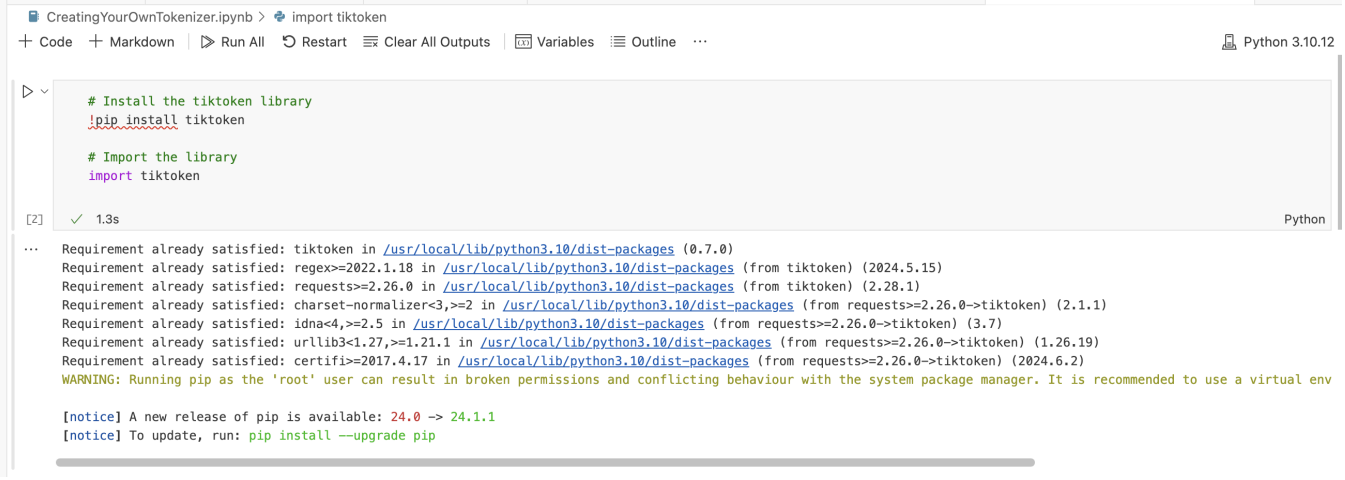
04-02-Create your own tokenizer

Youtube link
Attached the notebook for more info

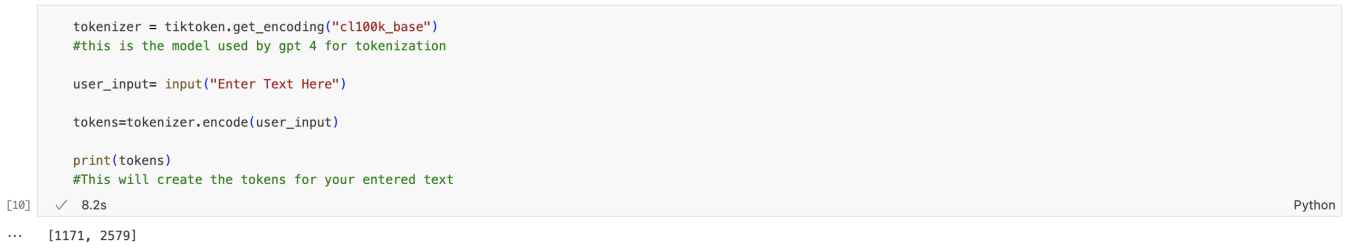
This exercise is just to create your own tokenizer to understand how it works. (Although the internal working is still blackbox, but it gives a better idea)

You can run this in your jupyter notebook or can try go/sandbox and run there.

1- Import tikoken package.
Official documentation: https://github.com/openai/tiktoken (go through its readme to understand better)



2- Use the below code to create the token encoding.



3- Now you can also use the same token to decode it as shown below

```
    decode = tokenizer.decode(tokens)
    print(decode)
```

0.0s Python

```
This is sample text . Lets see the tokens
```

4- In the above one, we can't see how the text was tokenized. so we will use another function (decode_tokens_byte) to understand how text was converted to token and whether it was split while tokenizing.

```
    user_input=input("")
    tokens = tokenizer.encode(user_input)
    decode_to_bytes = tokenizer.decode_tokens_bytes(tokens)
    print(tokens)
    print(decode_to_bytes)

    #Tokenize this sentence and showcase the requirement . You'll see space is not a token itself but appended with the words.
    # Also you'll see how thw words have been broken down.
```

10.9s Python

```
[5319, 2779, 553, 420, 11914, 323, 35883, 279, 2612]
[b'TO', b'ken', b'ize', b' this', b' sentence', b' and', b' showcase', b' the', b' output']
```

5- Created a sample code to highlight the tokenized text and showcase the token and character count .

```
encode = tokenizer.encode(user_input)
decode = tokenizer.decode_tokens_bytes(encode)

for token in decode:
    token_list.append(token.decode())

character_count = sum(len(i) for i in token_list)
length = len(encode)
#This is basically used for coloring each token and repeat the color after 6.
for tk in token_list:
    if count == 0:
        print('\x1b[0;47;1m' + tk + '\x1b[0m', end='')
        count+=1
    elif count == 1:
        print('\x1b[0;42;1m' + tk + '\x1b[0m', end='')
        count+=1
    elif count == 2:
        print('\x1b[0;43;1m' + tk + '\x1b[0m', end='')
        count+=1
    elif count == 3:
        print('\x1b[0;44;1m' + tk + '\x1b[0m', end='')
        count+=1
    elif count == 4:
        print('\x1b[0;46;1m' + tk + '\x1b[0m', end='')
        count+=1
    elif count == 5:
        print('\x1b[0;45;1m' + tk + '\x1b[0m', end='')
        count=0

print("\n\n" + str(token_list) +"\n")
print(str(encode) + "\n")
print("Token Count: " + str(length))
print("Characters: " + str(character_count))
```

18.0s

```
This is a longer tokenization and this sentence and required to understand the output. Also we will color each sentence

['This', ' is', ' a', ' sample', ' token', 'ization', ' of', ' this', ' sentence', ' as', ' required', ' to', ' understand', ' the', ' ou', 'put', '.', ' Also', ' we', ' will', ' color', ' this', ' sentence']

[2028, 374, 264, 6205, 4037, 2065, 315, 420, 11914, 439, 2631, 311, 3619, 279, 6033, 631, 13, 7429, 584, 690, 1933, 420, 11914]

Token Count: 23
Characters: 116
```