

AI VIET NAM – COURSE 2022

Python Basic 2– Exercise

Ngày 11 tháng 5 năm 2022

1. Viết function nhận input vào là 1 chuỗi ký tự (string), biến đổi string theo yêu cầu print ra màn hình.

- Input: input_string là một chuỗi ký tự bất kỳ
- Output: print ra theo điều kiện sau
 - Kiểm tra xem nếu chuỗi ký tự chỉ chứa số nguyên dương thì print "The string is number only" và thoát hàm
 - Nếu không chỉ chứa số nguyên dương thì print ra string theo các yêu cầu sau:
 - * Các chữ cái trong string đều là dạng viết thường
 - * Các chữ cái trong string đều viết hoa
 - * Chỉ chữ cái đầu tiên mỗi từ viết hoa

```
1 # Examples
2 problem1(input_string='2022')
3 >> The string is number only
4
5 problem1(input_string='be what you wanna be Darin-2005')
6 >> be what you wanna be darin-2005
7 BE WHAT YOU WANNA BE DARIN-2005
8 Be What You Wanna Be Darin-2005
9
10
11 problem1(input_string='The purpose of our lives is to be happy')
12 >> the purpose of our lives is to be happy
13 THE PURPOSE OF OUR LIVES IS TO BE HAPPY
14 The Purpose Of Our Lives Is To Be Happy
```

Code Listing 1: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ

2. Cho một chuỗi coded_str chứa các ký tự đã được mã hóa theo cách sau k[string], trong đó k là số lần string trong dấu [] được lặp lại. Điều kiện: coded_str là các chuỗi các ký tự tiếng Anh và string chỉ chứa các ký tự [a-z] và [A-Z] không có ký tự đặc biệt và không phải là số. Nếu là số thì bắt buộc phải là k và đứng trước []. Viết function giải mã coded_str và trả về string đã được giải mã.

- Note: k trong range [1, 9]
- Note: Giả sử coded_str luôn nhập đúng theo yêu cầu (Đơn giản hóa vấn đề để các bạn không cần check nhiều trường hợp lỗi)
- Input: coded_str là string đã được mã hóa bởi k và []

- Output: Trả về 1 string được giải mã bằng cách lặp lại k lần các string trong [] và giữ nguyên các ký tự còn lại
 - Input: coded_str = "2[abc]3[cd]ef"
 - Output: "abcabccdcddcdef"
 - Input: coded_str = "abc3[cd]xyz"
 - Output: "abccdcddcdxyz"

```

1 # Examples
2 result = decode_string(coded_str="2[abc]3[cd]ef")
3 print(result)
4 >> abcabccdcddcdef
5
6 result = decode_string(coded_str="abc3[cd]xyz")
7 print(result)
8 >> abccdcddcdxyz

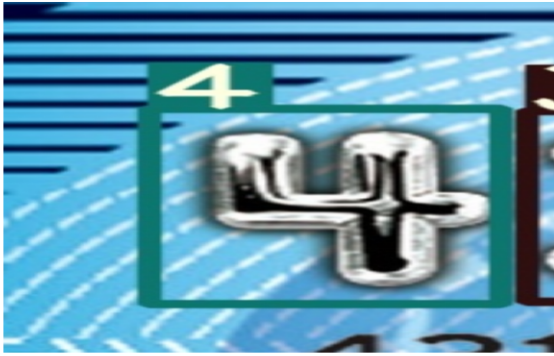
```

Code Listing 2: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ

- Viết 3 functions hỗ trợ việc thực hiện xử lý thông tin label trong file .txt của task object detection.:

