

Nama : Mutiara Irmadhani

Npm : 21083010079

Kelas : Sistem Operasi B

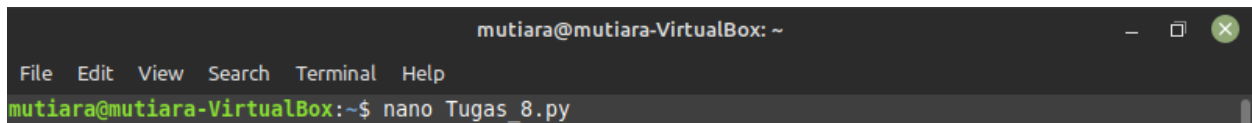
TUGAS 8

Multiprocessing

Soal latihan :

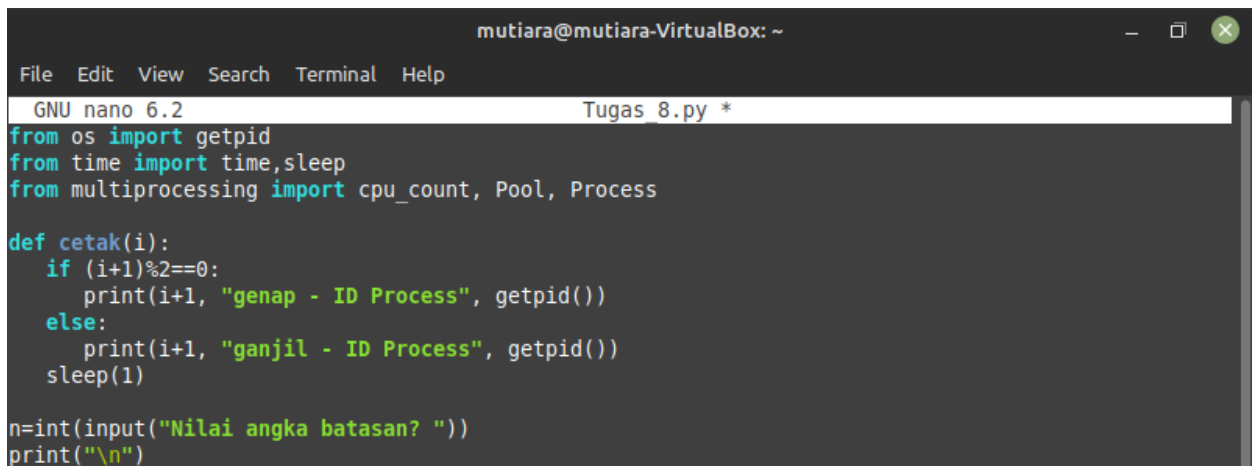
Dengan menggunakan pemrosesan paralel buatlah program yang dapat menentukan sebuah bilangan itu ganjil atau genap!

Buatlah file dengan menggunakan 'nano Tugas_8.py'



```
mutiara@mutiara-VirtualBox: ~  
File Edit View Search Terminal Help  
mutiara@mutiara-VirtualBox:~$ nano Tugas_8.py
```

Ketik scriptnya seperti pada gambar



```
mutiara@mutiara-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  
  
def cetak(i):  
    if (i+1)%2==0:  
        print(i+1, "genap - ID Process", getpid())  
    else:  
        print(i+1, "ganjil - ID Process", getpid())  
    sleep(1)  
  
n=int(input("Nilai angka batasan? "))  
print("\n")
```

Pertama Import modul yang diperlukan.

- getpid digunakan untuk mendapatkan ID proses
- time digunakan untuk mengambil waktu(dektik) pada proses dijalankan atau diakhiri
- sleep digunakan untuk memberi jeda waktu(detik)
- cpu_count digunakan untuk menghitung jumlah cpu yang tersedia
- 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 komputer

Fungsi bernama ‘cetak’ digunakan untuk mencetak angka dari variabel i beserta ID proses sejumlah parameter apakah angka yang masuk ganjil atau genap. Disini ketika angka yang dimasukkan menghasilkan 0 ketika di modulo 2, maka angka tersebut genap dan jika menghasilkan angka lainnya, maka angka tersebut berarti ganjil. Kita panggil fungsi sleep untuk memberi jeda waktu(detik) sebanyak parameter yang diberikan. Kemudian fungsi ‘n’ digunakan untuk User diminta menginputkan sebuah angka bulat yang digunakan sebagai Nilai Batasan. Lalu print (“\n”) untuk memberi spasi agar mudah dipahami.

