

Nama: Luqna Aziziyah  
NPM: 21083010020  
Kelas: Sistem Operasi (B)

Muat Built-in libraries yang akan kita gunakan pada tugas kali ini yaitu seperti berikut:

```
from os import getpid
from time import time, sleep
from multiprocessing import cpu_count, Pool, Process
```

Dimana memiliki arti:

**getpid** digunakan untuk mengambil ID proses

**time** digunakan untuk mengambil waktu(detik)

**sleep** digunakan untuk memberi jeda waktu(detik)

**cpu\_count** digunakan untuk melihat jumlah CPU

**Pool** adalah sebuah class pada library multiprocessing yang digunakan untuk melakukan pemrosesan paralel dengan menggunakan proses sebanyak jumlah CPU pada komputer

**Process** adalah sebuah class pada library multiprocessing yang digunakan untuk melakukan pemrosesan paralel dengan menggunakan proses secara beruntun pada computer

## TUGAS 8

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:

Contoh input :

3

Contoh Output :

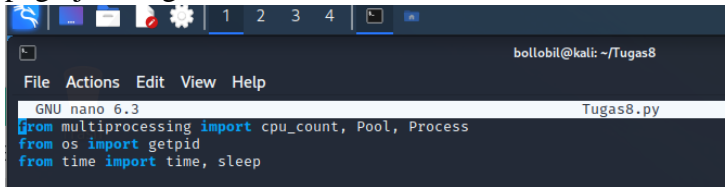
```
Sekuensial
1 Ganjil - ID proses ****
2 Genap - ID proses ****
3 Ganjil - ID proses ****

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
```

1. Pertama kita membuat sebuah file nano, yang disini saya namai dengan Tugas8.py
2. Selanjutnya saya me-import sebuah library bawaan dari python untuk memudahkan pengerjaan tugas kali ini.

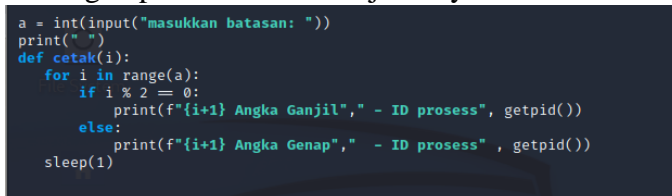


```

File Actions Edit View Help
GNU nano 6.3 Tugas8.py
from multiprocessing import cpu_count, Pool, Process
from os import getpid
from time import time, sleep

```

3. Dapat dilihat seperti gambar dibawah ini bahwa saya melakukan sebuah deklarasi yang saya beri nama cetak, sebelum itu saya membuat inputan user yang akan memiliki hubungan pada deklarasi lanjutannya.



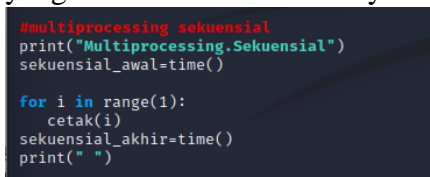
```

a = int(input("masukkan batasan: "))
print(" ")
def cetak(i):
    for i in range(a):
        if i % 2 == 0:
            print(f"{i+1} Angka Ganjil", " - ID proses", getpid())
        else:
            print(f"{i+1} Angka Genap", " - ID proses", getpid())
        sleep(1)

```

Dalam def cetak, saya membuat sebuah looping for, yang dibatasi oleh inputan yang akan diberikan oleh user, lalu saya juga menggunakan if else, yang akan menunjukkan perulangan dari inputan user itu merupakan bilangan ganjil atau genap.

4. Selanjutnya masuk ke multiprocessing sekuensial yang dimana kita menggunakan looping for dengan range 1 yaitu setiap perulangan hanya terjadi 1 kali, tidak lupa memberikan sekuensial\_awal dan sekuensial\_akhir guna menghitung waktu eksekusi yang akan kita buat setelahnya.



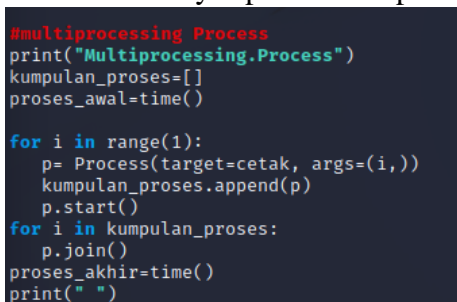
```

#multiprocessing sekuensial
print("Multiprocessing.Sekuensial")
sekuensial_awal=time()

for i in range(1):
    cetak(i)
sekuensial_akhir=time()
print(" ")

```

5. Selanjutnya pada bagian multiprocessing process, yang tidak jauh berbeda dari sekuensial, disini juga kita menggunakan looping for untuk perulangannya. Bedanya disini kita menyimpan sebuah prosesnya agar tidak meramba ke proses selanjutnya.



