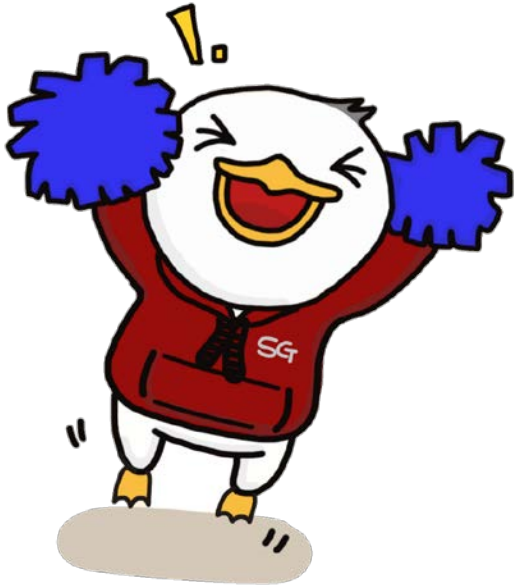
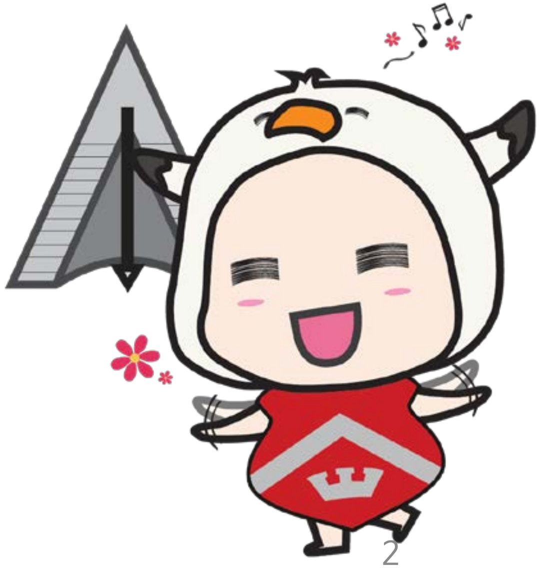
Project #4: Python Programming

Systems Programming
Department of Computer Science and Engineering
Sogang University



Due: May 15 (Wed), 11:59PM (KST)



Goal

1 Goal

The goal of this project is to improve your understanding Python programming.

1. Write all code in Python3.
2. Work with Colaboratory or Jupyter Notebook.
3. Use built-in function for input and output.
   - Inputs to the problem should be obtained from user.
   - Output should be printed on the screen.
4. Solve four problems and make a file (.ipynb) for each problem.

Problems

Problem 1-3 : https://app.codility.com/programmers/lessons/
Problem 4: https://computersciencewiki.org/index.php/Max-pooling_/_Pooling

1 Max counters

You are given N counters, initially set to 0, and you have two possible operations on them:

- increase(X) − counter X is increased by 1,
- max counter − all counters are set to the maximum value of any counter.

A non-empty array A of M integers is given. This array represents consecutive operations:

- if A[K] = X, such that $1 \leq X \leq N$, then operation K is increase(X),
- if A[K] = N + 1 then operation K is max counter.

1 Max counters

For example, given integer N = 5 and array A [3, 4, 4, 6, 1, 4, 4]

The values of the counters after each consecutive operation will be:

(0, 0, 1, 0, 0) → (0, 0, 1, 1, 0) → (0, 0, 1, 2, 0) → (2, 2, 2, 2, 2) →
(3, 2, 2, 2, 2) → (3, 2, 2, 3, 2) → (3, 2, 2, 4, 2)

For the above example, your output will be:

3 2 2 4 2

The objective is to calculate the value of every counter after all operations.

1 Max counters

Input:

5	# N
3 4 4 6 1 4 4	# Array A

Output:

3 2 2 4 2	# Result of counter

2 Common prime divisors

A *prime* is a positive integer X that has exactly two distinct divisors: 1 and X. The first few prime integers are 2, 3, 5, 7, 11 and 13.

A prime D is called a *prime divisor* of a positive integer P if there exists a positive integer K such that D * K = P. For example, 2 and 5 are prime divisors of 20.

You are given two positive integers N and M. The goal is to check whether the sets of prime divisors of integers N and M are exactly the same.

For example, given:

- N = 15 and M = 75, the prime divisors are the same: {3, 5};
- N = 10 and M = 30, the prime divisors aren't the same: {2, 5} is not equal to {2, 3, 5};
- N = 9 and M = 5, the prime divisors aren't the same: {3} is not equal to {5}.

