MR2020 (Summer 2024) Midterm Practice/Review

1. Suppose you have a numpy array that contains the following numbers assigned to a variable 'numbers':

```
array([8.4, 3.6, 1.3, 9.5, 6.4, 3.2, 7.4, 5.5, 4.1, 2.3])
```

Write a for-loop or list comprehension that creates a variable 'count' and ends with 'count' being equal to the number of elements in 'numbers' that are less than 4.

Use the following for-loop:

```
count = 0
for i in numbers:
    if i < 4:
        count += 1
```

or use the below list comprehension:

```
count = len([i for i in numbers if i < 4])
```

2. What would the following return for the variable A?

```
A = 2
for i in range(5):
    A *= 2
```

64

3. Create a list called 'mylist' that contains the numbers 2, 5, and 8.

```
mylist = [2,5,8]
```

4. Create a tuple called 'mytuple' that contains the numbers 8, 4, and 16.

```
mytuple = (8,4,16)
```

5. Create a dictionary called 'mydict' containing the following information:

Key	Value
City	Monterey, CA
Latitude	36.60
Longitude	-121.89

```
mydict = {'City':'Monterey, CA','Latitude':36.60,'Longitude':-121.89}
```

6. Write a Python comparison statement for each of the following English statements. You can assume that numpy was imported as np.

a. A is not equal to B.
`A != B`

b. X is greater than or equal to the mean of numpy array B.
`X >= np.mean(B)` Other possible answer: `X >= B.mean()`

c. X is equal to the maximum value in B.
`X == np.max(B)` Other possible answer: `X == B.max()`

d. A is less than B and B is less than C.
`A < B < C` Other possible answer: `A < B and B < C`

e. Either both A is equal to B and A is less than C or just A is greater than C.
`(A == B and A < C) or (A > C)`

7. Translate the following Python statements to written English.

a. `(A+B) < 2 and A >= 5`

The sum of A and B is less than 2 and A is greater than or equal to 5.

b. `A != B + 1`

A is not equal to B plus 1.

c. `if B < 5: B += 1`

If B is less than 5, then reassign B itself plus 1.

d. `A <= B or A == 0`

A is less than or equal to B or A is equal to zero.

e. `np.mean(A[:,0]) > B`

The numerical mean of A along its 1st column is greater than B.

8. Write a sample code that attempts to assign the variable A an array of zeros with the same dimensions of B using `np.zeros_like(B)`. However, prints "This operation failed" if the variable assignment fails. Assume that numpy is already imported as np.

```
try:
     A = np.zeros_like(B)
except:
     print('This operation failed.')
```

9. Generate a while loop that repeatedly creates a random integer from 1 to 10 and does not stop until the integer is 10. You can use `number = random.randint(1, 10)` to create the random number within the while loop.

```
number = 0
while number != 10:
     number = random.randint(1,10)
```

10. Suppose you have the following arrays of data for temperature (T) and relative humidity (RH) and that numpy is imported as np.

```
T = 30*np.random.random(1000)+5     # degrees C
RH = 100*np.random.random(1000)     # %
```

Write a function called 'myfunc' that returns 1 if the temperature exceeds 20°C and RH exceeds 80% but returns 0 if that condition is not met.

```
def myfunc(T,RH):
     if T > 20 and RH > 80:
          return 1
     else:
          return 0
```

11. Define a class called Cloud that is initialized with the following attributes (with data type for each attribute in parentheses): base (float), height (float), width (float), raining (boolean). Give the class a method (function) that computes the depth (height minus base) of the cloud. Then, in one line, create an object 'mycloud' that is a member of the class 'Cloud' with a base of 500, height of 2500, width of 1000, and with raining assigned False.

```
class Cloud:
     def __init__(self,base,height,width,raining):
          self.base = base
          self.height = height
          self.width = width
          self.raining = raining

     def depth(self):
          return self.height-self.base

cloud = Cloud(500,2500,1000,False)
```

12. For each separate, not related, problem, respond with what the following **bolded red** lines of code would return (or be evaluated as) in Python.

a. **`[j for j in range(5)]`**

[0, 1, 2, 3, 4]

b.

```
A = np.array([(2,3),(4,5)])
A[0,1]
```

3

c.

```
A = np.array([3,5,6,7,8,2,10,34,54,32])
A[4:]
```

```
np.array([8,2,10,34,54,32])
```

```
A = np.array([3,5,6,7,8,2,10,34,54,32])
A[2:5]
```

```
np.array([6,7,8])
```

```
A = np.array([3,5,6,7,8,2,10,34,54,32])
A[::2]
```

```
np.array([3,6,8,10,54])
```

d.

```
weather = {'temp':[43,45,47],'rain':['True','True','False']}
weather['temp'][2]
```

47