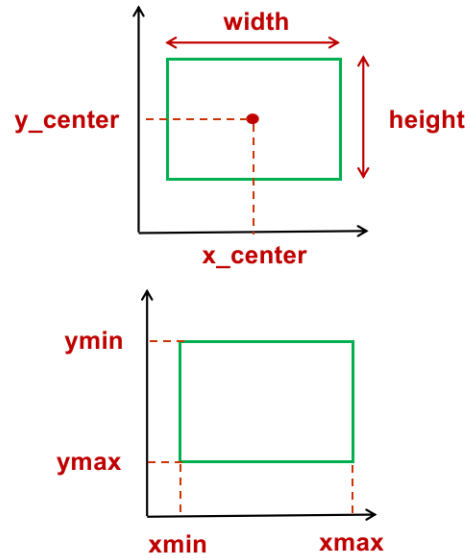


Hình 1: Thông tin trong label (8.txt) giúp vẽ bounding box cho các chữ số trên card



Bounding box: là hình chữ nhật được vẽ bao quanh 1 object nhằm xác định tọa độ của object trong ảnh
 - VD: bounding box là hình chữ nhật màu xanh lá cây đậm bao quanh object (số 4)

Tọa độ bounding box thường được biểu diễn bằng 2 loại:
 - Loại 1: x_center , y_center , width và height
 - Loại 2 (tlbr): $xmin$, $ymin$, $xmax$, $ymax$
 Hay còn gọi là: x_tl , y_tl , x_br , y_br



Hình 2: Giải thích về bounding box và các cách biểu diễn tọa độ bounding box

	id_class	x_center	y_center	width	height
1	4	0.1104870557261957	0.6189416670904745	0.045474490406478114	0.1156995372018512
2	3	0.15636044517132713	0.6189416670904745	0.039889903865331686	0.113156690230382
3	1	0.19944154134588535	0.6233916492905457	0.03510311540149186	0.10679957280170883
4	1	0.239331445211217	0.6202130905762091	0.04308109617455815	0.11061384325891271
5	7	0.3219035462124536	0.6183059553476071	0.0454744904064781	0.09917103188730106
6	8	0.3669791375802784	0.6189416670904745	0.035103115401491886	0.10044245537303566
7	0	0.40846463760022333	0.6214845140619437	0.03669871155610512	0.10552814931597407
8	1	0.4507000678128367	0.620136805167065	0.0430810961745582	0.11061384325891271
9	1	0.5384578563165664	0.6226796521385343	0.0430810961745582	0.11061384325891271

9 lines đầu tiên trong file 8.txt

Mỗi line sẽ chứa 5 thông tin và cách nhau = ' ' (1 space)

- **id_class**: id của class mà object nằm trong bounding box
- **x_center**: tọa độ trung tâm theo trục hoành (x) của bounding box
- **y_center**: tọa độ trung tâm theo trục tung (y) của bounding box
- **width**: chiều dài của bounding box
- **height**: chiều cao của bounding box

=> Lưu ý các tọa độ này đã được scale trong range [0,1]

Hình 3: Nội dung file 8.txt mỗi line chứa thông tin bounding box theo loại 1

```

1 # Given
2 # Download 8.txt file to /content/8.txt in google colab
3 !gdown --id 12LGCzELva7ZuNtAQ3xmdEPHscudsx78U
4
5 # Download 861.txt file to /content/861.txt in google colab
6 !gdown --id 1JevMcivRRhsN9EGz0lGltXG9erhEWyut
7
8 # example which shows content of 1 line in txt file (id, x_center, y_center,
   width, height)
9 example = '4 0.1104870557261957 0.6189416670904745 0.045474490406478114
   0.1156995372018512'
10 example.split(' ')
11 >> ['4', '0.1104870557261957', '0.6189416670904745', '0.045474490406478114', '
   0.1156995372018512']
12
13
14 # example which shows how to extract id_class, x_center, y_center, width, height
   in 1 line of txt file
15 id_class, x_center, y_center, width, height = example.split(' ')
16 print(f'id_class = {id_class}')
17 print(f'x_center = {x_center}')
18 print(f'y_center = {y_center}')
19 print(f'width = {width}')
20 print(f'height = {height}')
21 >> id_class = 4
22 x_center = 0.1104870557261957
23 y_center = 0.6189416670904745
24 width = 0.045474490406478114
25 height = 0.1156995372018512
26
27 # configuration
28 file1 = '/content/8.txt'
29 real_coor_file1 = '/content/8real.txt'
30 tl_br_coor_file1 = '/content/8tlbr.txt'
31 test_file1 = '/content/8test.txt'
32 height1 = 1484
33 width1 = 2365
34
35 file2 = '/content/861.txt'
36 real_coor_file2 = '/content/861real.txt'
37 tl_br_coor_file2 = '/content/861tlbr.txt'
38 test_file2 = '/content/861test.txt'
39 height2 = 541
40 width2 = 782

