Test 1 : Compsci 102/Neuroscience 104

Owen Astrachan
Shani Daily
Jennifer Groh

February 10, 2020

Name: Answer Key (1 point)

NetID/Login: ______________ (1 point)

Community standard acknowledgment (signature) ______________________________

	value	grade
Problem 1	36 pts.	
Problem 2	18 pts.	
Problem 3	12 pts.	
Problem 4	16 pts.	
TOTAL:	84 pts.	

This test has 10 pages, be sure your test has them all. Write your NetID *clearly* on each page of this test.

In writing code you do not need to worry about specifying the proper `import statements`. Don't worry about getting function names exactly right. Assume that all libraries we've discussed are imported in any code you write.

PROBLEM 1 : (***Name/Type/Value (36 points)***)

Consider the following variables and their values in answering the questions below.

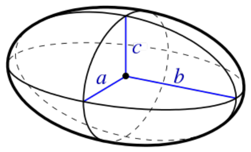
```
words = ["blueberry","banana","orange","cherry","grapefruit","lemon","tangerine"]
sent = "sometimes I lie awake at night and wonder"
```

Each variable in the left column is assigned a value. Provide the type and value of the variable after the assignment. The first two lines have been filled in. Types you can use are **int, float, list, string, boolean.** The *type* is worth two-points, the value is worth one-point.

Assignment	**Type**	**Value**
`a = words[0]`	String	`"blueberry"`
`b = len(sent.split()[1])`	int	1
`c = len(words)`	int	7
`d = sent[-2]`	string	‘e’
`e = len(words) / 2`	float	3.5
`f = 13 // 5`	int	2
`g = len(words[-1]) % 4`	int	1
`h = 4 * 2 + 7`	int	15
`i = words[2:4]`	list	[“orange”, “cherry”]
`j = sent.split()[-2:]`	list	[“and”, “wonder”]
`k = words[1][-3] == words[2][2]`	boolean	True (because ‘a’ == ‘a’)
`l = len(words[-2]) != 10/2`	boolean	True (because 5 == 5.0’)
`m = sent[4:9]`	string	“times”
`n = ’r’ in sent`	boolean	True

PROBLEM 2 : (*Formula I (18 points)*)

An *ellipsoid* is a solid with three axes typically labelled as a, b, c as shown below.



The volume of an ellipsoide is:

$$\frac{4}{3} \times \pi \times a \times b \times c$$

Part A (4 points)

Complete function `eVolume` below to return the volume of an ellipsoid whose axes are given by the parameters a, b, c. For example,

```
x = eVolume(2,3,4)
```

stores 100.5308 in variable x when `eVolume` is implemented correctly because

$$\frac{4 * \pi * 2 * 3 * 4}{3} = 100.5308$$

Use 3.14159 or `math.pi` for the value of π in writing the function.

```
def eVolume(a,b,c):
    """
    a,b,c are float or int values
    return volume of ellipsoid with axes a,b,c
    """


      return 4 * math.pi * a * b * c / 3
```

(continued)

You'll write a function to return the surface area of an ellipsoid using a formula given on the next page. You may find the helper function `expo`, shown below, useful. You'll be asked some questions about this function.

Part B (6 points)

The function `expo` below is a useful helper function in calculating ellipsoidal surface area. Its documentation and code indicate what it does. The table below illustrates some calls and corresponding return values. As a reminder, raising a number to the $\frac{1}{2}$ power is equivalent to taking the square root of the number; and the value of $x^0 == 1$ for all values of x.

call	**return value**	**reason**
`expo(2,3,3)`	216.0	6*6*6 = 216
`expo(5,20,0.5)`	10.0	$\sqrt{5 * 20} = 10$

```
def expo(a,b,exp):
    """
    a,b,exp are float or int values
    return the value of (a*b)^exp, that
    is the product of a and b raised to the exp power
    """
        return (a*b)**exp
```

What is the return value of `expo(2,2,2)`? 16 because (2*2)**2 == 4**2 == 16

What is the return value of `expo(8,8,0.5)`? 8 because expo(y,y,0.5) == y for any y >= 0

What is the return value of `expo(4197,4197,0)`? 1 because expo(x,y,0) == 1 for any x and y

