Final Exam Practice Questions

Note that while these questions are great practice for the final exam, they do **not** necessarily cover ever topic or question type that could appear on the exam. **You are strongly encourage to do additional studying beyond completing these questions.**

1. Consider the following statement:

"The complexity for minimum component cost has increased at a rate of roughly a factor of two per year … Certainly over the short term this rate can be expected to continue, if not increase. Over the longer term, the rate of increase is a bit more uncertain, although there is no reason to believe it will not remain nearly constant for at least 10 years."

This statement is known as: _____

2. Convert 45 base 8 to binary.

Answer: _____

3. Convert 77 base 10 to base 16.

Answer: _____

4. What is a computer?

5. How does data differ from information?

Consider the following code segment:

```
a = "12"
b = 8
print(a + b)
```

6. When this program runs:
    A. It will display 20
    B. It will display 128
    C. It will display 12, 8
    D. It will display 812
    E. None of the above answers are correct

Answer: _____

Consider the following code segment:

```
m = int(input())
n = int(input())

if (m < n):
  print("A")
elif (m > n) and (m == 0):
  print("B")

if (n == 0):
  print("C")
else:
  print("D")
```

7. If the user enters 0 for m and -4 for n then the output will be: _____

8. If the user enters 0 for m and 0 for n then the output will be: _____

9. If the user enters 0 for m and 1 for n then the output will be: _____

10. If the user enters 1 for m and 0 for n then the output will be: _____

Consider the following code segment:

```
i = 1
value = 0

while i < 32:
  for j in range(1,i):
    value = value + 1
  i = i * 2
  print(value)
```

11. The first value printed out by this code segment will be: _____

12. The second value printed out by this code segment will be: _____

13. The third value printed out by this code segment will be: _____

14. The total number of values that will be displayed by this code segment will be: _____

Consider the following program. It totals 10 numbers entered by the user and reports if the total is less than, equal to, or greater than 1.0.

```
total = 0.0

for i in range(0,10):
  v = float(input("Enter a value: "))
  total = total + v

if total < 1.0:
  print("Your total is less than 1.0")
elif total == 1.0:
  print("Your total is exactly 1.0")
else:
  print("Your total is greater than 1.0")
```

15. When I run the program and enter the numbers 0.2, 0.2, 0.2, 0.2, 0.0, 0.0, 0.0, 0.0, 0.1 and 0.1, the program correctly reports that these numbers total exactly 1.0. However, if I enter the numbers 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1 and 0.1, the program incorrectly reports that the total is less than 1.0, even though the sum of the numbers is exactly 1.0. What is causing the program to behave incorrectly in this case?

16. Briefly describe how you could prevent the program from incorrectly reporting that the second sum is less than 1.0. What problem could your change introduce?

17. A while loop is an example of a this type of loop:
    A. if-comparison
    B. numeric-exclusion
    C. numeric-comparison
    D. post-tested
    E. pre-tested

Answer: _____

18. What is the defining difference between a pre-tested loop and a post-tested loop?

Consider the following function definition and function call:

```
def some_function(x,y,z):
  print(x)
  print(y)
  print(z)

# Create variables and call the function
a = 5
b = [1,2,3]
c = "Hello World!"
some_function(a,b,c)
```

19. The name of this function is: _____

20. The **name(s)** of the formal parameter(s) are: _____

21. The **names(s)** of the actual parameter(s) are: _____

22. The **value(s)** of the parameter(s) passed to the function which are immutable are:

Consider the following code segment:

```
def f2(c,d):
  print(c) # Statement 1
  c.append(0)
  print(d) # Statement 2

def f1(a,b):
  b = b + 1
  f2(a,b)
  b = b + 1

i = [10,5]
j = 0
f1(i, j)

print(i) # Statement 3
print(j) # Statement 4
```

23. The value of c printed by statement 1 will be: _____

24. The value of d printed by statement 2 will be: _____

25. The value of i printed by statement 3 will be: _____

26. The value of j printed by statement 4 will be: _____

Consider the following code segment. Recall that % is the remainder operator when it is applied to two integers.

```
values = [2,4,8,16,32]

for i in range(0,3):
  values[i] = values[i+2]

for i in range(0,len(values)):
  if i % 2 == 0:
    values[i] = values[i] + 1

print(values)
```

27. At the end of this code segment, values[0] has the value: _____

28. At the end of this code segment, values[1] has the value: _____

29. At the end of this code segment, values[3] has the value: _____

30. At the end of this code segment, values[4] has the value: _____

31. What is a data structure?

32. What are two examples of data structures that we worked with during this course?

Consider the following code segment:

```
data = [5, 4, 3, 1, 2, 0, 1]

p = 0
for i in range(0, len(data)):
  if data[i] % 2 == 0:
    p = data[i]
  else:
    data[i] = p

print(data)
```

33. What is displayed by this code segment? _____

34. Which of the following statements about Python dictionaries is most correct?
    A. Each element in a dictionary has a unique integer index that starts with 0
    B. Dictionaries are an example of an ordered data structure
    C. Each value stored in a dictionary must be unique
    D. More than one of the above statements are correct
    E. None of the above statements are correct

Answer: _____

35. Describe two differences between text files and binary files:

36. Consider the following command used to start a python program:

```
python  myProgram.py  Bob  John  42
```

Inside the program there is a statement:

```
print(sys.argv[0])
```

When that statement executes it will display:

    A. python
    B. myProgram.py
    C. Bob

D. John
E. 42

Answer: _____

37. A data structure is:
    A. A building that holds data
    B. A dictionary
    C. A mechanism for organizing related pieces of data
    D. A structure for searching quickly
    E. A technique for modifying data

    Answer: _____

38. A file that can be viewed with an editor such as Notepad, Emacs, Kate, GEdit or Vi is referred to as a(n) _____ file:
    A. Binary File
    B. Buffer File
    C. Source file
    D. Symbolic File
    E. Text file

    Answer: _____

39. Assume that we are opening a file that already exists. What will happen if you open this file in write mode ("w")? How is that different from opening the file in append mode ("a")?