```

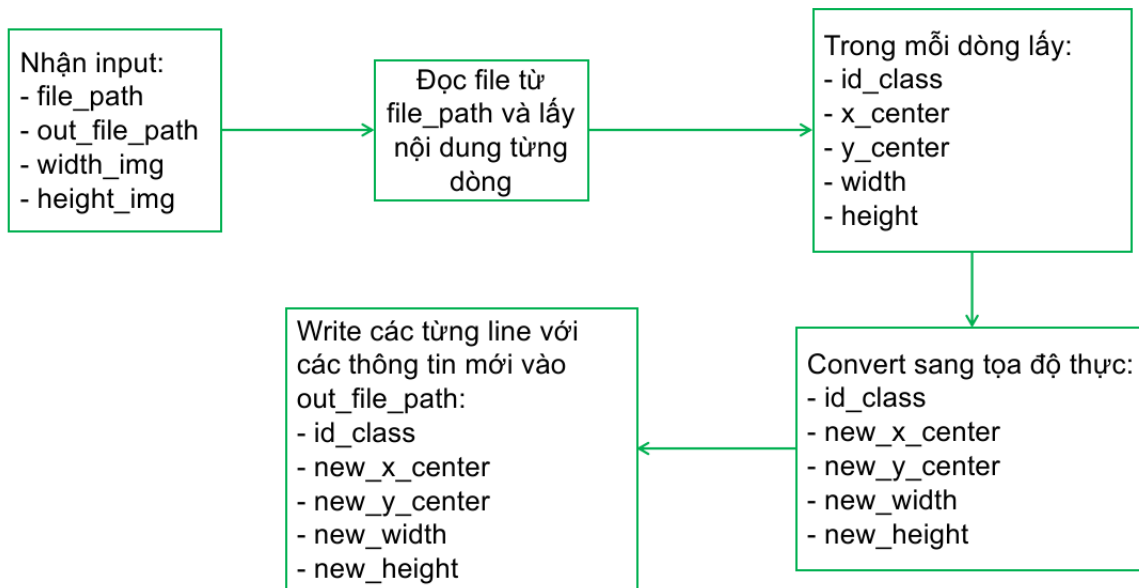
Code Listing 3: Cho trước các thông tin sau

Note Code Listing 3

- Các bạn nhập và chạy line 3, line 6 để download file 8.txt đến đường dẫn /content/8.txt và download file 861.txt đến đường dẫn /content/861.txt trong google colab
- line 9-25 là ví dụ cho thấy nội dung của 1 line trong file txt và cách lấy id_class, x_center, y_center, width và height của bounding box
- line 28-40 là configuration để chạy bài tập này. Các bạn cần nhập và chạy line 28-40 trong google colab
- Nếu các bạn không code theo như Code Listing 3 được thì dùng template sau [link](#)

