

Nama : Zulfaz Refie Ababil

NPM : 21083010122

Sistem Operasi B

## 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.

### Script:

```
zulfaz@zulfaz-VirtualBox: ~/Tugas Sisop8/Tugas8
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

a=int(input("Batas: "))
print("")

def oddeven(i):
    if (i%2) == 0:
        return "genap"
    else:
        return "ganjil"

def cetak(i):
    print(i+1, oddeven(i+1), "- punya ID proses", getpid())
    sleep(1)

# Proses Sekuensial:
print("Sekuensial")
sekuensial_awal = time()
for i in range(a):
    cetak(i)
sekuensial_akhir=time()
print("")

# Multiprocessing dengan kelas Process:
print("Multiprocessing.process")
kumpulan_proses=[]
process_awal=time()
for i in range(a):
    p=Process(target=cetak, args=(i, ))
    kumpulan_proses.append(p)
    p.start()
for i in kumpulan_proses:
    p.join()
process_akhir=time()
print("")

#Multiprocessing Dengan Kelas Pool:
print("Multiprocessing.pool")
pool_awal=time()
pool = Pool()
pool.map(cetak, range(0,a))
pool.close()
pool_akhir=time()
print("")

#Bandingkan Waktu Eksekusi
print("Waktu eksekusi Sekuensial:", sekuensial_akhir - sekuensial_awal, "detik")
print("Waktu eksekusi kelas Process:", process_akhir - process_awal, "detik")
print("Waktu eksekusi kelas Pool:", pool_akhir - pool_awal, "detik")

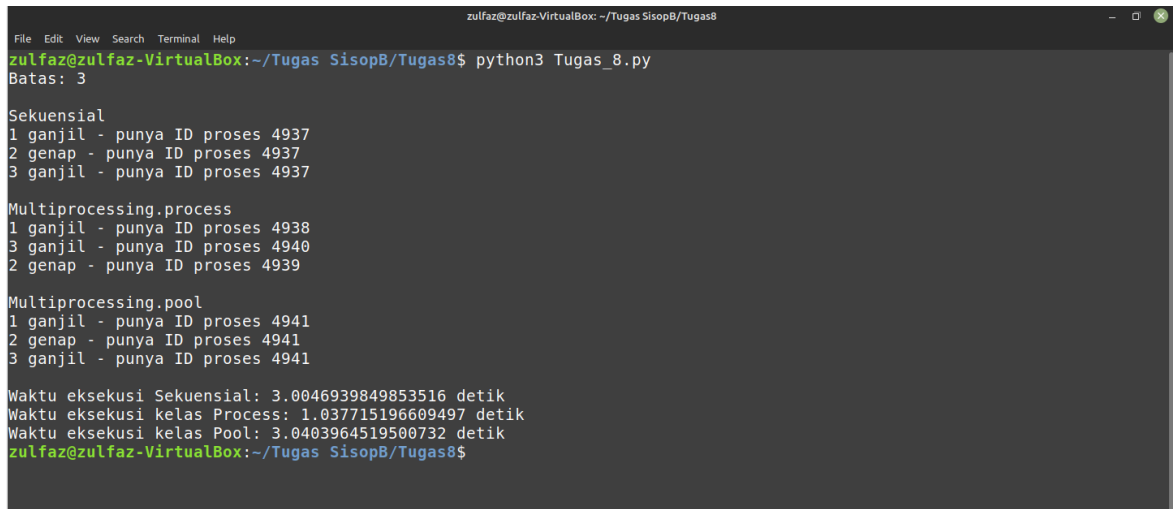
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^_ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo
```

Ketik script sesuai gambar diatas.

