x=12, y=9;
printf("%d", x&y);
```

B)

```
int x=13, y=9;
printf("%d", x|y);
```

C)

```
printf("%f\n", 1.5+5/2);
```

D)

```
int x=0, y=13;
for (int i=1;i<4;++i) {
    x+= (y%i);
}
printf("%d", x);
```

E)

```
int x=5, y=12;
if ((y--) || (x=0)) {
    x++;
} else {
    y=5;
}
if (x>y) {
    x++;
}
printf("%d %d", x, y);
```

F)

```
int x=7, y=4;
int z=1;
switch (x%y) {
    case 0: z++; break;
    case 1: z+=4; break;
    case 2: z--; break;
    default:
        z=3;
}
printf"%d", z);
```

Question 4) [6 marks (2 each)]

Suppose a system uses 34bit virtual addresses, 40bit physical addresses and a two level page table. Pages are 8KiB (8192 Bytes) each. Page table entries are 8 Bytes each.
(All addresses are expressed in base 10).

A) A process uses virtual address ranges:

- 4MiB starting at 81920
- 2MiB starting at 20455424

How much memory would be required to store the page table.

page : 81920/8192 = 10
4MiB = 2^22Bytes
How many pages is that? 2^22/2^13 = 2^9 pages
So, we need 2^9-page table entries.
How many entries fit per page of the table?
2^13/2^3 = 2^10 = 1024 entries per page of table.
How many pages of table do we need?
2^9/2^10 = 0.5
We need an extra page for the top level of the table.

page : 20455424/8192 = 2497
2MiB = 2^21Bytes
How many pages is that? 2^21/2^13 = 2^8 pages
So, we need 2^8-page table entries.
How many entries fit per page of the table?
2^13/2^3 = 2^10 = 1024 entries per page of table. How many pages of table do we need?
2^8/2^10 = 0.25
So, to store the table we need 2 +1 pages = (2+1) * 8KiB.
= 3*8 KiB = 24 KiB

B) Part of a page table for a process is given below:

Page	Frame
29	-
30	14
31	16
32	19
33	-

Which physical address do the following virtual addresses map to?
(If they would SEGFAULT say so.)

253951	122879
262242	155746

page = 253951/8192= 30.999 = 30
offset = 253951%8192 = 8191
phy = 14*8192 + 8191 = 122879

page = 262242/8192 = 32.0119= 32
offset = 262242%8192 = 98
phy = 19*8192 + 98 = 155746

C) A single threaded process accesses the following virtual addresses in order. It segfaults on the last one.

418831	page = 418831/8192= 51.1268 = 51
499722	page = 499722/8192= 61.0012 = 61
426981	page = 426981/8192= 52.1217 = 52
434275	page = 434275/8192= 53.0120 = 53

Assuming a process uses consistent virtual pages each time it runs, which of the following addresses would cause a segfault if accessed? (Circle your answer).

	Segfaults?	
434175	Yes / No	page = 434175/8192= 52.9998 = 52
442367	Yes / No	page = 442367/8192= 53.9998 = 53
418581	Yes / No	page = 418581/8192= 51.0963 = 51
435210	Yes / No	page = 435210/8192= 53.1262 = 53

Question 5) [9 marks]

Consider the following directory listing:

```
$ ls -ali
total 784
7602220 drwxr-xr-x  3 usr   usr      4096 Sep 15 13:34 .
7602177 drwxrwxrwt 17 root  root   131072 Sep 15 13:32 ..
7602221 -rwxr-xr-x  1 usr   usr    138856 Sep 15 13:29 act1
7602224 -rwxr-xr-x  2 usr   usr     64288 Sep 15 13:31 act2
7602224 -rwxr-xr-x  2 noddy noddy   64288 Sep 15 13:31 act3
7602226 drwxr-xr-x  2 guest guest    4096 Sep 15 13:33 backup
7602222 -rw-r--r--  2 guest guest   98037 Sep 15 13:30 doc
7602222 lrwxrwxrwx  2 guest guest       3 Sep 15 13:31 draft -> doc
7602223 -rw-r--r--  1 guest guest  294111 Sep 15 13:30 text
```

A) There are some inconsistencies in the listing. What are they? [4 marks]

1. the .. directory have t on other execute permission
2. the hard link- act2 and act3 (760224) have different owner and associated group, which should be same.
3. the symbolic link draft -> doc shares the same inode with doc, which the link should store in different inode.

B) How many **other** subdirectories does our parent directory have? [2 marks]

17 - 2 - 1 = 14

C) A system has the following ordinary users and groups: [3 marks]

User	Groups
noddy	watch
lutze	hist watch
lobsang	hist guild
boggis	guild

Consider the following directory listing:

```
-rwx---r-- 1 boggis guild 138856 Sep 15 13:29 p1
----rwxr-x 1 lutze  watch  70286 Sep 15 13:00 p2
-r--r-x--x 1 noddy  watch  64288 Sep 15 13:04 p3
```

Which users are allowed to:

1. Read from `p1`?

   boggis, noddy, lutze

2. Write to `p2`?

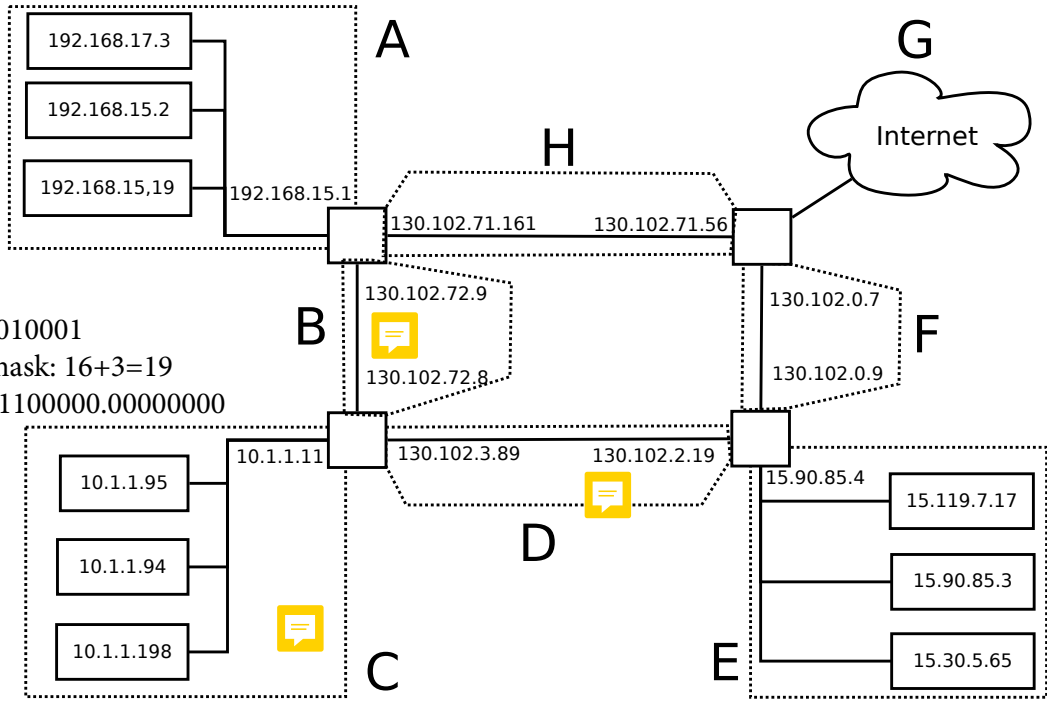
   noddy

3. Run `p3`?

   lutze lobsang, boggis

Question 6) [15 marks]

Consider the following network (assume that all networks are as small as possible):



min: 192.168.15.2
max: 192.168.17.3
15: 00001111 17: 00010001
length of the subnet mask: 16+3=19
11111111.11111111.11100000.00000000
Sm: 255.255.224.0

Na = AND(255.255.224.0, 192.168.15.2) = 192.168.0.0
Ba = OR(Na, Sm-1) = OR(192.168.0.0, 0.0.31.255) = 192.168.31.255
2^(32-19) - 2 - 3 = 8187

A) Fill in the details for each of the subnets in the diagram. [6 marks]
(Assume each network is as small as possible.)

Network	Netmask	Broadcast	CIDR
A	255.255.224.0	192.168.31.255	192.168.0.0/19
B	255.255.255.254	130.102.72.9	130.102.72.8/31
C	255.255.255.0	10.1.1.255	10.1.1.0/24
D	255.255.254.0	130.102.3.255	130.102.2.0/23

B) How many unused addresses are in each of the following networks? [2 marks]

A	8187
E	8388603
F	12

min: 15.30.5.65
max: 15.119.7.17
30: 00011110 119: 01110111
length of the subnet mask: 8+1=9
2^(32-9) - 2 - 3 = 8388603

min: 130.102.0.7
max: 130.102.0.9
7: 00000111 9: 00001001
length of the subnet mask: 24+4=28
2^(32-28) - 2 - 2 = 12

C) Which of the following are usable on the public internet? [2 marks]

192.168.15.19 Yes / No

10.1.1.198 Yes / No

130.102.0.8 Yes / No

15.30.5.65 Yes / No

D) An organisation owns the `8.19.29.0/19` block and they wish to break it into networks each containing 128 machines. What is the maximum number of such networks they could have? [2 marks]

Number of network addresses = 2^(32-19) = 8192
Each subnetwork should hold 128 machines (128 different IP address), including broadcast and network addresses this is 130. Requires 8 bits per subnetwork (does not fit 2^7).
Number of subnetworks =floor(8192 / (2^8)) = 32 networks

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

svn	application
wifi	link
IPv4	network
UDP	transport
HTTP	application
MAC	link