Yêu cầu đề bài

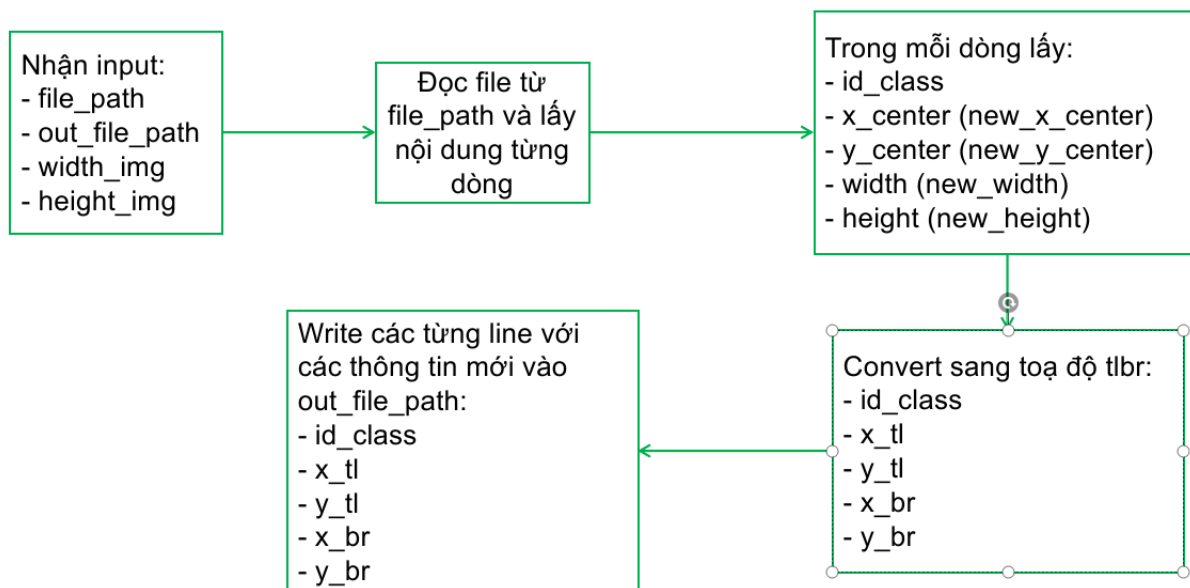
- **Viết function `show_content`** nhận input là đường dẫn file txt và print ra từng line nội dung trong file txt
 - Input: `file_path` đường dẫn đến file txt (đã được định nghĩa trong phần configuration)
 - Output: print ra từng line nội dung trong file txt
- **Viết function `write_real_coor_file`** nhận đường dẫn file label txt input, đường dẫn để write file txt output, `width_img` và `height_img` của ảnh. Function thực hiện đọc file label txt input và lấy `id_class`, `x_center`, `y_center`, `width` và `height` của bounding box (từng line trong file txt input). Sau đó convert sang tọa độ thực và write các tọa độ này thành từng dòng tương ứng vào file txt theo đường dẫn output



Hình 4: Các bước trong function `write_real_coor_file`

- Input:
 - * **`file_path`**: đường dẫn file label txt input (dùng để đọc từng line và lấy `id_class`, tọa độ, ...)(đã được định nghĩa trong phần configuration)
 - * **`out_file_path`**: đường dẫn file sau khi convert tọa độ sẽ write kết quả ra file này (đã được định nghĩa trong phần configuration)
 - * **`width_img`**: chiều dài ảnh (đã được định nghĩa trong phần configuration)
 - * **`height_img`**: chiều cao ảnh (đã được định nghĩa trong phần configuration)
- Output: write 1 file txt với nội dung là các tọa độ đã được convert sang tọa độ thực.
- **Công thức convert:**
 - * $\text{new_x_center} = \text{x_center} * \text{width_img}$
 - * $\text{new_y_center} = \text{y_center} * \text{height_img}$
 - * $\text{new_width} = \text{width} * \text{width_img}$
 - * $\text{new_height} = \text{height} * \text{height_img}$
- Sau đó write các `id_class`, `new_x_center`, `new_y_center`, `new_width`, `new_height` vào file output txt với các dòng tương ứng từ file input txt

