Week 1

Quiz: chapter 6

1. What does the following Python Program print out?

```
str1 = "Hello"
str2 = 'there'
bob = str1 + str2
print bob
```

- Hello there
- Hello
- Hello there
- Hellothere

2. What does the following Python program print out?

```
x = '40'
y = int(x) + 2
print y
```

- x2
- 402
- 42
- int402

3. How would you use the index operator [] to print out the letter q from the following string?

```
x = 'From marquard@uct.ac.za'
```

- print x[q]
- print x[8]
- print x[9]
- print x[-1]
- print x[7]

4. How would you use string slicing [:] to print out 'uct' from the following string?

```
x = 'From marquard@uct.ac.za'
```

- print x[14+17]
- print x[15:3]
- print x[15:18]
- print x[14/17]
- print x[14:3]
- **print x[14:17]**

5. What is the iteration variable in the following Python code?

```
for letter in 'banana' :
    print letter
```

- for
- **letter**
- print
- 'banana'
- in

6. What does the following Python code print out?

```
print len('banana')*7
```

- 0
- **42**
- bananabananabananabananabananabananabananabanana
- banana7

7. How would you print out the following variable in all upper case in Python?

```
greet = 'Hello Bob'
```

-
```
print upper(greet)
```
-
```
int i=0;
char c;
while (greet[i]){ c=greet[i];
 putchar (toupper(c));
 i++;}
```
-
```
puts greet.ucase;
```
-
```
print greet.upper()
```

8. Which of the following is **not** a valid string method in Python?

- shout()
- split()
- join()
- startswith()

9. What will the following Python code print out?

```
data = 'From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008'
pos = data.find('.')
print data[pos:pos+3]
```

- mar
- 09:14
- uct
- .ma

10. Which of the following string methods removes whitespace from both the beginning and end of a string?

- ○ strtrunc()
- ◉ strip()
- ○ wsrem()
- ○ rltrim()

Week 3

Quiz: chapter 7

1. Given the architecture and terminology we introduced in Chapter 1, where are files stored?

- ○ Motherboard
- ○ Central Processor
- ○ Main Memory
- ◉ Secondary memory

2. What is stored in a "file handle" that is returned from a successful **open() call?**

- ○ The handle contains the first 10 lines of a file
- ○ All the data from the file is read into memory and stored in the handle
- ○ The handle has a list of all of the files in a particular folder on the hard drive
- ◉ The handle is a connection to the file's data

3. What do we use the second parameter of the **open() call to indicate?**

- ○ What disk drive the file is stored on
- ◉ Whether we want to read data from the file or write data to the file
- ○ How large we expect the file to be
- ○ The list of folders to be searched to find the file we want to open

4. What Python function would you use if you wanted to prompt the user for a file name to open?

- ○ file_input()
- ○ alert()
- ○ gets()
- ◉ raw_input()

5. What is the purpose of the newline character in text files?

- ◉ It indicates the end of one line of text and the beginning of another line of text
- ○ It enables random movement throughout the file
- ○ It allows us to open more than one files and read them in a synchronized manner
- ○ It adds a new network connection to retrieve files from the network

6. If we open a file as follows:

```
xfile = open('mbox.txt')
```

What statement would we use to read the file one line at a time?

- ○
```
READ xfile INTO LINE
```
- ◉
```
for line in xfile:
```
- ○
```
while ((line = xfile.readLine()) != null) {
```
- ○
```
while (<xfile>) {
```

7. What is the purpose of the following Python code?

```
fhand = open('mbox.txt')
x = 0
for line in fhand:
      x = x + 1
print x
```

- ○ Remove the leading and trailing spaces from each line in mbox.txt
- ○ Reverse the order of the lines in mbox.txt
- ○ Convert the lines in mbox.txt to upper case
- ◉ Count the lines in the file 'mbox.txt'