Question 7) Consider a "unix" filesystem where: [8 marks, 2 each]

- blocks are 4KiB
- block pointers are 8 Bytes.
- inodes have:
  - 6 direct pointers
  - 2 single indirect pointers
  - 2 double indirect pointers.

A) What is the smallest file size which requires use of the second single indirect pointer?

Pointers per block: 4 KiB /8 bytes = 512
4 KiB * [(6) + (1)*512 + (0)*512^2] = 2072 KiB
2072 KiB + 1B

B) If each inode is 150B, what else is likely stored in the inode?

- inodes may store metadata (times of last change, access, modification), as well as owner and permission data.
- directory entires map file names to inodes

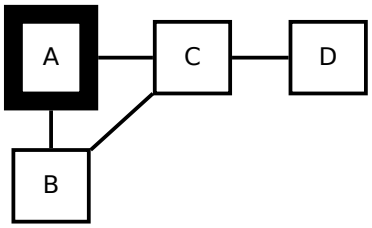
C) What is the maximum file size on this filesystem?

Pointers per block: 4 KiB /8 bytes = 512
4 KiB * [(6) + (2)*512 + (2)*512^2] = 2 101 272 KiB

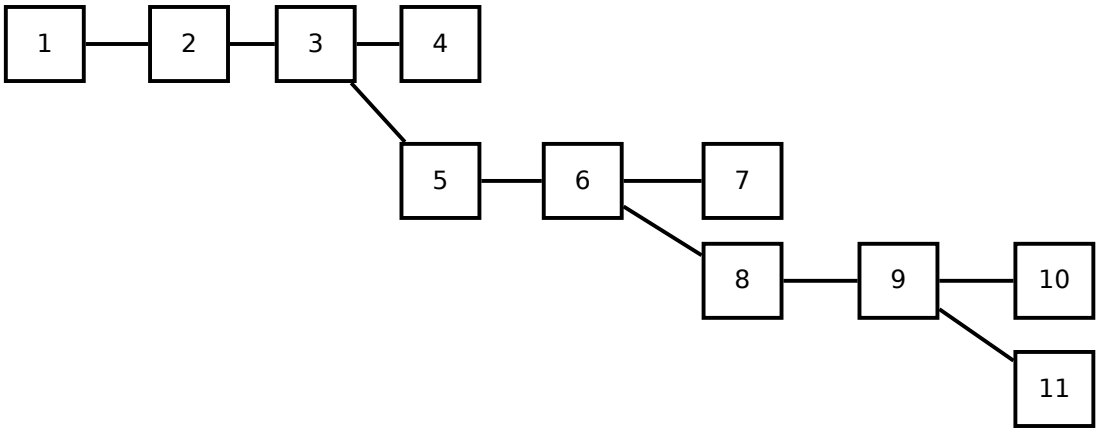
D) If this filesystem is stored on a spinning disk, how could fragmentation affect use of this filesystem?

File system fragmentation increases disk head movement or seek time (where it applies), which are known to hinder throughput. In addition, file systems cannot sustain unlimited fragmentation.

Question 8) [4 marks]



The diagram shows 4 processes (running programs `A, B, C, D`). Which calls, in which order will be required to reproduce this configuration (the first process is indicated with a thicker border).



Fill in the correct operations in the boxes below: `fork, execA, execB, execC, execD, fctl, pipeAB, pipeBC, pipeAC, pipeCD, pipeAC, sigaction`

1.	pipeAB	5.	pipeBC	9.	fork
2.	pipeAC	6.	fork	10.	exec(C)
3.	fork	7.	exec(B)	11.	exec(D)
4.	exec(A)	8.	pipeCD		

Question 9) [27 marks]

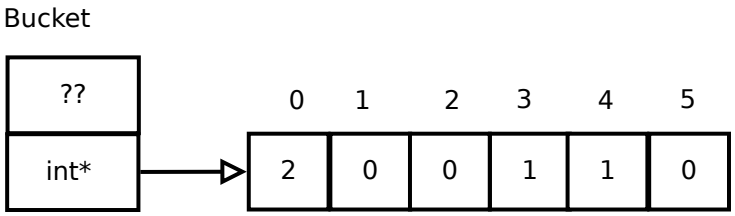
Where you are asked to write functions, you can also write helper functions or types if you need them. You may assume that system calls succeed.

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

We will be bucket sorting some integers in the range $[0, \ldots, N]$. Make an array of `int`s of size $N + 1$. Each time a number x is `seen()`, increment `array[x]`. To output the sorted sequence, loop through the array looking for indices which have a non-zero total.
Eg:

```
Bucket b;
init(&b, 5);
seen(&b, 0);
seen(&b, 4);
seen(&b, 0);
seen(&b, 3);
```

Would result in a data structure like this:



Printing would output:

```
0,0,3,4
```

A) Write a `typedef`ed `struct` type declaration for `Bucket`. You should look at parts **B**–**E** to see what you might need. [3 marks]

B) Implement `void init(Bucket* b, int upperLimit)` [3 marks]

C) Implement `void cleanup(Bucket* b)` [3 marks]

D) Implement `void print_sorted(Bucket* b, FILE* f)` : output comma separated values [2 marks]

E) Implement `void seen(Bucket* b, int value)` : silently reject any values which are out of range. Your implementation should be thread-safe. [5 marks]

F) Implement `void do_thing(int fd, Bucket* b)` where `fd` is a connected socket file descriptor. Read a line of text (no more than 200 chars) from that socket. The line will contain space separated integers. Add each integer to the bucket, output the number of values read to the socket and close it. [5 marks]

G) Implement `void startup(int limit, int conc, int* conv)` where [6 marks]

- `limit` is the largest number expected for a Bucket.
- `conv` is an array of connected socket file descriptors.
- `conc` is the number of desceriptors in the array.

The function should create a `Bucket`, run an instance of `do_thing()` for each descriptor in separate threads. Once all of the threads have finished, print out the values in the bucket to `stdout`.

Question 10) [10 marks]

A) [9 marks]

Write a C program to execute the following shell command without using a shell. You are not permitted to call `system()`. You may omit `#includes`. You may assume that all system calls succeed.

```
./translate mode 7 < source > output
```

B) Running `svn status` produces the following output: [1 mark]

```
? Z.h
M X.c
D P.c
A B.c
```

What affect would `svn commit` have on the repository?

the modified version o Z.h will be sent to the svn repository
P.c will be deleted from the svn repository
B.c will be added to version control

END OF EXAMINATION

Working space

This page left blank

Working space

This page left blank

Working space

This page left blank

You may detach this sheet.
Do not record answers on this sheet.

Example addresses:

broadcast	130.102.17.255
netmask	255.255.255.0
CIDR	130.102.17.0/24

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

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

Layers
link
application
onion
physical
network
gooey caramel
transport

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

[ ]	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