- Viết function `write_tlbr_coor_file` nhận đường dẫn file label txt input (file output từ function `write_real_coor_file`), đường dẫn để write file txt output, `width_img` và `height_img` của ảnh. Function thực hiện đọc file txt input (file output từ function `write_real_coor_file`) và lấy `id_class`, `x_center`, `y_center`, `width` và `height` của bounding box (từng line trong file txt input). Sau đó convert sang tọa độ tlbr và write các tọa độ này thành từng dòng tương ứng vào file txt theo đường dẫn output



Hình 5: Các bước trong function `function write_tlbr_coor_file`

- Input:
 - * **file_path:** đường dẫn file label txt input (file output từ function `write_real_coor_file`) (dùng để đọc từng line và lấy `id_class`, tọa độ, ...)(đã được định nghĩa trong phần configuration)
 - * **out_file_path:** đường dẫn file sau khi convert sang tlbr tọa độ sẽ write kết quả ra file này (đã được định nghĩa trong phần configuration)
 - * **width_img:** chiều dài ảnh (đã được định nghĩa trong phần configuration)
 - * **height_img:** chiều cao ảnh (đã được định nghĩa trong phần configuration)
- Output: write 1 file txt với nội dung là các tọa độ đã được convert sang tọa độ tlbr.
- Công thức convert:

$$* x_tl = \frac{1}{width_img} * (x_center - \frac{width}{2})$$

$$* y_tl = \frac{1}{height_img} * (y_center - \frac{height}{2})$$

$$* x_br = \frac{1}{width_img} * (x_center + \frac{width}{2})$$

$$* y_br = \frac{1}{height_img} * (y_center + \frac{height}{2})$$
- Sau đó write các `id_class`, `x_tl`, `y_tl`, `x_br`, `y_br` vào file output txt với các dòng tương ứng từ file input txt

```

1 #Example
2 # show_content function
3 show_content(file1)
4 >>
5 4 0.1104870557261957 0.6189416670904745 0.045474490406478114 0.1156995372018512
6
7 3 0.15636044517132713 0.6189416670904745 0.039889903865331686 0.113156690230382
8
9 1 0.19944154134588535 0.6233916492905457 0.03510311540149186 0.10679957280170883
10 ...
11
12 # write_real_coor_file function
13 write_real_coor_file(file_path=file1,
14                       out_file_path=real_coor_file1,
15                       width_img=width1,
16                       height_img=height1)
17 show_content(real_coor_file1)
18 >>
19 4 261.3018867924528 918.5094339622641 107.54716981132074 171.69811320754718
20
21 3 369.79245283018867 918.5094339622641 94.33962264150944 167.92452830188688
22
23 1 471.6792452830189 925.1132075471698 83.01886792452824 158.4905660377359
24 ...
25
26 # write_tlbr_coor_file function
27 write_tlbr_coor_file(file_path=real_coor_file1,
28                       out_file_path=tl_br_coor_file1,
29                       width_img=width1,
30                       height_img=height1)
31 show_content(tl_br_coor_file1)
32 >>
33 4 0.08774981052295665 0.5610918984895488 0.13322430092943474 0.6767914356914001
34
35 3 0.13641549323866128 0.5623633219752834 0.17630539710399298 0.6755200122056655
36
37 1 0.18188998364513945 0.5699918628896914 0.21699309904663128 0.6767914356914001
38 ...

```

Code Listing 4: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ

4. Viết function lựa chọn regression loss function để tính loss :

- $MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$
- $MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$
- $RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$
- **n** chính là **số lượng samples (num_samples)**, với **i** là mỗi sample cụ thể. Ở đây các bạn có thể hiểu là cứ mỗi **i** thì sẽ có **1 cặp y_i là target và \hat{y} là predict.**
- Input:
 - Người dùng **nhập số lượng sample (num_samples)** được tạo ra (chỉ nhận integer numbers)
 - Người dùng **nhập loss name (MAE, MSE, RMSE-(optional))** chỉ cần MAE và MSE, bạn nào muốn làm thêm RMSE đều được.