Part C (8 points)

The surface area of an ellipsoide cannot be calculated exactly without using trigonometric functions, but can be approximated reasonably well by the formula below where $p = 1.6075$.

$$4 \times \pi \times \left(\frac{(a \times b)^p + (a \times c)^p + (b \times c)^p}{3} \right)^{\frac{1}{p}}$$

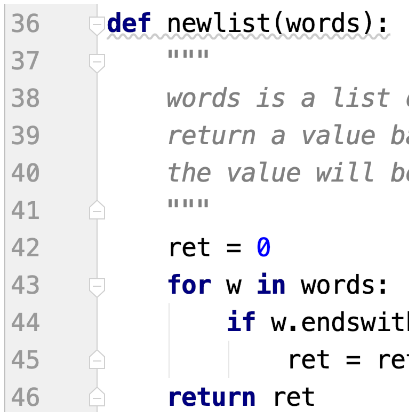
Complete the function `eSurface` so that it returns the surface area of an ellipsoid whose axes are given by the parameters a, b, c. You should use the `return` statement shown which requires assignment to a variable named `area` that will be returned.

You do **not** need to call the helper function `expo` on the previous page, but you'll write less code if you do call it. As a hint, the value of the call `expo(a,b,p)` represents $(a \times b)^p$ in the formula above – note that p is a variable already defined. You can create/use other variables and helper functions.

```
def eSurface(a,b,c):
    """
    a,b,c are float values
    return (approximate) surface area of ellipsoid with these axes
    """

    p = 1.6075


    inner = expo(a,b,p) + expo(a,c,p) + expo(b,c,p)

     area = 4 * math.pi * (inner/3)**(1/p)









    return area
```

PROBLEM 3 : (*Creating New from Old (12 points)*)

In this problem you'll use the function `newlist` shown here. We will call this version the **original int version** since it returns an integer value.

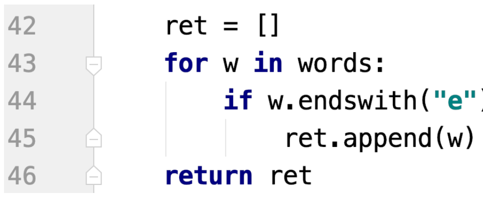


```
36  def newlist(words):
37      """
38      words is a list of strings
39      return a value based on all the strings in words
40      the value will be an int or a list in this test
41      """
42      ret = 0
43      for w in words:
44          if w.endswith("e"):
45              ret = ret + 1
46      return ret
```

The two lines of code below print the single integer 3 since three strings in `words` end with `"e"`.

```
words = ["we", "should", "smile", "more", "on", "this", "day"]
print(newlist(words))
```

If the function `newlist` is changed as shown below, starting on line 42, the same two lines will print `['we, 'smile, 'more']` We call this version the **modified list vesion** since it returns a list.



```
42      ret = []
43      for w in words:
44          if w.endswith("e"):
45              ret.append(w)
46      return ret
```

You'll write different versions of `newlist` by changing line 44 in the function.

Part A (4 points)

Count or add words whose length is greater than four.

Indicate how to modify *only* line 44 of each version so that these two lines

```
words = ["we", "should", "smile", "more", "on", "this", "day"]
print(newlist(words))
```

print the int value 2 for the **original int version** and the list `['should', 'smile']` for the **modified list version** — the function will count or return a list of words whose length is greater than four.

Write the new line 44 below.

if len(w) > 4:

Part B (4 points)

Count or add words whose first and last letters are different.

Indicate how to modify *only* line 44 of each version so that these two lines

```
words = ["emotion", "brainstem", "amygdala", "electrolyte", "neuron"]
print(newlist(words))
```

will print the int value 2 for the **original int version** and the list `['emotion', 'brainstem']` for the **modified list version** because the function will count or return a list of those words that start and end with different letters.

Write the new line 44 below.

if w[0] != w[-1]:

Part C (4 points)

Count or add words that contain exactly four vowels.