8. If you write a Python program to read a text file and you see extra blank lines in the output that are not present in the file input as shown below, what Python string function will likely solve the problem?

```
From: stephen.marquard@uct.ac.za

From: louis@media.berkeley.edu

From: zqian@umich.edu

From: rjlowe@iupui.edu

...
```

- ○ split()
- ○ trim()
- ◉ rstrip()
- ○ ljust()

9. The following code sequence fails with a traceback when the user enters a file that does not exist. How would you avoid the traceback and make it so you could print out your own error message when a bad file name was entered?

```
fname = raw_input('Enter the file name: ')
fhand = open(fname)
```

- ○ signal handlers
- ◉ try / except
- ○ try / catch / finally
- ○ setjmp / longjmp

10. What does the following Python code do?

```
fhand = open('mbox-short.txt')
inp = fhand.read()
```

- (●) Reads the entire file into the variable **inp as a string**
- ( ) Checks to see if the file exists and can be written
- ( ) Prompts the user for a file name
- ( ) Turns the text in the file into a graphic image like a PNG or JPG

Week 4

Quiz: chapter 8

1. How are "collection" variables different from normal variables?

- ( ) Collection variables merge streams of output into a single stream
- ( ) Collection variables pull multiple network documents together
- ( ) Collection variables can only store a single value
- (●) Collection variables can store multiple values in a single variable

2. What are the Python keywords used to construct a loop to iterate through a list?

- ( ) def / return
- ( ) try / except
- (●) for / in
- ( ) foreach / in

3. For the following list, how would you print out 'Sally'?

```
friends = [ 'Joseph', 'Glenn', 'Sally' ]
```

- ( ) print friends[2:1]
- (•) print friends[2]
- ( ) print friends['Sally']
- ( ) print friends[3]

4. What would the following Python code print out?

```
fruit = 'Banana'
fruit[0] = 'b'
print fruit
```

- ( ) banana
- ( ) b
- ( ) Banana
- ( ) B
- ( ) [0]
- (•) Nothing would print - the program fails with a traceback

5. Which of the following Python statements would print out the length of a list stored in the variable **data**?

- (•) print len(data)
- ( ) print strlen(data)
- ( ) print length(data)
- ( ) print data.Len
- ( ) print data.length
- ( ) print data.length()

6. What type of data is produced when you call the **range()** function?

```
x = range(5)
```

- ( ) A string
- (•) A list of integers
- ( ) A list of words
- ( ) A list of characters
- ( ) A boolean (true/false) value

7. What does the following Python code print out?

```
a = [1, 2, 3]
b = [4, 5, 6]
c = a + b
print len(c)
```

- ( ) [1, 2, 3]
- ( ) 21
- ( ) 15
- (•) 6
- ( ) [4, 5, 6]
- ( ) [1, 2, 3, 4, 5, 6]

8. Which of the following slicing operations will produce the list [12, 3]?

```
t = [9, 41, 12, 3, 74, 15]
```

- ( ) t[12:3]
- ( ) t[:]
- ( ) t[2:2]
- (•) t[2:4]
- ( ) t[1:3]

9. What list method adds a new item to the end of an existing list?

- ○ forward()
- ○ index()
- ◉ append()
- ○ add()
- ○ push()
- ○ pop()

10. What will the following Python code print out?

```
friends = [ 'Joseph', 'Glenn', 'Sally' ]
friends.sort()
print friends[0]
```

- ○ Sally
- ○ Joseph
- ○ friends
- ◉ Glenn

Assignment

8.4 Open the file romeo.txt and read it line by line. For each line, split the line into a list of words using the split() method. The program should build a list of words. For each word on each line check to see if the word is already in the list and if not append it to the list. When the program completes, sort and print the resulting words in alphabetical order.

```
fname = raw_input("Enter file name: ")
fh = open(fname)
lst = list()
words = list()
for line in fh:
    words = line.split()

    for i in range(len(words)):
        if words [i] not in lst:
            lst.append(words[i])
            lst.sort()

print lst
```

assignment

8.5 Open the file mbox-short.txt and read it line by line. When you find a line that starts with 'From ' like the following line:

From stephen.marquard@uct.ac.za Sat Jan 5 09:14:16 2008

You will parse the From line using split() and print out the second word in the line (i.e. the entire address of the person who sent the message). Then print out a count at the end.

