

Nama : Fajri Uswatul Hanifah

NPM : 21083010058

Kelas : Sistem Operasi A

## MULTIPROCESSING

### Soal Latihan

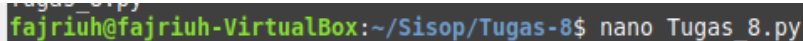
Dengan menggunakan pemrosesan paralel buatlah program yang dapat menentukan sebuah bilangan itu ganjil atau genap!

- 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.

Jawab :

1. Membuat file

Berikut demonya :

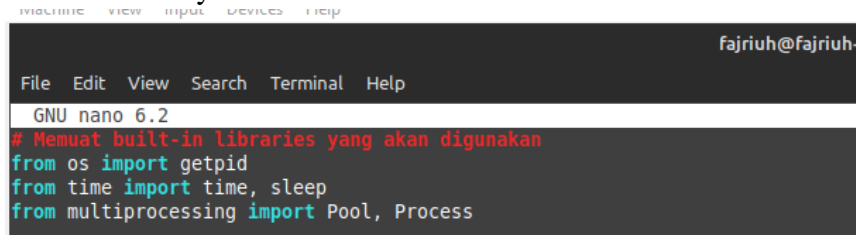


```
fajriuh@fajriuh-VirtualBox:~/Sisop/Tugas-8$ nano Tugas_8.py
```

Untuk membuat contoh dari operasi Multiprocessing kita memerlukan suatu file, maka dari itu kita harus membuat file terlebih dahulu hal ini dilakukan dengan cara menjalankan command “nano Tugas\_8.py” seperti gambar scrip diatas.

2. Menuliskan isi file .sh

Berikut demonya :



```
GNU nano 6.2
# Memuat built-in libraries yang akan digunakan
from os import getpid
from time import time, sleep
from multiprocessing import Pool, Process
```

- Mengimport modul yang diperlukan

Terdapat 5, yaitu :

1. getpid digunakan untuk mengambil ID proses
2. time digunakan untuk mengambil waktu( dektik)
3. cpu\_count digunakan untuk melihat jumlah CPU
4. Pool dalah sebuah class pada library multiprocessing yang digunakan untuk melakukan pemrosesan paralel dengan menggunakan proses sebanyak jumlah CPU pada computer
5. Process adalah sebuah class pada library multiprocessing yang digunakan untuk melakukan pemrosesan paralel dengan menggunakan proses secara beruntun pada computer

```
# Inisialisasi function yang akan digunakan
def cetak(i):
    bil = i % 2
    if bil == 0:
        print(i, "Genap - ID proses", getpid())
    else:
        print(i, "Ganjil - ID proses", getpid())
    sleep(1)
```

- Menginisialisasi fungsi yang akan digunakan  
Fungsi ini bernama fungsi cetak(i) yang digunakan untuk mencetak angka variable i beserta ID proses sejumlah parameter angka yang dimasukan. Jika hasil parameter ditambahkan 1 dan dimodulo 2 dan hasilnya sama dengan 0, maka masuk ke kondisi 1, jika tidak maka masuk ke kondisi 2. Selanjutnya memanggil fungsi sleep untuk memberikan jeda tiap detik setiap parameter yang diberikan.

```
# Input bilangan
x = int(input("Input bilangan: "))
```

- Membuat input bilangan  
Melakukan inisialisasi pengimputan variable n untuk menuliskan nilai batasan pada script

```
# Sekuensial
print("\nPemrosesan Sekuensial")
sekuensial_awal = time()

for i in range(1, x + 1):
    cetak(i)

sekuensial_akhir = time()
```

- Membuat scrip proses Sekuensial  
Pada Scrip diatas dilakukan proses sekuensial variable sekuensial\_awal digunakan untuk mendapatkan waktu durasi sebelum proses sekuensial berlangsung proses dijalankan menggunakan looping for sebanyak angka yang dimasukan, dan memanggil fungsi cetak yang telah diisi untuk mencetak setiap angka ganjil atau genap dengan id prosesnya masing-masing variable sekuensial\_akhir digunakan untuk mendapatkan waktu durasi

```

File Edit View Search Terminal Help
GNU nano 6.2

# Kelas Process
print("\nMultiprocessing dengan multiprocessing.Process")

kumpulan_proses = []
process_awal = time()

for i in range(1, x + 1):
    p = Process(target=cetak, args=(i,))
    kumpulan_proses.append(p)
    p.start()

for i in kumpulan_proses:
    p.join()

process_akhir = time()

```

- Membuat scrip Proses Multiprocessing Kelas Process

Mencetak tulisan dengan multiprocessing process, kemudian terdapat kumpulan proses untuk menyimpan list, terdapat process awal yang digunakan untuk mendapatkan waktu awal mulainya proses.

