A1.

**Answer:** The first path '/tmp/data/images/' is absolute path and the second 'tmp/data/images/' and third './tmp/data/images/' path is relative path.

A2.

**Answer:** The '~' represents root directory and 'echo ~' will print the rood directory which is '/home/nsl3' for me.

A3.

**Answer:** The command 'mkdir {1..10}' will create 10 folders named as '1, 10, 2, 3, 4, 5, 6, 7, 8, 9'

A4.

**Answer:** '-p' is short for --parents, it creates the entire directory tree up to the given directory. But if '-p' is not given it will try to create the given directory and if the direcotry exists then it will give an error.

After the comparison above, it is clear that 'mkdir -p data' or in other words adding '-p' option is the best practice.

A5.

**Answer:** First option: I will copy the image from that directory using command in the terminal and past it somewhere else and then open the image using image viewer. Like,

‘cp file/path/of\_png path/of/destination’ Or

Second option: I will read the image using Pillow or OpenCV and plot the image. Like,

import cv2

import matplotlib.pyplot as plt

img = cv2.imread('/home/nsl3/Documents/question-set-emon/nsl-ra-evaluation-stage-1/datasets/cats\_and\_dogs\_filtered/validation/cats/cat.2000.jpg', 1)

plt.imshow(img)

plt.show()

A6.

**Answer:** Using 'grep' command like grep -nr 'string\_to\_search' file/directory/file\_name.txt

B1.

**Answer:** \*arg passes variable number of non-keyworded arguments list and on which operation of the list can be performed.

\*\*args passes variable number of keyword arguments dictionary to function on which operation of a dictionary can be performed.

B2.

**Answer:** iterator(10, 10) call will simply execute the function and pass

iterator(start=10, 20) will show 'SyntaxError: positional argument follows keyword argument'

iterator(10, end=10) will also pass like the first one

iterator(start=10, end=20) will also pass like the first one

B3.

**Answer:** No, we can not do the assignment operation as the string class is immutable and those operation will raise 'TypeError: 'str' object does not support item assignment'.

B4.

**Answer:** The best way to call the 'get\_address()' method when 'address' is not required is to use '\_' as variable, i.e,

\_ = get\_address()

name, \_ = get\_info()

B5.

**Answer:** The breaking point of the method is when the function 'do\_math(x, y)' will be called by the value of 'y=0' and it will raise 'ZeroDivisionError: division by zero'

I will use try and except block to fix the issue, i.e,

def do\_math(number1, number2):

try:

return number1/number2

except:

print('Divided by Zero error')

C1.

**Answer:** According to the nameing convension and practice of python 3 language

the '\_variable\_name' refers to weakly private variable

the '\_\_variable\_name' refers to strongly private variable

the 'varibale\_name\_' is used when we need to declear a variable which name is same as reserved keywords of python. For example,

if\_ = 5

C2.

**Answer:** Using setitem and getitem magic method of python

# \_\_setitem\_\_, \_\_getitem\_\_

class LaptopBrand:

def \_\_init\_\_(self, floors):

self.\_floors = [None]\*floors

def \_\_repr\_\_(self):

return "Building([{}, {}, {}, {}])".format(\*self.\_floors)

def \_\_setitem\_\_(self, floor\_number, data):

self.\_floors[floor\_number] = data

def \_\_getitem\_\_(self, floor\_number):

return self.\_floors[floor\_number]

# make a building with 3 floors

laptopBrand = LaptopBrand(2)

laptopBrand[0] = 'DELL'

laptopBrand[1] = 'HP'

print(laptopBrand[0])

C3.

**Answer:** 'obj.f()' will print 'f from class A' because it is the first class that is inherited.

C4.

**Answer:** Using magic function call, i.e,

class Layer:

def \_\_init\_\_(self, name):

self.name = name

def ... (self):

pass

def \_\_call\_\_(self, \*args):

#do operations

return self

layer = Layer("custom layer name")

y = layer(image)

C5.

**Answer:** Using super() we can call \_\_init\_\_() method of super class, i.e,

# super class

class Image:

def \_\_init\_\_(self, width, height):

self.width = width

self.height = height

# child class

class FlowerImage(Image):

def \_\_init\_\_(self, width, height, flower\_name):

super().\_\_init\_\_(width, height)

pass

D1.

**Answer:** It depends on the query type on the data stored though but I will mostly use 'dictionary' of python as it stores key and values. If we need to get a single entity then with a unique name or number(may be ID card no.) we can find the entity within O(logn) time complexity.

D2.

**Answer:** Please check the 'recur\_directory\_print.py' file.

D3.

**Answer:** Please check the 'fibonacci\_gen.py' file.

D4.

**Answer:** We can memorize frequent inputs and thus the speed of translation will go higher, i.e,

set\_bn\_text = set()

def check(bangla\_text):

if bangla\_text in set\_bn\_text:

return 'yes'

else:

return 'no'

def predict\_bangla\_text\_to\_english\_text(bangla\_text):

if check(bangla\_text) == 'no':

english\_text = translator.predict(bangla\_text)

set\_bn\_text.add(bangla\_text)

return english\_text

E1.

**Answer:** a. A[1:3, 1:3]

b. A[:, :-1]

c. A[1:, :]

E2.

**Answer:** np.sum(A, axis=-1)

F1.

**Answer:** Please check 'predict\_exp.py' file.

F2.

**Answer:** We can try different methods separately or combination of them to fix this overfitting issue.

First I changed the Optimizer from 'RMSprop' to 'Adam' which gave me a bit good result at the very beginning but after 7 epochs it also tends to overfit.

Then, I lowered the learning rate to '1e-5' which showed a satisfying result as train and val accuracy came close enough.

Again, If we set a dropout layer, it also shows us a great result as two accurcy curve comes much closer.

Please check the ‘solv\_overfit.py’ file. Here, in 'solv\_overfit.py' file, I changed learning rate to 1e-5, optimizer to Adam, and added a dropout layer which reduces both losses and pushes validation accuracy a bit.

Likewise, there are several methods to solve the overfitting issue such as changing learning rate, optimizer, network size. Increasing the amount of data is also a great way to overcome overfit and if we are bound to small dataset then we can augment the data too. Even loss function changing is also a great way to stop overfitting.