==

Python Advanced (like really advanced)

==

Submission Guidelines for assignment

----

1. In the file readme.txt in the team-name directory, which contains the contribution of each team member, roll number and references (cite where you get code/code snippets from).

2. Rename the directory team-name to actual team name instead of
E.g. Coders

3. Compress the directory to <team_name>.tar.gz (nothing more nothing less)
e.g. coders.tar.gz

4. Submit one assignment per team. Please.

General Instructions

- **Unless otherwise stated, you CAN'T use any kind of loops or comprehensions (list, dict, etc).**

- For task0, put your code in appropriate locations in task0.py

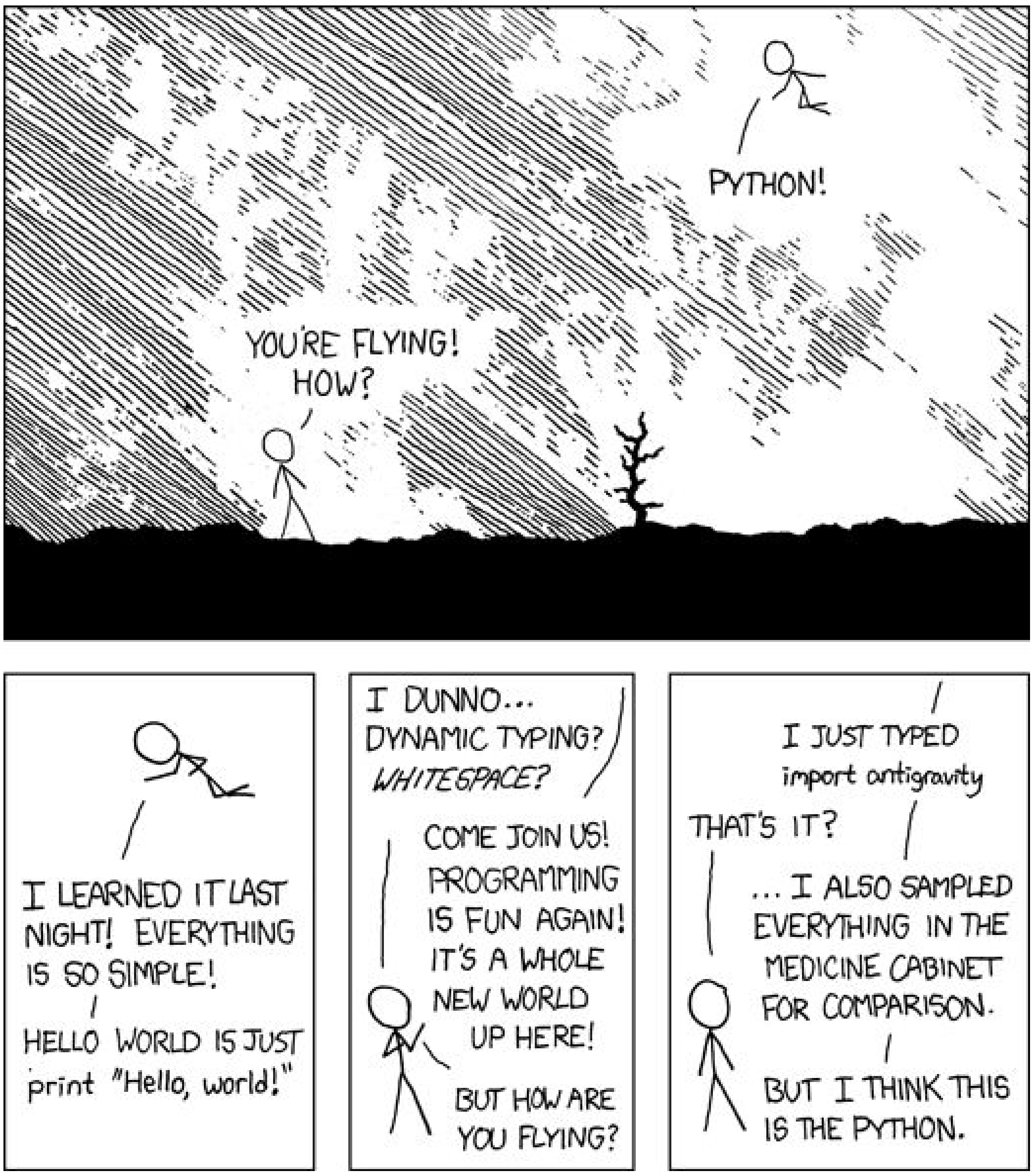
- For other tasks, put your code at appropriate locations in inlab4_tasks.py

- By matrix, we mean two dimensional numpy arrays.

- If we mention array, we mean an n-dimensional numpy array.

