Problem 2 -Python

Generating files:

```
import random
import string
from uuid import uuid4


def get_file_name(path='generated_files/') -> str:
    return path + str(uuid4()) + '.txt'


def char_generator(size=1000, chars=string.ascii_uppercase) -> str:
    return ''.join(random.choice(chars) for _ in range(size))


if __name__ == '__main__':
    for _ in range(100):
        with open(get_file_name(), 'w+') as f:
            f.write(char_generator())
```

Checking palindrome serially

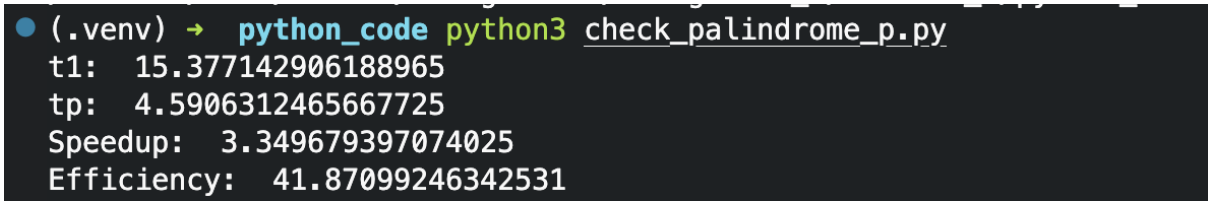
```
import glob


def check_palindrome(s, _=None):
    '''
    This function will check if string 's' has palindrome of length 5,
    if so it returns 1 or else it returns 0
    '''
    for i in range(len(s)):
        for j in range(i+1, len(s)+1):
            temp = s[i:j]
            # Checking palindrome and if length is 5
            if (temp == temp[::-1]) and (len(temp) == 5):
                return 1
    return 0


def get_all_files(path='generated_files'):
    '''
    This functions returns all the file (extention .txt) inside a folder(path)
```

```
    '''
    return [file for file in glob.glob(path + "/*.txt")]


def get_string_from_file(file):
    '''
    This function returns all the text inside a file
    '''
    with open(file, 'r') as f:
        return f.read().rstrip()


if __name__ == '__main__':
    for file in get_all_files():
        print(check_palindrome(get_string_from_file(file=file)))
```

Checking palindrome using parallel processing

```
import multiprocessing as mp
import time

from check_palindrome import (check_palindrome, get_all_files,
                              get_string_from_file)

if __name__ == '__main__':
    start_time = time.time()

    for file in get_all_files():
        _ = check_palindrome(get_string_from_file(file=file))

    t1 = time.time() - start_time

    list_string = [get_string_from_file(file) for file in get_all_files()]

    start_time = time.time()
    pool = mp.Pool(mp.cpu_count())
    result = pool.starmap(
        check_palindrome,
        [(s, None) for s in list_string])
    pool.close()

    tp = time.time() - start_time

    print("t1: ", t1)
    print("tp: ", tp)

    speed_up = t1/tp
    efficiency = (speed_up/mp.cpu_count()) * 100

    print("Speedup: ", speed_up)
    print("Efficiency: ", efficiency)
```

Output:


(.venv) ➜ python_code python3 check_palindrome_p.py
t1: 15.377142906188965
tp: 4.5906312465667725
Speedup: 3.349679397074025
Efficiency: 41.87099246342531