Practice Questions

Note: These questions are for you to practice for the final exam. The format and coverage of the final questions may not be the same.

Question 1

a. Give one example of a process or task which would be easier to implement in Scratch than Python, and briefly explain why it is easier in Scratch. Give one example of a process or task which would be easier to implement in Python than Scratch, and briefly explain why it is easier in Python.

b. Suppose a byte in the computer memory contains the following value 01110101. What kind of data can this value represent? Who decides what this value represents?

c. What is the binary representation of the integer value 88 in a byte?

Question 2

Consider the following Python program

```
def bar1(y):
    x = len(y) - 1
    while(x >= 0):
        if(x % 2 == 0):
            y[x] = ''
        x = x - 1
    return

# Main
x = 5
names =[ 'Jones', 'Tutu', 'Aziz', 'Butcher', 'Malkin', 'Chan' ]
result = bar1(names)
```

What are the values of `x`, `names` and `result` after this program is executed?

Question 3

Consider the following Python program

```
def bar2(y):
    x = len(y) - 1
    z = []
    while(x >= 0):
        if(x % 2 == 0):
            z.append(y[x])
        x -= 1
    return z
# Main
x  =  5
names = [ 'Jones', 'Tutu', 'Aziz', 'Butcher', 'Malkin', 'Chan' ]
result = bar2(names)
```

What are the values of `x`, `names` and `result` after this program is executed?

Question 4

Consider the following Python program

```
extensions = { 'Jones' : 547,  'Aziz' : 352, 'Butcher' : 978,
               'Malkin' : 177, 'Chan' : 444, 'Tutu' : 277 }
calls = [ 'Aaronson', 'Freeman', 'Aziz', 'Sastry', 'Chen',
          'Jones', 'Tutu', 'Luongo' ]
s = []
n = 0
for p in calls:
    if p in extensions:
        s.append(extensions[p])
    else:
        s.append(None)
        n += 1
```

What are the values of `extensions`, `calls`, `s` and `n` after this program is executed?

Question 5

Briefly **summarize** what the following Python function does when called with a list of strings as its first argument and a small integer as its second argument.

```
def foo(x, y):
    z = []
    for w in x:
        if len(w) >= y :
            z.append(w)
        else:
            z.append(w + ' ' * (y - len(w)))
    return z
```

Question 6

Briefly **summarize** what the following Python function does when called with a dictionary as its argument.

```
def grap(x):
    z = {}
    for y in x:
        z[x[y]] = y
    return z
```

Question 7

Define a function `get_value` which implements the following task:

- Inputs: A dictionary `old_values` and a variable `arg` (of unknown but immutable type).
- Outputs: If `arg` is a key in the dictionary, the function returns the value associated with it. Otherwise the function calls `compute(arg)`, associates this value with `arg` in the dictionary and returns this value.
- You should assume that the function `compute(arg)` is defined and returns some value. Also note that the dictionary `old_values` may also be modified as a side-effect.

Question 8

To answer the next two questions, it may be useful to know that method get() for dictionaries has the following description in the Python documentation:

> `get(key[, default])`: Returns the value for key if key is in the dictionary, else default. If default is not given, it defaults to None, so this method never raises a KeyError.

Consider the following Python code. :

```
def bar(string1, string2):
    result = []
    for ch in string1:
        if ch in string2:
            result.append(ch)
    return result

def foo(text, alpha):
    result = {}
    text = bar(text, alpha)
    for ch in text:
        result[ch] = result.get(ch, 0) + 1
    return result

text = "WELL, a += 20 is the same as a = a + 20 !"
r1 = foo(text, "aeiouy")
text = text.lower()
r2 = foo(text, "aeiouy")
```

a) Briefly explain (summarize) what the functions `bar` and `foo` do.

b) What are the values of `r1` and `r2` after this code is executed?

Question 9

a) Define a Python function that accepts as arguments two strings and returns a list that contains all the characters in its first argument that are not in the second argument. The characters in the list will appear in the same order and multiplicity as they appear in the first argument.

b) Define a Python function that accepts as arguments two strings and returns a dictionary whose keys are the characters that appear in the function's first argument but not in its second argument. Each character in the dictionary is associated with the percentage of times this character appears (as a fraction of the times this character appears over the total number of characters) in the first argument.

Question 10

a) Summarize what the following Python function does when it is called with a list of numeric values:

```
def fun(vals):
    size = len(vals)
    if size == 0 :
        return None

    newvals= []
    for val in vals:
        newvals.append(val*val)

    newvals.sort()

    i = size // 2
    if size % 2 ==0:
        return (newvals[i-1] + newvals[i]) / 2.0
    else:
        return newvals[i]
```

b) What would the call **`fun([ 5, -5, 2, -3, -10 ])`** return?

Question 11

a) Write the code for a function **median** that sorts its input list and calculates and returns the median (middle value) of the values in the list. if the list is empty, **median** returns **None**. If the list has an even number of values **median** also replaces the two middle values of the list with the median value.

b) Suppose we execute the following code :

```
list = [ 5, -5, 2, -2, -10, 8 ]
result = median(list)
```

What would the values of result and list be?