Indicate how to modify *only* line 44 of each version so that these two lines

```
words = ["emotion", "brainstem", "amygdala", "electrolyte", "neuron"]
print(newlist(words))
```

will print the int value 2 for the **original int version** and the list `['emotion', 'electrolyte']` for the **modified list version** because the function will count or return a list of those words that have exactly four vowels. You can can call the function `vowelCount` shown here in writing the new line 44.

```
def vowelCount(word):
    """
    word is a string
    returns an int, the number of vowels in word
    """
    t = 0
    for ch in word:
        if ch in "aeiou":
            t = t + 1
    return t
```

Write the new line 44 below.

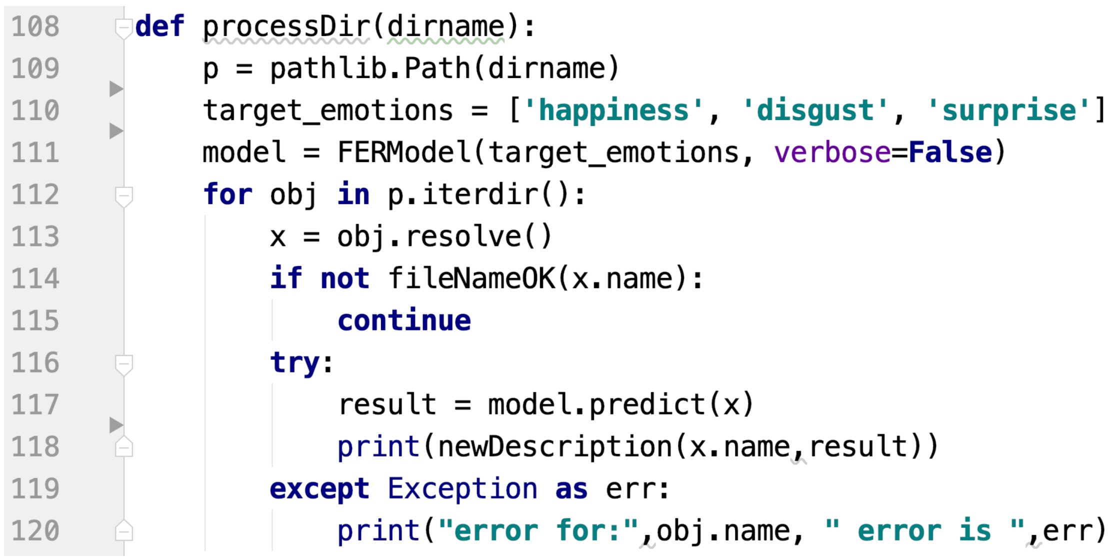
if vowelcount(w) == 4:

if len(w) > 4:

if len(w) > 4:

PROBLEM 4 : (*Moon Site: emotions (16 points)*)

You ran function `processDir` shown below over all the photos/images in a folder as part of finishing lab 3.



```python
def processDir(dirname):
    p = pathlib.Path(dirname)
    target_emotions = ['happiness', 'disgust', 'surprise']
    model = FERModel(target_emotions, verbose=False)
    for obj in p.iterdir():
        x = obj.resolve()
        if not fileNameOK(x.name):
            continue
        try:
            result = model.predict(x)
            print(newDescription(x.name,result))
        except Exception as err:
            print("error for:",obj.name, " error is ",err)
```