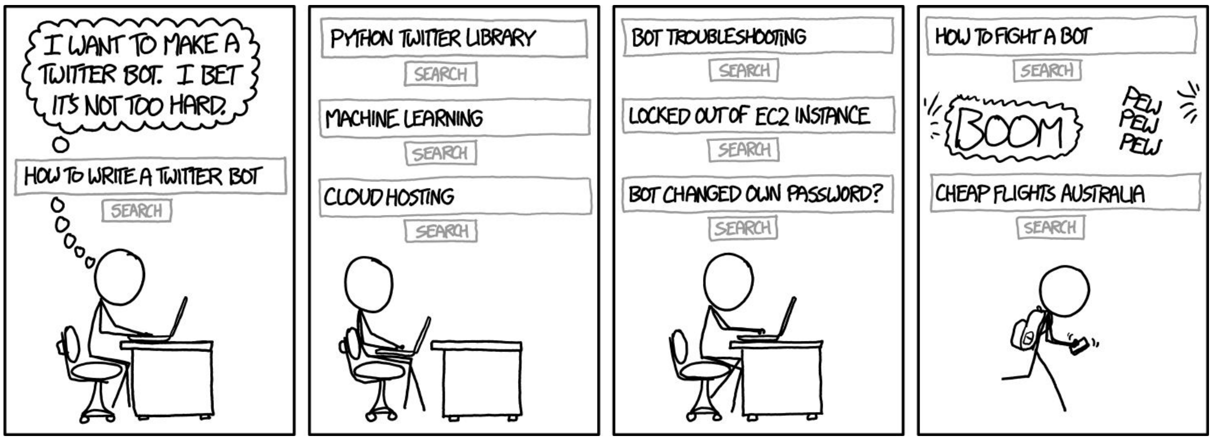
- Make sure you know what you write, you might be asked to explain your code at a later point in time.

- The submission will be graded automatically, so stick to the naming conventions strictly.


PYTHON!
YOU'RE FLYING! HOW?
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

(Source: xkcd)

(Never sample everything from the medicine cabinet! :P)



(Source: xkcd)

(That is what happens at times; this is more of an inspiration to NOT do this!)

****
Task 0
****

Do as directed in task0.py
ProTip: some of the functions that you might find useful for this task are: reshape, argmax, amax, copy, matmul and astype.

****
Task 1
****

Write a function `checkerboard` which accepts a **positive integer** n and returns a $n \times n$ **integer matrix** M which contains checkerboard pattern. Note that M can have 0's and 1's only. Top left corner of M should be 0.

ProTip: just basic indexing should do the trick! Lookup zeros, astype

****
Task 2
****

Write a function `LeakyReLU` which takes a parameter called '`leak`' and an array and computes $g(x)$ for each element x in the array. The function should not change the input array. Input can be either an int or a float array, but, the output must be a float array. Assume that 0 <= `leak` < 1.

$$g(x) = \begin{cases} x & if\ x > 0 \\ leak \times x & otherwise \end{cases}$$

****
Task 3.1
****

Write a function to normalise a matrix of shape (m, n) along its columns. By normalise, we mean that the mean of each column should be 0 and variance of each column should be 1.
To do this, first compute the mean of input along its columns. Subtract the mean from input matrix. Divide the result by its standard deviation along its columns and return it.

ProTip: Useful functions include mean and std. Use reshape wisely and it will become a cakewalk.

****

Task 3.2

****

Write a function `mean_filter` to implement a mean filter of 1-d array of shape (n,) and kernel size $2k + 1$. Output should be a float array.

ProTip: lookup pad, cumsum. You don’t really need a for loop.

****

Task 4

****

Write a function to get the positions of top_k values of each row of a matrix of shape (m,n). Concretely, if A is the input matrix of shape (m,n), then return an integer matrix R, such that: R_{ij} is the index of an element in ith row of A which has rank j when that row is sorted in decreasing order.

For example, if the input array ‘arr’ is:

```
[[ 6  4  4  7]
 [ 7  8  1  0]
 [11 10  5 11]]
```

Then top_k(arr, 2) will be:

```
[[3 0]
 [1 0]
 [3 0]]
```

Explanation for first row: when we sort `6,4,4,7` in decreasing order, we get: `7, 6, 4, 4`. Top 2 elements are `7` and `6`. And, their positions in arr are: `3` and `0` respectively. So, the first row of the result is `3,0`.

Note: for breaking ties, higher indices should come first.(See the last row of result). k is an integer.

ProTip: If you use argsort wisely, then you should be able to do it in 1 line.
(I dare you to do it in 1 line!)

****
Task 5
****

Magic square is a n*n array where sum of each row, each column and each diagonal is same. Write a function `is_magic_square` which takes a matrix of shape (n, n) and returns `True` if the input is a non-trivial magic square and `False` if it isn't. Assume **n is odd**. In a non-trivial magic square, you have to make sure that all the elements are distinct.

****
Task 6
****

The file sml.csv contains 1 header + 7350 rows of data (total 7351 lines). It has 7 columns (as mentioned in the header): instance, algorithm, randomSeed, epsilon, horizon, REG.

You will generate three plots: one for each instance.

The plot will have horizon on the x axis and regret on y axis. Use log scale on both axes. It will contain 7 lines: one for each algorithm (counting epsilon-greedy as three algorithms). For other algos, ignore epsilon. Each point will give the average regret from the fifty random runs at the particular horizon for the algorithm. Make sure you provide a clear key (legend) so the plot is easy to follow. Your plots should contain titles on both axes as well as on the main plot. **You are free to use literally anything in python.**

Name your script as task6.py. We should be able to call it as:

```
python3 task6.py --data sml.csv
--data: the path of data file
```

Your script should produce three plots:
instance1.png, instance2.png and instance3.png. Note that these three plots should be present in your submission directory .

We will run it over different files. These will be different only in terms of horizons and randomSeed. Other things (instances, algorithms, epsilon) will be same.

ProTip: Pandas will be really helpful here. See pandas.DataFrame.groupby.

****
Task 7
****
Write a Python program task.py which processes the numpy matrix present in message.npy. Use linear contrast enhancement in this “decoder” to uncover the message. Use matplotlib to view the images. Save the result in processed image.png (your code should do that automatically). Also, write the key obtained in key.txt

Submit task7.py, processedimage.png, key.txt in the same folder where all other files already are present.

****
That’s all folks!
Quite Simple! Ain’t all this!?
****