Question 12

Recall that in Python RGB images are represented by two dimensional array of pixels. Each pixel is a tuple of three numbers representing the amount for red, green and blue color is in the pixel. The position of each pixel is identified by the pair (column, row) which contains the column and row number for the pixel. The position of the pixel at the top-left corner is (0,0). The size of an image is also represented by the pair (width, height) containing the image's width and length.

Assume that the following commands are executed:

```
import Image

def vf(image):
    newimage = Image.new("RGB", image.size)

    (w, h) = image.size
    maxc = w-1

    for r in range(h) :
        for c in range(w):
            pix = image.getpixel((maxc-c, r))
            newimage.putpixel((c, r), pix)
    return newimage

# Main
image = Image.open("dog.jpg")
image = vf(image)
image.save("newdog.jpg")
```

Are the images in **newdog.jpg** and **dog.jpg** different after the program completes? If you believe that the two images are different, describe their differences.

Question 13

Write a function hf that accepts an image as its input and returns a new image which is the horizontal flip of the original image. That is, the top of the original image is the bottom of the new image and the bottom of the original image is the top of the new image.

Question 14

Summarize what the following Python function does when called with an image object:

```
def hm(image):
    (w, h) = image.size

    newsize = tuple([w, 2*h])
    newimage = Image.new("RGB", newsize)

    for c in range(w) :
        for r in range(h):
            pix = image.getpixel((c,r))
            newimage.putpixel((c, r), pix)
    maxh = h-1
    for c in range(w) :
        for r in range(h):
            pix = image.getpixel((c, maxh-r))
            newimage.putpixel((c, h+r), pix)

    return newimage
```

Question 15

Define a function **vm** which accepts an image as its input and returns a new image which has the same height and two times the width of the original image. The left half of the new image contains the old image and the right half has the mirror image of the left half. The original image is not changed.

Question 16

To answer the following two questions, it may be useful to know the following:

- If d is a dictionary, d.items() returns a copy of d's list of (key, value) pairs
- When the sort() function is applied to a group of tuples, the tuples in the group are ordered according to their first component. The tuples with the same first component are ordered according to their second component, and so on.

Assume that the following commands are executed:

```
def foo(dict):
    result = {}
    for it in dict:
        val = dict[it]
        if val not in result:
            result[val] = []
        result[val].append(it)
    return result

# Main
students = { 'Susan': 90, 'John': 65, 'Mary': 90, 'Jeff': 50,
'Jane': 65, 'Lora': 95 }

students1 = foo(students)
students2 = sorted(students1.items())
```

What are the values of students1 and students2 when these commands complete?

Question 17

Assume that the following commands are executed:

```
def foo(dict):
    result = {}
    for it in dict:
        val = dict[it]
        if val not in result:
            result[val] = []
        result[val].append(it)
    return result


def bar(dict):
    x = sorted(dict.items())
    result = []
    for i in range(len(x)):
        key, vals = x[i][0], x[i][1]
        vals.sort()
        for val in vals:
            result.append(tuple([key, val]))
    return result

# Main
```

```
students = { 'Susan': 90, 'John': 65, 'Mary': 90, 'Jeff': 50,
'Jane': 65, 'Lora': 95 }

students1 = foo(students)
students2 = bar(students1)
```

What are the values of students1 and students2 when these commands complete?

Question 18

Write a function `wordFreq` that accepts as its argument the name of a text file and returns a dictionary whose keys are the words in the given file and its values are the frequencies for each word, i.e. the number of times the word appeared in the text file.

For instance, if you store the previous paragraph into a file and run wordFreq on it, it will return the dictionary:

{'and': 2, 'words': 1, 'appeared': 1, 'text': 2, 'argument': 1, 'Write': 1, 'as': 1, 'are': 2, 'file': 2, 'in': 2, 'its': 2, 'whose': 1, 'given': 1, 'i.e.': 1, 'for': 1, 'file.': 1, 'returns': 1, 'word,': 1, 'function': 1, 'dictionary': 1, 'that': 1, 'keys': 1, 'frequencies': 1, 'number': 1, 'wordFreq': 1, 'a': 3, 'word': 1, 'name': 1, 'of': 2, 'accepts': 1, 'times': 1, 'values': 1, 'each': 1, 'the': 7}

Question 19

Write a function `getDict` that accepts as its argument the name of a CSV file that stores a dictionary and does the following: It opens the file, reads in the lines, creates the dictionary that was stored in the given file and returns that dictionary. The dictionary is stored in the file in the following manner: The first line of the file has the title "Keys, Values". Every other line contains a key and a value separated by a comma. Note that keys and values are both strings.

For instance if the file opps.csv contains the lines:

```
Key,Value
yes,no
front,back
back,front
true,false
right,left
right,wrong
```

The call getDict("opps.csv") will return:

{'front': 'back', 'yes': 'no', 'right': 'wrong', 'true': 'false', 'back': 'front'}

Question 20

Write a function `getDNAfreqs` that accepts as its argument the name of a FASTA file that stores a single DNA sequence and does the following: It opens the file, reads in the sequence, and creates and returns a dictionary with the frequencies of the DNA nucleotides A, C, G, T that appear in the sequence. The dictionary has another entry with the key “other” whose value is the number of characters other that A, C, G, T that appear in the sequence.

[*Hint:* You should use BIopython’s seq and SeqIO, but you may ignore the import part of the program. Just write the function.]