- Output: Print ra **loss name, sample, predict, target, loss**
 - **loss name:** là loss mà người dùng chọn
 - **sample:** là thứ tự sample được tạo ra (ví dụ num_samples=5, thì sẽ có 5 samples và mỗi sample là sample-0, sample-1, sample-2, sample-3, sample-4)
 - **predict:** là số mà model dự đoán (chỉ cần dùng random tạo random một số trong range [0,10))
 - **target:** là số target mà mong muốn mode dự đoán đúng (chỉ cần dùng random tạo random một số trong range [0,10))
 - **loss:** là kết quả khi đưa predict và target vào hàm loss
 - **note:** ví dụ num_sample=5 thì sẽ có 5 cặp predict và target.

Note: Các bạn lưu ý

- Dùng **.isnumeric()** method để kiểm tra **num_samples** có hợp lệ hay không (vd: x='10', num_samples.isnumeric() sẽ trả về True ngược lại là False). Nếu **không hợp lệ print 'number of samples must be an integer number'** và dừng chương trình.
- Dùng vòng lặp **for**, lặp lại **num_samples** lần. Mỗi lần dùng **random modules** tạo một con số ngẫu nhiên trong range **[0.0, 10.0)** cho **predict** và **target**. Sau đó đưa predict và target vào loss function và print ra kết quả mỗi lần lặp.
- Dùng **random.uniform(0,10)** để tạo ra một số ngẫu nhiên trong range [0,10)
- Giả xử người dùng luôn nhập đúng loss name **MSE, MAE, và RMSE** (đơn giản bước này để các bạn không cần check tên hợp lệ)
- Dùng **abs()** để tính trị tuyệt đối ví dụ **abs(-3)** sẽ trả về 3
- Dùng **math.sqrt()** để tính căn bậc 2

```

1 exercise4()
2 >> Input number of samples (integer number) which are generated: 5
3 Input loss name: RMSE
4 loss name: RMSE, sample: 0, pred: 6.659262156575629, target: 4.5905830130732355,
   loss: 4.279433398761796
5 loss name: RMSE, sample: 1, pred: 4.592264312227207, target: 8.447168720237958,
   loss: 14.860287994900718
6 loss name: RMSE, sample: 2, pred: 8.701801828625959, target: 9.280646891626386,
   loss: 0.3350616069599687
7 loss name: RMSE, sample: 3, pred: 4.799972972282257, target: 9.877147335937869,
   loss: 25.777699518961764
8 loss name: RMSE, sample: 4, pred: 0.20159822778697878, target: 5.540221923628147,
   loss: 28.50090296579681
9 final RMSE: 3.8406610234536727
10
11
12 exercise4()
13 >> Input number of samples (integer number) which are generated: aa
14 number of samples must be an integer number

```

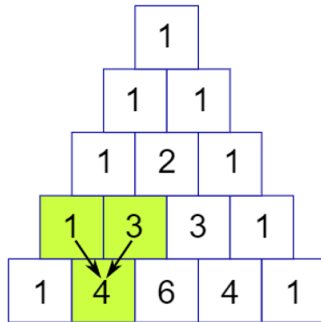
Code Listing 5: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ

5. (OPTIONAL) Viết các function thực hiện tam giác Pascal và dãy số Fibonacci.