2 Common prime divisors

You pair two elements that have the same index for a given lists A and B.

For example, given:

A = [15, 10, 3]
B = [75, 30, 5]

the result in this input should be 1, because only one pair (15, 75) has the same set of prime divisors.

You must use the datatype “set” to save the prime divisor for each number.

The result of prime divisor 75 must be {3, 5}.

And check whether each pair of prime divisor set has the same common prime divisor.

2 Common prime divisors

Input:

15 10 3	# Array A
75 30 5	# Array B

Output:

1	# Number of set of prime divisors.

3 Min abs sum of two

Let A be a non-empty array consisting of N integers.

The *abs sum of two* for a pair of indices (P, Q) is the absolute value $|A[P] + A[Q]|$, for $0 \le P \le Q < N$.

For example, the following array A:

A = [1, 4, −3]

the result in this input should be 1, as explained next page.

3 Min abs sum of two

For example, the following array A:

A = [1, 4, −3]

has pairs of indices (0, 0), (0, 1), (0, 2), (1, 1), (1, 2), (2, 2).

The abs sum of two for the pair (0, 0) is A[0] + A[0] = |1 + 1| = 2.

The abs sum of two for the pair (0, 1) is A[0] + A[1] = |1 + 4| = 5.

The abs sum of two for the pair (0, 2) is A[0] + A[2] = |1 + (−3)| = 2.

The abs sum of two for the pair (1, 1) is A[1] + A[1] = |4 + 4| = 8.

The abs sum of two for the pair (1, 2) is A[1] + A[2] = |4 + (−3)| = 1.

The abs sum of two for the pair (2, 2) is A[2] + A[2] = |(−3) + (−3)| = 6.

3 Min abs sum of two

Input:

−8 4 5 −10 3 # Array A

Output:

3 # Result of min abs sum of two

|(−8) + 5| = 3

4 Max pooling

You want to compress for a given two-dimensional numpy array.

So, You would like to reduce the size of numpy array by half, with the maximum number for each 2×2 size area as the representative value.

This method is called max pooling.

0.35	0.55	0.99	0.42
0.08	0.95	0.40	0.15
0.06	0.78	0.23	0.47
0.44	0.59	0.94	0.04

2×2 Max Pooling →

0.95	0.99
0.78	0.94

4 Max pooling

For example the following two-dimensional numpy array A:

A = [[0.35 0.55 0.99 0.42]
[0.08 0.95 0.40 0.15]
[0.06 0.78 0.23 0.47]
[0.44 0.59 0.94 0.04]]

the result in this input will be:

B = [[0.95 0.99]
[0.78 0. 94]]

You must use the numpy and datatype “numpy.ndarray” to generate array A, and also use numpy for processing time.

4 Max pooling

Input:

4 # size of the square matrix

Output:

```
[[0.35 0.55 0.99 0.42]          # Randomly generated numpy array
 [0.08 0.95 0.40 0.15]          #     np.random.random((size, size))
 [0.06 0.78 0.23 0.47]
 [0.44 0.59 0.94 0.04]]
[[0.95 0.99]                    # Result of max pooling about array
 [0.78 0.94]]
```

Submission

1

Things

(1) Jupyter notebook files with outputs

- Do not clear outputs. You must have outputs.
- You can download @ File 〉 Download .ipynb
- Each problem code named "p1.ipynb", The numeric part should be **problem number**.

(2) A document file

- This document file should describe how you implemented your programs.
- When you write your document file, think like this: a new employee needs to take over your work and extend your programs.
- He/she should be able to understand your code and be able to work on it after reading your documents.
- A sample document will be posted on cyber campus.

2 Instructions

- Make a directory named "sp20161234_proj4". The numeric part should be **your student ID**.
- Put all the files in the directory, and compress the directory itself using tar or zip.
- When you make a tar file, do NOT use the z option (which makes a gz compressed file.)

Example:

sp20161234_proj4/
  document.docx
  p1.ipynb
  p2.ipynb
  p3.ipynb
  p4.ipynb

2 Instructions

The file for submission

sp20161234_proj4.tar or sp20161234_proj4.zip

Upload this file on the cyber campus.

Late Submission

Late submissions are accepted for five days after the deadline. 10% of the points are deducted for each day. Submissions are not accepted after five days.