# CSSE2310: 2012 mid-sem exam answers

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

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

#### [**Other exam papers**](https://docs.google.com/folder/d/0B6_D4T6LJ-uwYzY1YWMzNjYtMzUyZC00OTEyLWJlMjktOGExYWUwOTc4NDE3/edit)

### 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. Which of the following programs can be used to look for the word “spider" in a given file?

(a) ls, chmod

(b) less, ls, chmod

(c) ls, less

(d) grep, less

(e) ls, grep, chmod

Discussion:

grep can search files, less can view a file (and you can search within it).

chmod will change file permissions

ls will list contents of the current directory, and not show file contents or search.

2. Assume the environment contains the following:

PATH=/bin:/usr/bin

USER=joel

What will be output by the following bash commands:

PATH=$path:/opt/bin

me=USER

echo "$me path is $PATH"

(a) joel path is /bin:/usr/bin:/opt/bin

(b) USER path is /bin:/usr/bin:/opt/bin

(c) USER path is :/opt/bin

(d) joel path is /bin:/usr/bin:/opt/bin

(e) $me path is $PATH

Discussion:

Remember that variables in Bash are case sensitive.

$me was set to “USER”, because no $ was put in front of USER in the line “me=USER”.

$path does, not exist so is empty when used for “PATH=$path:/opt/bin”

3. The output from ls -l for a file includes the following:

-rwxr--rw- 1 me folk

Who is allowed to modify the contents of the file?

(a) me

(b) me and users in the folk group

(c) me and users not in folk

(d) everybody but me

(e) users in folk but not me

Discussion:

The user ‘me’ and others have write permissions. Users in the folk group do not have write, and do not get it from others, so they are left out.

4. Which line in the following function has a compile error or warning under -Wall -pedantic?

int factorial(int n) { /\* 1 \*/

if (n>1) { /\* 2 \*/

return n\*factorial(n-1); /\* 3 \*/

} else { /\* 4 \*/

return; /\* 5 \*/

} /\* 6 \*/

} /\* 7 \*/

(a) line 1

(b) line 2

(c) line 3

(d) line 5

(e) They are all correct.

Discussion:

Line 5 has a return with no value. As the function specifies a return type of int, this is invalid.

5. Which of the following will produce a program called fred?

(a) gcc fred.c

(b) gcc -o fred.c fred

(c) gcc fred.c -c

(d) gcc -o fred fred.c

(e) none of the above

Discussion:

The -o option of gcc specifies the output filename.

6. What are the values of i and j after the following?

int i, j=0;

for (i=0; i<10; ++i) {

j+=3+i;

}

(a) i==9, j==75

(b) i==10, j==75

(c) i==10, j==85

(d) i==9, j==85

(e) none of the above

Discussion:

i will keep incrementing until it reaches the end condition for the loop. This means it will be incremented to 10 before the loop finishes.

j starts at 0, and has 3 + i added to it each time. so: j = 3, 3 + 4, 3 + 4 + 5, etc until it is 75

7. Consider the following function:

void f(void) {

int arr[7];

int\* p1=malloc(7\*sizeof(int));

int\*\* p2=malloc(2\*sizeof(int\*));

// code removed

}

No other allocations are performed. What code would need to be added to the bottom of the function to ensure that all memory allocated by f() is released?

(a) free(p1); free(p2);

(b) free(arr); free(p1); free(p2);

(c) free(p1); free(\*p2);

(d) free(p1); free(p2[0]); free(p2[1]);

(e) free(p1); free(\*p2); free(p2);

Discussion:

Only two variables were malloc’d. There is no need to dereference variables in this case when freeing them.

8. What is the value of a after the following has executed?

int a=1, b=2, c=3;

if ((a==2) || b) {

a=4;

} else {

a=0;

}

if (b-c) {

a=a+3;

} else {

a=a+10;

}

(a) 7

(b) 4

(c) 14

(d) 10

(e) None of the above

Discussion:

(a==2) is false, but b is non-zero, so makes the expression true. This means a=4.

b-c == -1, which is non-zero, so a = a + 3 = 7.