- Muat *built-in libraries* yang akan digunakan:
  1. *getpid* digunakan untuk mengambil ID proses
  2. *time* digunakan untuk mengambil waktu(detik)
  3. *sleep* digunakan untuk memberi jeda waktu(detik)
  4. *cpu\_count* digunakan untuk melihat jumlah CPU
  5. *Pool* adalah sebuah class pada library multiprocessing yang digunakan untuk melakukan pemrosesan paralel dengan menggunakan proses sebanyak jumlah CPU pada komputer
  6. *Process* adalah sebuah class pada library multiprocessing yang digunakan untuk melakukan pemrosesan paralel dengan menggunakan proses secara beruntun pada komputer
- Memberikan variabel *a* sebagai input untuk user.
- Mendefinisikan fungsi *oddeven* untuk mengecek bilangan ganjil atau genap menggunakan *if-else*. Jika  $i \% 2 == 0$ , maka bilangan tersebut adalah genap dan jika tidak, maka bilangan tersebut adalah ganjil.
- Mendefinisikan fungsi cetak yang digunakan untuk mencetak angka dari variabel *i* beserta ID proses sejumlah parameter yang diberikan. Kemudian fungsi *oddeven*. Sedangkan fungsi *sleep* untuk memberi jeda waktu (detik) sebanyak parameter yang diberikan.
- Proses sekuensial
  1. *sekuensial\_awal* digunakan untuk mendapatkan waktu durasi sebelum proses berlangsung.
  2. Proses sekuensial dilakukan dengan melakukan perulangan *i* dengan range berkisar pada *a*. Untuk *i* yang berada dalam range *a*, maka akan dilakukan fungsi cetak pada *i*.
  3. *sekuensial\_akhir* adalah variabel untuk mendapatkan waktu durasi setelah proses berlangsung.
- Multiprocessing.Process
  1. Variabel *kumpulan\_proses* untuk menampung kumpulan proses.
  2. *process\_awal* digunakan untuk mendapatkan waktu durasi sebelum proses berlangsung
  3. Menggunakan for loop untuk semua elemen dalam range *a*, dilakukan fungsi *Process* yang membutuhkan argumen *target* dan *args* yang akan dieksekusi oleh proses dan argument (*i*) yang akan diteruskan ke fungsi *target* yaitu fungsi cetak. Hasil dari fungsi *Process* ditambahkan ke dalam variabel yang *kumpulan\_proses*. Kemudian *Process* dijalankan Lalu untuk semua elemen yang berada di variabel *kumpulan\_proses* akan ditampung dan digabung menjadi satu menggunakan (*p.join()*) agar tidak merambah ke proses selanjutnya.
  4. Variabel *proses\_akhir* adalah variabel untuk mendapatkan waktu durasi setelah proses berlangsung
- Dalam multiprocessing.Pool, terlebih dahulu menginisialisasikan variabel *pool = fungsi Pool*. Kemudian menggunakan fungsi *map* dengan fungsi cetak sebagai

target, dan `a` sebagai argument yang diteruskan. Menggunakan `pool.close()` untuk menolak task baru atau untuk berhenti. Serta didefinisikan `pool_awal` dan `pool_akhir` sebagai variabel yang mendapatkan waktu sebelum dan setelah proses.

- Proses terakhir yaitu bandingkan setiap jenis eksekusi dengan waktu akhir – waktu awal untuk melihat berapa lama pemrosesan berlangsung. Kemudian print waktu eksekusi sekuensial, kelas process, dan kelas pool untuk melihat hasil waktu berapa detiknya.



```
zulfaz@zulfaz-VirtualBox: ~/Tugas Sisop8/Tugas8
File Edit View Search Terminal Help
zulfaz@zulfaz-VirtualBox:~/Tugas Sisop8/Tugas8$ python3 Tugas_8.py
Batas: 3

Sekuenisial
1 ganjil - punya ID proses 4937
2 genap - punya ID proses 4937
3 ganjil - punya ID proses 4937

Multiprocessing.process
1 ganjil - punya ID proses 4938
3 ganjil - punya ID proses 4940
2 genap - punya ID proses 4939

Multiprocessing.pool
1 ganjil - punya ID proses 4941
2 genap - punya ID proses 4941
3 ganjil - punya ID proses 4941

Waktu eksekusi Sekuenisial: 3.0046939849853516 detik
Waktu eksekusi kelas Process: 1.037715196609497 detik
Waktu eksekusi kelas Pool: 3.0403964519500732 detik
zulfaz@zulfaz-VirtualBox:~/Tugas Sisop8/Tugas8$
```