Kedua looping for digunakan untuk memproses menggunakan looping for sebanyak angka yang dimasukkan dan menggunakan fungsi cetak yang sudah kita isi di awal untuk mencetak setiap angka ganjil atau genap p.start() disini digunakan untuk mengeksekusi fungsi cetak di kelas process. Dan p.join() disini digunakan agar proses ditunggu hingga proses sebelumnya selesai. Sehingga akan menghasilkan id proses yang berbeda-beda tiap prosesnya. variable process\_akhir digunakan untuk mendapatkan waktu berakhirnya proses dijalankan

```

# Kelas Pool
print("\nMultiprocessing dengan multiprocessing.Pool")
pool_awal = time()

pool = Pool()
pool.map(cetak, range(1, x + 1))
pool.close()

pool_akhir = time()

```

- Membuat Scrip Process Multiprocessing tahap kelas pool

Pada process multiprocessing tahap kelas pool variable pool\_awal digunakan untuk mendapatkan waktu awal mulainya proses dijalankan variable pool digunakan untuk menjalankan fungsi Pool meninisiasi pool.map dengan fungsi map() digunakan untuk memetakan pemanggilan fungsi cetak ke dalam setiap CPU yang tersedia sebanyak 0-n kali yang mana 'n' adalah inputan batasan dari user variable pool\_akhir digunakan untuk mendapatkan waktu berakhirnya proses dijalankan.

```

# Perbandingan waktu eksekusi
print("\nWaktu eksekusi sekuensial          :", 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")

```

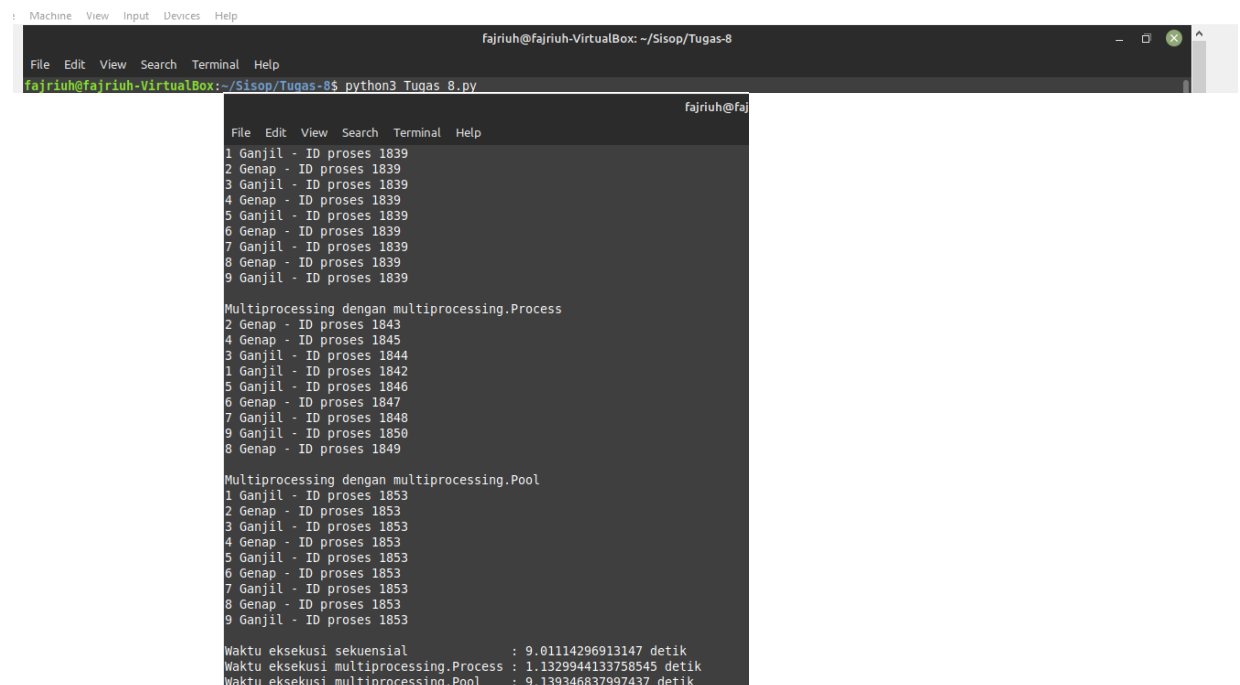
- Membuat scrip untuk membandingkan waktu eksekusi