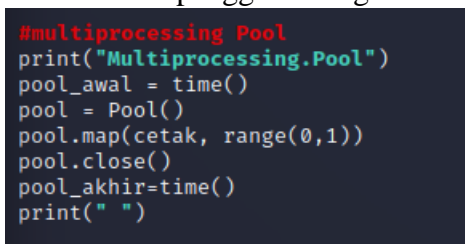
```

#multiprocessing Process
print("Multiprocessing.Process")
kumpulan_proses=[]
proses_awal=time()

for i in range(1):
    p= Process(target=cetak, args=(i,))
    kumpulan_proses.append(p)
    p.start()
for i in kumpulan_proses:
    p.join()
proses_akhir=time()
print(" ")

```

6. Selanjutnya pada multiprocessing pool, disini kita menggunakan fungsi .map() guna memetakan panggilan fungsi cetak kedalam sebanyak 1 kali



```

#multiprocessing Pool
print("Multiprocessing.Pool")
pool_awal = time()
pool = Pool()
pool.map(cetak, range(0,1))
pool.close()
pool_akhir=time()
print(" ")

```

7. Terakhir kita akan membuat sebuah lama waktu pengekseskuan itu berlangsung, caranya yaitu dengan waktu akhir – waktu awal, sseperti yang telah kita deklarasikan pada setiap awal dan akhir dari sebuah program multiprocessing

```
#waktu eksekusi
print("Waktu eksekusi Sekuensial          : ",sekuensial_akhir - sekuensial_awal, "detik")
print("Waktu eksekusi Multiprocessing.Process: ",proses_akhir - proses_awal, "detik")
print("Waktu eksekusi Multiprocessing.Pool   : ",pool_akhir - pool_awal, "detik")
```

8. Untuk outputnya dapat dilihat seperti berikut

```
(bollobil@kali)-[~/Tugas8]
$ python3 Tugas8.py
masukkan batasan: 3

Multiprocessing.Sekuensial
1 Angka Ganjil - ID prosess 47137
2 Angka Genap  - ID prosess 47137
3 Angka Ganjil - ID prosess 47137

Multiprocessing.Process
1 Angka Ganjil - ID prosess 47170
2 Angka Genap  - ID prosess 47170
3 Angka Ganjil - ID prosess 47170

Multiprocessing.Pool
1 Angka Ganjil - ID prosess 47175
2 Angka Genap  - ID prosess 47175
3 Angka Ganjil - ID prosess 47175

Waktu eksekusi Sekuensial          : 1.001103162765503 detik
Waktu eksekusi Multiprocessing.Process: 1.007685899734497 detik
Waktu eksekusi Multiprocessing.Pool   : 1.0311930179595947 detik
```

```
(bollobil@kali)-[~/Tugas8]
$ python3 Tugas8.py
masukkan batasan: 5

Multiprocessing.Sekuensial
1 Angka Ganjil - ID prosess 49401
2 Angka Genap  - ID prosess 49401
3 Angka Ganjil - ID prosess 49401
4 Angka Genap  - ID prosess 49401
5 Angka Ganjil - ID prosess 49401

Multiprocessing.Process
1 Angka Ganjil - ID prosess 49414
2 Angka Genap  - ID prosess 49414
3 Angka Ganjil - ID prosess 49414
4 Angka Genap  - ID prosess 49414
5 Angka Ganjil - ID prosess 49414

Multiprocessing.Pool
1 Angka Ganjil - ID prosess 49419
2 Angka Genap  - ID prosess 49419
3 Angka Ganjil - ID prosess 49419
4 Angka Genap  - ID prosess 49419
5 Angka Ganjil - ID prosess 49419

Waktu eksekusi Sekuensial          : 1.0005238056182861 detik
Waktu eksekusi Multiprocessing.Process: 1.0112278461456299 detik
Waktu eksekusi Multiprocessing.Pool   : 1.089967966079712 detik
```