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

Proses pertama gunakan sekuensial processing.

- sekuensial_awal adalah variabel untuk mendapatkan waktu durasi sebelum proses sekuensial processing berlangsung.
- sekuensial_akhir adalah variabel untuk mendapatkan waktu durasi setelah proses sekuensial processing berlangsung.
- Lakukan Looping sebanyak angka yang dimasukkan oleh user, dan gunakan fungsi ‘cetak’ yang sudah kita isi di awal untuk mencetak setiap angka ganjil atau genap dengan id prosesnya masing – masing.
- Lalu print (“\n”) untuk memberi spasi agar mudah dipahami.

```
#Multiprocessing Dengan Kelas Process:
process_awal=time()
print("Multiprocessing.process")
for i in range(n):
    p=Process(target=cetak, args=(i, ))
    p.start()
    p.join()
process_akhir=time()
print("\n")
```

Proses kedua gunakan multiprocessing dengan kelas process

- process_awal adalah variabel untuk mendapatkan waktu awal mulainya proses dijalankan
- process_akhir adalah variabel untuk mendapatkan waktu berakhirnya proses dijalankan
- Dapat diperhatikan dengan seksama bahwa ID proses tiap memanggil fungsi cetak adalah berbeda-beda. Ini menandakan bahwa tiap pemanggilan fungsi cetak ditangani oleh satu proses saja. Kemudian untuk pemanggilan selanjut'nya ditangani oleh proses yang lain.
- Lakukan Looping sebanyak angka yang dimasukkan oleh user, dan gunakan fungsi cetak yang sudah kita isi di awal untuk mencetak setiap angka ganjil atau genap dengan id proses masing – masing
- p.start() digunakan untuk mengeksekusi fungsi cetak di kelas process

- p.join() digunakan agar proses ditunggu hingga proses sebelumnya selesai. Sehingga akan menghasilkan id proses yang berbeda – beda tiap prosesnya.
- Lalu print ("\n") untuk memberi spasi agar mudah dipahami.

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

Proses Ketiga gunakan multiprocessing dengan kelas pool

- pool_awal adalah variabel untuk mendapatkan waktu awal mulainya proses dijalankan
- pool_akhir adalah variabel untuk mendapatkan waktu berakhirnya proses proses dijalankan
- 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.
- Lalu print ("\n") untuk memberi spasi agar mudah dipahami.

```
#Bandingkan Waktu Eksekusi
print("Hasil Perbandingan waktu")
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")
```

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. Lalu print ("\n") untuk memberi spasi agar mudah dipahami.

Outputnya:

```
mutiara@mutiara-VirtualBox: ~  
File Edit View Search Terminal Help  
mutiara@mutiara-VirtualBox:~$ python3 Tugas_8.py  
Nilai angka batasan? 4  
  
Sekuensial  
1 ganjil - ID Process 2351  
2 genap - ID Process 2351  
3 ganjil - ID Process 2351  
4 genap - ID Process 2351  
  
Multiprocessing.process  
1 ganjil - ID Process 2352  
2 genap - ID Process 2353  
3 ganjil - ID Process 2354  
4 genap - ID Process 2355  
  
Multiprocessing.pool  
1 ganjil - ID Process 2356  
2 genap - ID Process 2356  
3 ganjil - ID Process 2356  
4 genap - ID Process 2356  
  
Hasil Perbandingan waktu  
Waktu eksekusi Sekuensial: 4.004572868347168 detik  
Waktu eksekusi kelas Process: 4.0285539627075195 detik  
Waktu eksekusi kelas Pool: 4.034626245498657 detik  
mutiara@mutiara-VirtualBox:~$
```

Lakukan pemanggilan output dengan python3 namafile. Kemudian Hasil outputnya user memasukkan angka 4 sebagai Batasan.

- Pada sekuensial, terlihat bahwa setiap ID processnya itu sama. Hal ini karena sekuensial akan eksekusi pada pemroses yang sama.
- Pada multiprocessing dengan kelas process, terlihat bahwa setiap ID processnya itu berbeda dan beruntun. Hal ini menunjukkan bahwa tiap pemanggilan fungsi cetak dilakukan oleh satu proses saja.
- Pada multiprocessing dengan kelas pool, terlihat ada 1 ID process yang sama namun berulang. Karena di laptop saya ada 1 CPU maka dipetakan menjadi 1 pemrosesan di tiap CPU yang sama. Tidak apa tidak urut, karena begitupula parallel processing.