Hint: make sure not to include the lines that start with 'From:'.

```
fname = raw_input("Enter file name: ")
if len(fname) < 1 : fname = "mbox-short.txt"

fh = open(fname)
count = 0

for line in fh:
    if not line.startswith('From'):
        continue
    if line.startswith('From') and not line.startswith('From:'):
        lst = line.split()
        count = count+1
        print lst[1]


print "There were", count, "lines in the file with From as the first word"
```

Week 5

Quiz: chapter 9

1. How are Python dictionaries different from Python lists?

   - (●) Python lists maintain order and dictionaries do not maintain order
   - ( ) Python lists store multiple values and dictionaries store a single value
   - ( ) Python lists can store strings and dictionaries can only store words
   - ( ) Python dictionaries are a collection and lists are not a collection

1. How are Python dictionaries different from Python lists?

- ( ) Python lists store multiple values and dictionaries store a single value
- (•) Python lists are indexed using integers and dictionaries can use strings as indexes
- ( ) Python dictionaries are a collection and lists are not a collection
- ( ) Python lists can store strings and dictionaries can only store words

2. What is a term commonly used to describe the Python dictionary feature in other programming languages?

- ( ) Lambdas
- ( ) Closures
- ( ) Sequences
- (•) Associative arrays

3. What would the following Python code print out?

```
stuff = dict()
print stuff['candy']
```

- ( ) -1
- ( ) 0
- (•) The program would fail with a traceback
- ( ) candy

4. What would the following Python code print out?

```
stuff = dict()
print stuff.get('candy',-1)
```

- ( ) 0
- ( ) 'candy'
- (•) -1
- ( ) The program would fail with a traceback

5. (T/F) When you add items to a dictionary they remain in the order in which you added them.

- (●) False
- ( ) True

6. What is a common use of Python dictionaries in a program?

- ( ) Sorting a list of names into alphabetical order
- ( ) Splitting a line of input into words using a space as a delimiter
- (●) Building a histogram counting the occurrences of various strings in a file
- ( ) Computing an average of a set of numbers

7. Which of the following lines of Python is equivalent to the following sequence of statements assuming that **counts** is a dictionary?

```
if key in counts:
    counts[key] = counts[key] + 1
else:
    counts[key] = 1
```

- ( ) counts[key] = key + 1
- ( ) counts[key] = (key in counts) + 1
- (●) counts[key] = counts.get(key,0) + 1
- ( ) counts[key] = (counts[key] * 1) + 1
- ( ) counts[key] = counts.get(key,-1) + 1

8. In the following Python, what does the **for** loop iterate through?

```
x = dict()
...
for y in x :
    ...
```

- (●) It loops through the keys in the dictionary
- ( ) It loops through the values in the dictionary
- ( ) It loops through all of the dictionaries in the program
- ( ) It loops through the integers in the range from zero through the length of the dictionary

9. Which method in a dictionary object gives you a list of the values in the dictionary?

- (●) values()
- ( ) items()
- ( ) keys()
- ( ) all()
- ( ) each()

10. What is the purpose of the second parameter of the **get()** method for Python dictionaries?

- ( ) An alternate key to use if the first key cannot be found
- ( ) The value to retrieve
- (●) To provide a default value if the key is not found
- ( ) The key to retrieve

9.4 Write a program to read through the mbox-short.txt and figure out who has the sent the greatest number of mail messages. The program looks for 'From ' lines and takes the second word of those lines as the person who sent the mail. The program creates a Python dictionary that maps the sender's mail address to a count of the number of times they appear in the file. After the dictionary is produced, the program reads through the dictionary using a maximum loop to find the most prolific committer.

```
name = raw_input("Enter file:")
if len(name) < 1 : name = "mbox-short.txt"
handle = open(name)

counts = dict()
for line in handle:
    words = line.split()
    if line.startswith('From') and not line.startswith('From:'):
        counts[words[1]] = counts.get(words[1], 0)+1
        # save the emails and counts into the dict counts

maxcount = 0
maxsender = ''
for name, count in counts.items():
    if count > maxcount:
        maxcount = count
        maxsender = name

print maxsender, maxcount
```