Pascal's Triangle

- Input: level là số lượng hàng
- Output: Print ra kết quả từng hàng theo Pascal's Triangle

Pascal's Triangle



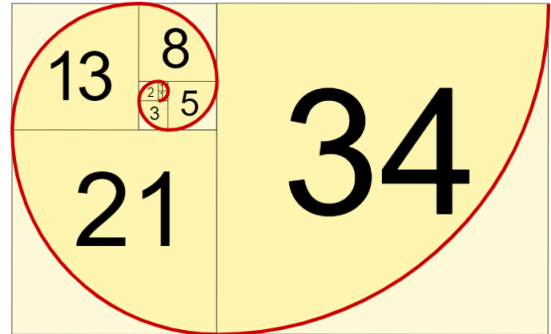
Level = 5

<https://www.mathsisfun.com>

Fibonacci Sequence

0, 1, 1, 2, 3, 5, 8, 13, 21

Length = 9



Fibonacci Sequence

- Input: length là độ dài của chuỗi Fibonacci
- Output: Print ra kết quả các elemet trong chuỗi Fibonacci
- Các bạn nên tìm hiểu các công thức có thể thực hiện được Pascal's Triangle và Fibonacci Sequence
- Nếu các bạn có kiến thức về list hoặc các kiến thức khác chưa có trong bài học thì các bạn vẫn sử dụng được

NOTE: Các bạn có thể dùng function in như sau để in kết quả trên cùng 1 hàng

```
1 value1 = 10
2 value2 = 20
3 value3 = 30
4 print(value1, end=' ')
5 print(value2, end=' ')
6 print(value3, end=' ')
7
8 >> 10 20 30
```

Code Listing 6: Print value trên cùng 1 hàng

```
1 calc_pas(level=5)
2 >> 1
3 1 1
4 1 2 1
5 1 3 3 1
6 1 4 6 4 1
7
8 calc_fib(length=9)
9 >> Fibonacci sequence:
10 0 1 1 2 3 5 8 13 21
```

Code Listing 7: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ

$$\sin(x) \approx \sum_{n=0}^{\infty} (-1)^n \frac{x^{(2n+1)}}{(2n+1)!} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \dots$$

$$\cos(x) \approx \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n}}{(2n)!} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \frac{x^{10}}{10!} + \dots$$

$$\sinh(x) \approx \sum_{n=0}^{\infty} \frac{x^{(2n+1)}}{(2n+1)!} = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \frac{x^7}{7!} + \frac{x^9}{9!} + \dots$$

$$\cosh(x) \approx \sum_{n=0}^{\infty} \frac{x^{2n}}{(2n)!} = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \frac{x^6}{6!} + \frac{x^8}{8!} + \frac{x^{10}}{10!} + \dots$$

6. (OPTIONAL) Viết 4 functions để ước lượng các hàm số sau.

- Input: x (số muốn tính toán) và n (số lượng bậc muốn xấp xỉ)
- Output: Kết quả ước lượng hàm tương ứng với x. Ví dụ hàm $\cos(x=0)$ thì output = 1

NOTE: Các bạn chú ý các điều kiện sau

- x là radian
- n là số nguyên dương ≥ 0
- các bạn nên viết một hàm tính giai thừa riêng

```

11 approx_sin(x=3.14, n=10)
12 >> 0.0015926529267151343
13
14 approx_cos(x=3.14, n=10)
15 >> -0.9999987316527259
16
17 approx_sinh(x=3.14, n=10)
18 >> 11.53029203039954
19
20 approx_cosh(x=3.14, n=10)
21 >> 11.573574828234543

```

Code Listing 8: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ

7. (OPTIONAL) Cho một số nguyên dương n, viết phương trình đảo ngược thứ tự các vị trí trong số n. Chỉ dùng while (hay for) và những phép toán cơ bản như +, -, *, /, %, // ...

- Input: n là một dãy số nguyên dương.
- Output: Đảo ngược vị trí các số trong n. Ví dụ input: 12345678910, output: 1987654321

NOTE: Các bạn chú ý các điều kiện sau

- Không được ép kiểu sang string

- Chỉ sử dụng while hoặc for loop

```
22 reverse_number(n=12345678910)
23 >> 1987654321
24
25 reverse_number(n=123456789)
26 >> 987654321
```

Code Listing 9: Đây là các ví dụ các bạn không cần thiết đặt tên giống ví dụ