Answers to Practice Questions

Question 1

a. Any task that involves animation of many agents is easier implemented in Scratch than Python. It is more difficult to create graphics and animate them in Python.

Any process that involves extensive calculations or reading/storing data to files is much easier implemented in Python than Scratch. Scratch does not support files and its operations with numerical values are limited.

b. This value can represent any type of data: can be part of an integer or floating point number, can be part of a string, part of a picture, etc.
The program that uses this byte will determine what this sequence of 0s and 1s represents and use it accordingly.

c.
```
88 / 2 = 44 rem  0
44 / 2 = 22 rem  0
22 / 2 = 11 rem  0
11 / 2 =  5 rem  1
 5 / 2 =  2 rem  1
 2 / 2 =  1 rem  0
 1 / 2 =  0 rem  1
```

So, 88 in 8 bits is : 01011000

Question 2

```
x → 5

names → ['', 'Tutu', '', 'Butcher', '', 'Chan']

result → None
```
(the function does not return any value)

Question 3

```
x → 5

names → ['Jones', 'Tutu', 'Aziz', 'Butcher', 'Malkin', 'Chan']

result → ['Malkin', 'Aziz', 'Jones']
```

Question 4

`Extensions` is the same

`Calls` is the same

`s` → `[None, None, 352, None, None, 547, 277, None]`

`n` → `5`

Question 5

The function returns a new list containing the strings in the original list in the same order, but with the following changes:

- Every string of the original list whose length is smaller than the second argument, the string is padded on the right with blanks to make its length equal to the second argument.

Question 6

It returns a new dictionary that is the reverse of its argument. That is, the keys of the new dictionary are the values of the old dictionary and each key m in the new dictionary is associated with a value n, if n is a key in the old dictionary and is associated with a value m in it.

Note that if a value m in the old dictionary is associated with many keys n1, n2, ,…, nk, m will be associated with only one of them in the new dictionary.

Question 7

```
def get_value(old_values, arg):
    if arg in old_values:
        return old_values[arg]
    else:
        new_value = compute(arg)
        old_values[arg] = new_value
        return new_value
```

Question 8

a) Function `bar` returns a list with the characters of its first argument that also appear in its second argument. Characters appear in the result in the same order and multiplicity that appear in the first argument.
Function `foo` returns a dictionary whose keys are the characters that appear in the function's first and second argument. The value associated with each character is the number of times this character appears in the first argument.

b) r1 → {'a': 5, 'i': 1, 'e': 2}

r2 → {'a': 5, 'i': 1, 'e': 3}

Question 9

a)

```
def filter(string1, string2):
    result = []
    for ch in string1:
        if ch not in string2:
            result.append(ch)
    return result
```

b)

```
def percentages(text, alpha):
    result = {}
    length = len(text)
    text = bar(text, alpha)
    for ch in text:
        result[ch] = result.get(ch, 0) + 1
    for key in result:
        result[key] = result[key]/float(length)
    return result
```

Question 10

a) `fun` calculates and returns the median (middle value) of the squares of the values contained in its input list. If the input list is empty, `fun` returns `None`.

b) `fun([ 5, -5, 2, -3, -10 ])` will return 25

Question 11

a)

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

    vals.sort()

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

b)

```
result → 0.0

list → [-10, -5, 0.0, 0.0, 5, 8]
```

Question 12

The two images are different. The new image is a vertical flip of the original image. That is, the left side of the original image is the right side of the new image and the right side of the original image is the left side of the new image. For instance, if dog.jpg is the image on the left, newdog.jpg would be the image on the right:

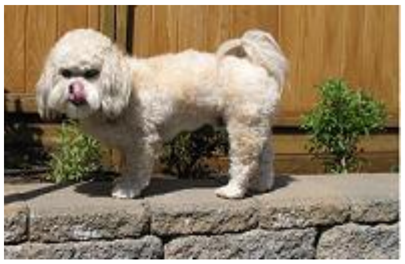
Question 13

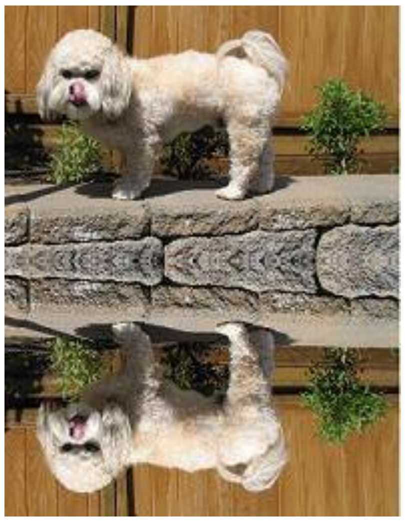
```
def hf(image):
    newimage = Image.new("RGB", image.size)

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

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

Question 14

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

For instance if hm is called with the left image as its argument, it will return the image shown on the right:





Question 15

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

    newsize = tuple([2*w, 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)
    maxw = w-1
    for c in range(w) :
        for r in range(h):
            pix = image.getpixel((maxw-c, r))
            newimage.putpixel((w+c, r), pix)

    return newimage
```

Question 16

students1 → {65: ['Jane', 'John'], 50: ['Jeff'], 90: ['Susan', 'Mary'], 95: ['Lora']}

Note: The pairs in Student1 may be in different order as the dictionary does not maintain any order.

students2 → [(50, ['Jeff']), (65, ['Jane', 'John']), (90, ['Mary’, 'Susan'']), (95, ['Lora'])]

Question 17

students1 → {65: ['Jane', 'John'], 50: ['Jeff'], 90: ['Mary', 'Susan'], 95: ['Lora']}

Note: The pairs in Student1 may be in different order.

students2 → [(50, 'Jeff'), (65, 'Jane'), (65, 'John'), (90, 'Mary'), (90, 'Susan'), (95, 'Lora')]

Question 18

```
def wordFreq(fileName):
    file = open(fileName, "r")
    dict = {}

    for line in file :
        words = line.split()
        for word in words :
            if word in dict:
                dict[word] += 1
            else:
                dict[word] = 1
    file.close()
    return dict
```

Question 19

```
def getDict(fileName):
    file = open(fileName, "r")
    dict = {}

    # Skip first line
    file.readline()

    for line in file :
        line = line.strip()
        words = line.split(",")
        dict[words[0]] = words[1]

    file.close()
    return dict
```

Question 20

```
def getDNAfreqs(fileName):
    record = SeqIO.read(fileName, "fasta")
    dict = {}
    dict["other"] = 0

    for char in str(record.seq) :
        if char in "ACGT" :
            if char not in dict :
                dict[char] = 0
            dict[char] += 1
        else:
            dict["other"] += 1

    return dict
```