Ini dilakukan dengan menggunakan perintah `print()`. Dengan ketentuan waktu eksekusi sekuensial dilakukan dengan memanggil variable sekuensial\_akhir dikurangi sekuensial\_awal, waktu eksekusi kelas process dilakukan dengan memanggil variable process\_akhir dikurangi process\_awal, dan waktu eksekusi kelas pool dilakukan dengan memanggil variable pool\_akhir dikurangi pool\_awal.

### 3. Menjalankan skrip

Setelah file yang kita buat disimpan, maka kita dapat menguji skrip kita dengan menjalankannya menggunakan command “`python3 Tugas_8.py`”

Berikut demonya :



```

Machine View Input Devices Help
fajriuh@fajriuh-VirtualBox: ~/Sisop/Tugas-8
File Edit View Search Terminal Help
fajriuh@fajriuh-VirtualBox:~/Sisop/Tugas-8$ python3 Tugas_8.py
fajriuh@fajriuh
File Edit View Search Terminal Help
1 Ganjil - ID proses 1839
2 Genap - ID proses 1839
3 Ganjil - ID proses 1839
4 Genap - ID proses 1839
5 Ganjil - ID proses 1839
6 Genap - ID proses 1839
7 Ganjil - ID proses 1839
8 Genap - ID proses 1839
9 Ganjil - ID proses 1839
Multiprocessing dengan multiprocessing.Process
2 Genap - ID proses 1843
4 Genap - ID proses 1845
3 Ganjil - ID proses 1844
1 Ganjil - ID proses 1842
5 Ganjil - ID proses 1846
6 Genap - ID proses 1847
7 Ganjil - ID proses 1848
9 Ganjil - ID proses 1850
8 Genap - ID proses 1849
Multiprocessing dengan multiprocessing.Pool
1 Ganjil - ID proses 1853
2 Genap - ID proses 1853
3 Ganjil - ID proses 1853
4 Genap - ID proses 1853
5 Ganjil - ID proses 1853
6 Genap - ID proses 1853
7 Ganjil - ID proses 1853
8 Genap - ID proses 1853
9 Ganjil - ID proses 1853
Waktu eksekusi sekuensial : 9.01114296913147 detik
Waktu eksekusi multiprocessing.Process : 1.1329944133758545 detik
Waktu eksekusi multiprocessing.Pool : 9.139346837997437 detik

```

Dari scrip diatas hasil yang dikeluarkan adalah. Terdapat tahap sekuensial memprintout id proses pada masing2 bilangan yang di inputkan, kemudian dilanjutkan dengan tahap multiprocessing kelas process memprintout id process, masing masing bilangan dengan id yang berbeda dikarenakan tiap pemanggilan fungsi cetak dilakukan oleh satu proses saja dan yang terakhir ada tahap multiprocessing kelas pool memprintout id process pada masing2 bilangan Memprintout seluruh waktu yang dilakukan pada tahap multiprocessing,

Maka dapat kita simpulkan bahwa Pada bagian akhir ditampilkan durasi waktu eksekusi untuk setiap jenis pemrosesan yang telah dijalankan. Dapat dilihat bahwa multiprocessing dengan kelas Process memiliki durasi waktu yang paling singkat dibandingkan dengan proses lainnya dan pemrosesan sekuensial menjadi proses yang memiliki durasi waktu terlama.