40. List 3 common sorting algorithms:

41. In a relational database, data is organized into:
    A. Families
    B. Folders
    C. Groups
    D. Objects
    E. Tables

    Answer: _____

42. Within a database, a column which contains a unique value associated with each row is known as a:
    A. File Key
    B. Foreign Key

C. Principle Key
D. Primary Key
E. Schema Key

Answer: _____

43. What is the defining difference between a recursive function and a non-recursive function?

44. **[3 marks]** The following function is supposed to sort the elements of a list into increasing (ascending) order. All of the comments correctly state what the lines of code are supposed to do. However, the program contains 3 bugs. Locate each of the bugs and circle it. Write one sentence or less next to each bug describing what you would need to change to correct the bug.

```
# Sort some data and display it on the screen
# Parameters:
#   data: a list to sort into ascending order
# Returns: (None)
def sortAndDisplay(data):
  # Loop over the valid indices of the list in increasing order
  for i in range(len(data)):
    # Loop over the valid indices of the list in decreasing order
    for j in range(len(data), 0, -1):
      # If two adjacent elements are out of order
      if data[j] < data[j-1]:
        # Swap the element at pos. j with the element at pos. j-1
        t = data[j]
        data[j] = data[j-1]
        data[j-1] = data[j]

  # Display the result on the screen
  return(data)
```

45. **[9 marks]** In this question you will write a program that partitions integers as being either less than or greater than a cutoff value. Your program will begin by reading the cutoff value from the user. Then it will continue reading integers from the user until the user enters a blank line. Your program must include appropriate prompts and an appropriate output message before the result.

    The output of your program will be all numbers entered by the user that are less than the cutoff value in the same order they were entered, followed by the cutoff value, followed by all of the integers greater than the cutoff value in the same order they were entered.

    Example: If the user started by entering a cutoff value of 5, and then entered the following integers: 1, 9, 8, 2, 4, 0, 6 (one on each line), then your program should display the values 1, 2, 4 and 0 (the numbers less than the cutoff value), followed by 5 (the cutoff value), followed by the values 9, 8 and 6 (the numbers larger than the cutoff value). The output should be formatted so that one value appears on each line.

46. **[10 marks]** Write a Python program which reads lines from a file and counts the number of times that each unique line occurs. Once all of the lines have been read, the program should display each unique line and the total number of times that it occurred in the file. Prompt the user to enter the name of the file. Ensure that your program runs correctly when the file is empty. The order in which the items are displayed is not important. You do not need to catch errors such as trying to open a file that doesn't exist or blank lines in the input file.

Sample Input File:

```
Carrot
Radish
Carrot
Carrot
Apple
Radish
Carrot
```

Expected Output:

```
2 Radish
4 Carrot
1 Apple
```

47. **[10 marks]** Create a Python function that computes and displays the average water depth in a lake. The elevation / water depth data will be stored in a 2-dimensional list which will be passed as the only parameter to the function. Each element in the list represents a 1 square metre area, with a value less than 0 indicating the square is underwater (the magnitude of the number indicates how far under water). A zero or positive number indicates the elevation of an island or land area. Note that when you compute the average water depth, you should only include values that are underwater. Your function should print the depth of the water as a positive number formatted to two decimal places.

    Example: A lake containing a small island

-2.0	-1.0	-0.5	-0.5
-1.0	-0.5	0.5	-0.5
-1.0	0.5	1.0	-0.5
-1.0	-0.5	-0.5	-1.0

    The 2-dimensional list shown above contains 13 water squares and 3 land squares. The average water depth is (2.0 + 1.0 + 0.5 + 0.5 + 1.0 + 0.5 + 0.5 + 1.0 + 0.5 + 1.0 + 0.5 + 0.5 + 1.0) / 13 or approximately 0.81 metres. Your program should display an appropriate message if the list that is provided does not contain any underwater squares.

48. **[10 marks]** A palindrome is a sequence of characters that is identical forwards and backwards. For example, the words 'level', 'radar' and 'racecar' are all palindromes. Any string with only one character is also a palindrome. In general, palindromes can be identified by checking to see if the first and last characters are identical, and then checking to see if the remaining characters in the string also represent a palindrome. Write a python program that reads a string from the user and tells the user whether or not the string that was entered is a palindrome.

    Recall that the built-in Python function len(s) returns the integer length of the string, s, and that you can extract characters from the middle of a string by enclosing a range in square brackets.

49. **[5 marks]** Write a function, count_occurrences, which determines how many times a specific character appears within a string. Your function will take two parameters which are the character to search for, and the string that will be searched. It will return an integer result, which is the count of the number of times the character was present.

    Examples: count_occurrences("o", "Hello World!") should return 2
    count_occurrences("z", "Hello World!") should return 0.

50. **[9 marks]** Using the function you wrote in the previous question, create a program that counts the number of vowels in a file (recall that the letters a, e, i, o and u are vowels). Your program should read the name of the file from the user (with an appropriate prompt), determine the number of vowels, and then display the total number of vowels in the file with an appropriate message.

    Ensure that the count that is reported considers both upper and lower case letters. You may find it useful to use the upper() method for strings. When applied to a string, it returns a new copy of the string with all letters converted to upper case.