# CSSE1001: Sem. 1 2008 exam answers

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

## Get more out of your study time. Join UQAttic's Semester 1, 2012 revision chat.

#### **[Other exam papers](https://docs.google.com/folder/d/0B_gnBD5zp3rqT2hrLWhvd2Rkc1U/edit)**

### Read this page first!

### 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.

### Style.

### 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.

### If you are confident that your answer is correct, then just type it in red below each question.

### 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.

### You can disable automatic email alerts of comment discussions via the "Discussions" button.

The following answers were initially made by a past tutor of this course, the original version is published at<http://codefisher.org/csse1001/> with the disclaimer: “I don't guarantee the answers are 100% correct, however as someone who achieved a 7 for the subject, used Python regularly ever since, and tutored it 3 times I would suggest the chance of an error is small.”

Feel free to modify, add to, or comment on these answers.

1. b)

2. a) integer division always rounds down toward negative infinity.

3. b) Concatenating strings is done with the + operator

4. d) You can't add (concatenate) int's with str's

5. c) Though maybe not perfectly intuitive, this can be done. It add the string to its self a given

number of times.

6. a) Counting from 0, 'a' is the 2nd character.

7. c) staring at the 1st character up to but not including the 3rd.

8. a) staring at the ­3rd character (the last is ­1) up to but not including the ­1th character.

9. c) going from the 2nd down to 0th character in steps of ­1.

10. c) splits on all cases of the space, a second argument is needed to limit this. The space is not included.

11. d) the pop removed the character in the 2nd position, which also happens to be 2, and then

inserts 9 at position 1.

12. e) can't use square brackets to access something that does not exist. It gives an error.

13. d) The get function takes two arguments. The first is the key to find, the second the value to

return if it is not found. It defaults to None.

14. b) The second argument was give, so it was given instead of None.

15. d)

16. b)

17. d)

18. d) This function sums all items in the list, between the two indexes.

19. e) Here we will run over the end of the list trying to get the 3rd element and this will give and

error

20. a) It will never go around the loop and just return the initial value of t.

21. b) This function join the two lists together into a list of tuples.

22. e) The second list is shorter, so when getting the 2nd item of ys it will result in an error.

23. b) The the last item on the second list will not be included.

24. a) we have seen 0 lines, and have a total of 0. The “close f” is a typo, should be “f.close()”

25. d) increment the count, and add the value to the sum.

26. c) if the statement is true we want to set a to half, so we have to check, according to the loop

invariant that half\*\*3 <= n, which is equivalent to n > half\*\*3

27. d) if no \_\_init\_\_ method is given a default one is supplied. As with all methods there must

be a reference to the object as the first argument.

28. d) it can do any, all, or none of the given.

29. c) there is a default \_\_str\_\_ method which returns the object type and memory address.

Which is not very useful.

30. b) self.x is set to 3 by \_\_init\_\_ and f() and g() are returning self.x and fg() is adding f() and

g() so that is 6

31. c) in B1 the f() is changed to returning self.x squared so that gives 9 + 3 = 12

32. a) B2 is setting self.x to 12 (don't miss the 4\*x) and then g() is now giving negative self.x so

12 – 12 = 0

33. d) fg() is now changed to subtracting f() from g() so 12 ­ (­12) = 24

34. c) which should be clear if you have done the second assignment. b) is also true, just c) is

more descriptive.

35. c) It works almost the same way as you may have done multiplication for the first time. It

adds a together b times.

36. b) we want all the items in the last except the last. pop() returns the item it removed and c

does not include the last element.

37. b) The sublists just found add to the sublists with the first element added to all of them.Ignore the second ) in both b and c.

38. c) the time taken will be linearly dependent on the length of the list since it runs over all of

them once.

39. b) if you doubled the length of the list, it only takes once extra loop up to find it, since it is

just continuously splitting the range the list it is looking over in half.

40. d) all the odd number less then 10 with 1 added to them.