9. The following declares a function pointer, how many parameters does the function it would point at require?

int \* (\*fnptr)(void\*, void (\*)(int\*, int\*, int\*), void\* (\*)(char\*, char\*, char\*))

(a) 2

(b) 3

(c) 4

(d) 6

(e) 7

Discussion:

fnptr is a pointer to a function that takes 3 parameters: a void pointer, a function pointer that takes three int pointers and returns void, and a function pointer that takes three char pointers and returns a void pointer; and returns an int pointer.

Therefore the function that fnptr points at would take 3 parameters.

10. What is output by the following code?

int s(int a, int b) {

a=a & b;

return a;

}

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

int a=7, b=8, c=0;

c=s(a,b);

printf("%d %d %d\n", a, b, c);

return 0;

}

(a) 7 8 7

(b) 15 8 15

(c) 7 8 0

(d) 0 8 15

(e) 15 8 0

Discussion:

a and b are passed to the function s. Inside s, the arguments a and b are used to do:

a = a & b = 7 & 8 = 0 (recall that & is bitwise and)

This value is then returned, so c = 0.

Note that due to variable scope, a is not modified within in the main function, so it stays as 7.

11. What types are the variables declared in the following:

char\* c, d[][];

long e;

(a) c: a string, d: an array of strings, e: long string

(b) c: a string, d: a string, e: long integer

(c) c: a character, d: a string, e: long integer

(d) c: a string, an array of characters, e: long string

(e) the above is not legal C

Discussion:

It is not legal to declare a multidimensional array in this way in C.

12. Consider the following loop:

for (int i=0;(i==0) || (g(i)!=0);++i) {

h(i);

}

Which of these have exactly the same effect for all functions f() and g()?

int i=0; // first

while (g(i)!=0) {

h(i);

++i;

}

int i=0; // second

do {

h(i);

i++;

} while (g(i)!=0);

int i=0; // third

while ((i==0) || (g(i)!=0)) {

h(i);

i++;

}

a) First

b) Second

c) Third

d) First and third

e) Second and third

Discussion:

The third code fragment is equivalent to the for loop. The second code fragment would also have the same result.

The first code fragment does not include the i==0 condition, so may not be equivalent (we don’t know what g(0) would return).

13. Suppose you have edited a file (which is already part of the project) in a working directory. What sequence of commands is required to record those changes in your svn repository?

(a) svn update

(b) svn rm, svn add, svn commit

(c) svn checkout

(d) svn commit

(e) svn status

Discussion:

The file is already part of the project, so does not need to be added again. We do not want to remove the file either. We only need to commit our changes, so the only option is (d).

14. What is the purpose of the TLB?

(a) To speed up page table lookup.

(b) To allow lecturers to wear hats.

(c) To allow larger virtual address spaces.

(d) To allow larger physical address spaces.

(e) To provide memory protection.

Discussion:

The Translation Lookaside Buffer has a fixed number of slots to map virtual addresses to page table entries. This makes page table lookups much faster for pages that have recently been accessed.

15. Consider the following page table for a system with 4KB pages:

Virtual Physical

|  |  |
| --- | --- |
| 0 | Invalid |
| 1 | 4 |
| 2 | 1 |
| 3 | 7 |
| 4 | 2 |

Which physical address does virtual address 8191 map to?

(a) 8191

(b) 20479

(c) 28671

(d) 12287

(e) None, segfault

Discussion:

4KB = 4096 B

First, find which page the virtual address is in (remember we use integer division):

8191 / 4096 = 1

From the table, we can see that virtual page 1 maps to physical frame 4.

Next, find the offset within this page:

8191 % 4096 = 4095

Finally, calculate the physical address:

frame \* page size + offset = 4 \* 4096 + 4095 = 20479