Week 6

Quiz: chapter 10

1. What is the difference between a Python tuple and Python list?

- ○ Lists are indexed by integers and tuples are indexed by strings
- ○ Lists maintain the order of the items and tuples do not maintain order
- ○ Tuples can be expanded after they are created and lists cannot
- ● Lists are mutable and tuples are not mutable

2. Which of the following methods work both in Python lists and Python tuples?

- ○ pop()
- ○ reverse()
- ○ append()
- ○ sort()
- ● index()

3. What will end up in the variable **y** after this code is executed?

```
x , y = 3, 4
```

- ● 4
- ○ A two item tuple
- ○ 3
- ○ A dictionary with the key 3 mapped to the value 4
- ○ A two item list

4. In the following Python code, what will end up in the variable **y**?

```
x = { 'chuck' : 1 , 'fred' : 42, 'jan': 100}
y = x.items()
```

- ( ) A list of strings
- ( ) A tuple with three integers
- (•) A list of tuples
- ( ) A list of integers

5. Which of the following tuples is greater than **x** in the following Python sequence?

```
x = (5, 1, 3)
if ??? > x :
   ...
```

- ( ) (0, 1000, 2000)
- ( ) (5, 0, 300)
- (•) (6, 0, 0)
- ( ) (4, 100, 200)

6. What does the following Python code accomplish, assuming the **c** is a non-empty dictionary?

```
tmp = list()
for k, v in c.items() :
    tmp.append( (v, k) )
```

- ( ) It computes the largest of all of the values in the dictionary
- ( ) It sorts the dictionary based on its key values
- (•) It creates a list of tuples where each tuple is a value, key pair
- ( ) It computes the average of all of the values in the dictionary

7. If the variable **data** is a Python list, how do we sort it in reverse order?

- ( ) data = sortrev(data)
- (•) data.sort(reverse=True)
- ( ) data.sort.reverse()
- ( ) data = data.sort(-1)

8. Using the following tuple, how would you print 'Wed'?

```
days = ('Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun')
```

- print days[1]
- (●) print days[2]
- print days{2}
- print days(2)
- print days.get(1,-1)

9. In the following Python loop, why are there two iteration variables (k and v)?

```
c = {'a':10, 'b':1, 'c':22}
for k, v in c.items() :
    ...
```

- (●) Because the items() method in dictionaries returns a list of tuples
- Because for each item we want the previous and current key
- Because there are two items in the dictionary
- Because the keys for the dictionary are strings

10. Given that Python lists and Python tuples are quite similar - when might you prefer to use a tuple over a list?

- For a list of items you intend to sort in place
- (●) For a temporary variable that you will use and discard without modifying
- For a list of items that will be extended as new items are found
- For a list of items that want to use strings as key values instead of integers

10.2 Write a program to read through the mbox-short.txt and figure out the distribution by hour of the day for each of the messages. You can pull the hour out from the 'From ' line by finding the time and then splitting the string a second time using a colon.

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
```

Once you have accumulated the counts for each hour, print out the counts, sorted by hour as shown below.

```
name = raw_input("Enter file:")
if len(name) < 1 : name = "mbox-short.txt"
handle = open(name)

counts = dict()

for line in handle:
    if line.startswith('From') and not line.startswith('From:'):
        words = line.split()
        time = words[5].split(':')
        counts[time[0]] = counts.get(time[0], 0)+1

# dict cannot be sorted
# use list to sort instead
sortedtime = list()
for key, value in counts.items():
    sortedtime.append((key, value))
sortedtime.sort()

# print
for value, key in sortedtime:
    print value, key
```