

Nama : Fiqih Pavita Andharluana

NPM : 21083010042

Kelas : Sistem Operasi – A

## Dokumentasi Tugas 8 (Multiprocessing)

Soal latihan :

Dengan menggunakan pemrosesan paralel buatlah program yang dapat menentukan sebuah bilangan itu ganjil atau genap!

Batasan:

Nilai yang dijadikan argumen pada fungsi sleep() adalah satu detik.

Masukkan jumlah'nya satu dan berupa bilangan bulat.

Masukkan adalah batas dari perulangan tersebut.

Setelah perulangan selesai program menampilkan

waktu eksekusi pemrosesan sekuensial dan paralel.

Output:

```
fiqih@fiqih-VirtualBox:~$ python3 Tugas_8.py
Masukkan angka:3
Sekuensial
1 Ganjil - punya ID proses 5359
2 Genap - punya ID proses 5359
3 Ganjil - punya ID proses 5359
Multiprocessing.process
1 Ganjil - punya ID proses 5360
2 Genap - punya ID proses 5361
3 Ganjil - punya ID proses 5362
Multiprocessing.pool
1 Ganjil - punya ID proses 5363
2 Genap - punya ID proses 5363
3 Ganjil - punya ID proses 5363
Waktu eksekusi Sekuensial : 3.0034728050231934 detik
Waktu eksekusi Multiprocessing.process : 2.0584700107574463 detik
Waktu eksekusi Multiprocessing.pool : 3.107203483581543 detik
```

Contoh input :

3

Contoh Output :

```
Sekuensial
1 Ganjil - ID proses ****
2 Genap - ID proses ****
3 Ganjil - ID proses ****

multiprocessing.Process
1 Ganjil - ID proses ****
2 Genap - ID proses ****
3 Ganjil - ID proses ****

multiprocessing.Pool
1 Ganjil - ID proses ****
2 Genap - ID proses ****
3 Ganjil - ID proses ****

Waktu eksekusi sekuensial : ** detik
Waktu eksekusi multiprocessing.Process : ** detik
Waktu eksekusi multiprocessing.Pool : ** detik
```

Berdasarkan perbandingan waktu eksekusi, proses yang paling cepat adalah menggunakan multiprocessing.process

Script:

```
fiqih@fiqih-VirtualBox: ~
File Edit View Search Terminal Help
GNU nano 6.2 Tugas_8.py
from os import getpid
from time import time, sleep
from multiprocessing import cpu_count, Pool, Process

# Inisialisasi fungsi yang akan digunakan
angka = int(input("Masukkan angka:"))

def cetak(i):
    if i % 2 == 1:
        print(i+1, "Genap", "- punya ID proses", getpid())
    else:
        print(i+1, "Ganjil", "- punya ID proses", getpid())
        sleep(i)
```

Pertama melakukan import beberapa built-in libraries. Selanjutnya membuat variabel angka yang berisi inputan berupa integer sebagai batas dari perulangan. Membuat function cetak yang didalamnya terdapat if else. Untuk i yang dimodulus 2 bersisa 1 adalah bilangan negatif, sedangkan untuk lainnya adalah genap.

```

fiqih@fiqih-VirtualBox: ~
File Edit View Search Terminal Help
GNU nano 6.2 Tugas_8.py
# Sekuensial
print("Sekuensial")
sekuensial_awal = time()

for i in range(angka):
    cetak(i)

sekuensial_akhir = time()

```

Pada sekuensial, pertama membuat variabel `sekuensial_awal` yang digunakan untuk mendapatkan waktu sebelum eksekusi. Selanjutnya proses berlangsung dengan rangenya adalah variabel angka yang pertama kali kita imputkan. Terakhir, membuat variabel `sekuensial_akhir` yang digunakan untuk mendapatkan waktu setelah eksekusi

```

fiqih@fiqih-VirtualBox: ~
File Edit View Search Terminal Help
GNU nano 6.2 Tugas_8.py
# Multiprocessing process
print("Multiprocessing.process")
kumpulan_proses = []
process_awal = time()

for i in range(angka):
    p = Process(target=cetak, args=(i, ))
    kumpulan_proses.append(p)
    p.start()

for i in kumpulan_proses:
    p.join()

process_akhir = time()

```

Pada Multiprocessing process, pertama membuat variabel `kumpulan_proses` yg digunakan untuk menampung proses-proses. Lalu, membuat variabel `process_awal` untuk mendapatkan waktu sebelum eksekusi. Lalu, proses berlangsung. Membuat perulangan `for i in kumpulan_proses:` untuk menggabungkan proses-proses agar tidak loncat ke proses sebelumnya. Terakhir, membuat variabel `process_akhir` untuk mendapatkan waktu setelah eksekusi

```

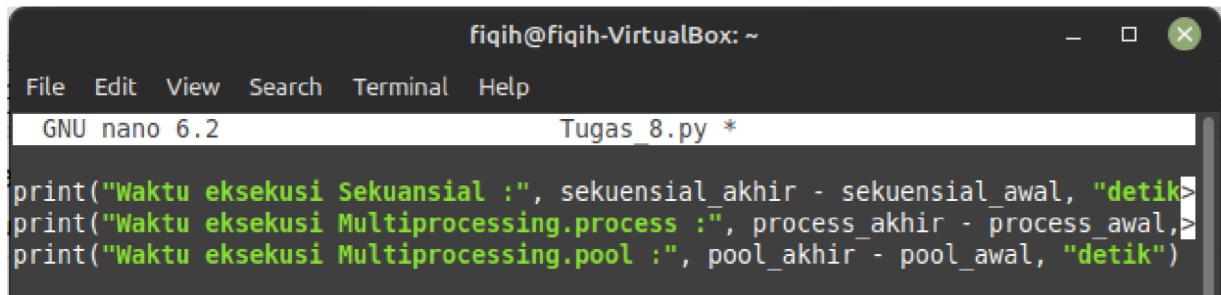
fiqih@fiqih-VirtualBox: ~
File Edit View Search Terminal Help
GNU nano 6.2 Tugas_8.py *
# Multiprocessing pool
print("Multiprocessing.pool")
pool_awal = time()

pool = Pool()
pool.map(cetak, range(0,angka))
pool.close()

pool_akhir = time()

```

Pada multiprocessing pool, pertama membuat variabel `pool_awal` yang digunakan untuk mendapatkan waktu sebelum eksekusi. Selanjutnya proses berlangsung dengan rangenya adalah `(0,angka)` variabel angka yang pertama kali kita imputkan. Terakhir, membuat variabel `pool_akhir` yang digunakan untuk mendapatkan waktu setelah eksekusi

A screenshot of a terminal window titled 'fiqih@fiqih-VirtualBox: ~'. The window shows the nano 6.2 text editor editing a file named 'Tugas 8.py'. The code in the editor consists of three print statements that calculate and display execution times in seconds ('detik') for sequential, multiprocessing process, and multiprocessing pool methods. The code is as follows:

```
print("Waktu eksekusi Sekuansial :", sekuensial_akhir - sekuensial_awal, "detik")
print("Waktu eksekusi Multiprocessing.process :", process_akhir - process_awal, "detik")
print("Waktu eksekusi Multiprocessing.pool :", pool_akhir - pool_awal, "detik")
```

Terakhir, membandingkan waktu eksekusi dengan mengurangi waktu eksekusi awal dengan waktu eksekusi. Berlaku untuk proses sekuensial, multiprocessing process, serta multiprocessing pool.