The output of the call `processDir("photos")` is shown below on the right when run over the photos you had whose filenames are shown on the left. The identified emotion is not shown here, some of the output has been truncated.

```
18d19sf.jpg          Group 18 has a picture of a 19 year old Asian Female ...
01a18WM.png          Group 01 has a picture of a 18 year old White Male ...
09n18WM.png          Group 09 has a picture of a 18 year old White Male ...
09s18AF.jpg          Group 09 has a picture of a 18 year old Black ...
20f18WM.png          Group 20 has a picture of a 18 year old White Male ...
07a18HM.png          Group 07 has a picture of a 18 year old Hispanic Male ...
19n21HM.JPG          Group 19 has a picture of a 21 year old Hispanic Male ...
0420SF.png           Group 04 has a picture of a 0S year old ERROR ERROR ...
05r18SF.png          Group 05 has a picture of a 18 year old Asian Female ...
19s18SF.png          Group 19 has a picture of a 18 year old Asian Female ...
01d20WF.png          Group 01 has a picture of a 20 year old White Female ...
07d18WF.png          Group 07 has a picture of a 18 year old White Female ...
11f19BF.png          Group 11 has a picture of a 19 year old ERROR Female ...
03h18TM.png          Group 03 has a picture of a 18 year old Two or more races Male ...
03s18SF.png          Group 03 has a picture of a 18 year old Asian Female ...
10r19WF.png          Group 10 has a picture of a 19 year old White Female ...
21n20WF.jpg          Group 21 has a picture of a 20 year old White Female ...
06h36AF.png          Group 06 has a picture of a 36 year old Black ...
02r21AM.png          Group 02 has a picture of a 21 year old Black ...
05a20SF.png          Group 05 has a picture of a 20 year old Asian Female ...
16f18sf.png          Group 16 has a picture of a 18 year old Asian Female ...
```

Part A (6 points)

As shown in the 8th line of output, the filename `0420SF.png` generates the output `Group 04 has a picture of a 0S year old ERROR ERROR ...`

Explain three things: (1) why the first word `ERROR` appears, (2) why the second word `ERROR` appears, and (3) why the three words `0S year old` also appear on that line, which don't correctly identify the age of the person in the photo. Your answer should be brief, one to three sentences is enough. Be sure to explain all three things.

The filename is missing the character that codes an emotion, which should be the third character (index 2) in the filename.

As a result, identifying the gender and ethnicity each generate an error because the indexes are off by one for each of those.

"0S" appears as age because the slice [3:5] is used or age, and because of the missing emotion this yields "0S" instead of the intended "20"

Part B (4 points)

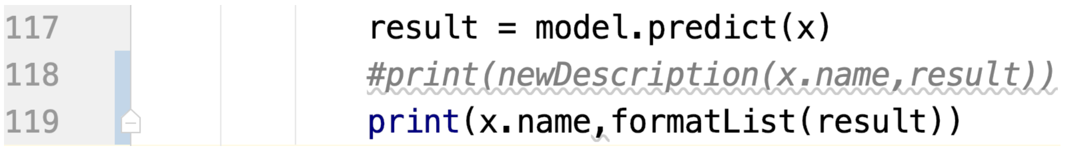
Explain in a sentence or two the **general purpose** of lines 109 and 112. You should explain how these lines are used in the function works as it generates the output shown.

Together these lines allow the code to loop over all the files/objects in a folder/directory.

Line 109 stores a value in p that allows p.iterdir() to be used in the 112 for loop so that obj represents each file/object in the directory indicated by parameter dirname

Part C (6 points)

If line 118 is commented out and a line 119 added as shown here



```
117            result = model.predict(x)
118            #print(newDescription(x.name,result))
119            print(x.name,formatList(result))
```

the output changes as partially shown

```
18d19sf.jpg [[happiness,48.35], [surprise,36.16], [disgust,15.49]]
01a18WM.png [[surprise,76.90], [happiness,23.10], [disgust,0.00]]
...
... (several lines not shown)
...
05a20SF.png [[happiness,75.56], [surprise,17.92], [disgust,6.51]]
16f18sf.png [[happiness,93.81], [disgust,6.19], [surprise,0.00]]
```

You can see in the output that `result` is a list of three elements, each of the three elements is a two-element list (a string and a float), and the three elemnts are sorted by the float value from high to low. The float is the confidence/probability that the photo has the corresponding string emotion.

Fill in the assignments to variables `a` and `b` below on new lines 119 and 120 with expressions using the list `result` so that these four lines change the output as shown

```
(119)  a =   result[0][0]

(120)  b =   result[0][1]

(121)  label = "has emotion " + a +" with confidence "+str(b)+"%"

(122)  print(x.name,label)
```

so that the output changes to:

```
18d19sf.jpg has emotion happiness with confidence 48.35477822059955%
01a18WM.png has emotion surprise with confidence 76.89772471963316%
...
... (several lines not shown)
...
05a20SF.png has emotion happiness with confidence 75.56164418965923%
16f18sf.png has emotion happiness with confidence 93.81421601932611%
```