

Nama : Adhisa Shilfadianis Iffadah

NPM : 21083010016

Kelas : Sistem Operasi B

## **Tugas 8**

### **Multiprocessing**

Pemrograman paralel adalah sebuah teknik eksekusi perintah yang mana dilakukan secara bersamaan pada CPU. Seluruh bahasa pemrograman yang populer dapat melakukan pemrograman paralel dengan modul bawaan atau memang pengaturan defaultnya seperti itu. Manfaat Multiprocessing adalah menggunakan CPU untuk komputasi, Tidak berbagi sumber daya memori, memerlukan sumber daya memori dan waktu yang tidak sedikit, dan tidak memerlukan sinkronisasi memori.

#### **Latihan soal :**

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 jumlahnya satu dan berupa bilangan bulat
- Masukkan adalah batasan dari perulangan tersebut
- Setelah perulangan selesai program menampilkan waktu eksekusi pemrosesan sekuensial dan paralel

#### **Penyelesaian :**

**Code**

```
adhisa@adhisa-VirtualBox: ~
File Edit View Search Terminal Help
GNU nano 6.2 tugas8.py
from os import getpid
from time import time, sleep
from multiprocessing import cpu_count, Pool, Process

x = int(input())

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

print('\nSekuensial')
sekuensial_awal = time()

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

sekuensial_akhir = time()

print('\nmultiprocessing.Process')
kumpulan_proses = []
process_awal = time()

[ Read 47 lines ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify
              ^C Location
              ^_ Go To Line

19:37
```

## Output

```
adhisa@adhisa-VirtualBox: ~
File Edit View Search Terminal Help
adhisa@adhisa-VirtualBox:~$ nano tugas8.py
adhisa@adhisa-VirtualBox:~$ python3 tugas8.py
3

Sekuensial
1 Ganjil - punya ID proses 2350
2 Genap - punya ID proses 2350
3 Ganjil - punya ID proses 2350

multiprocessing.Process
1 Ganjil - punya ID proses 2351
2 Genap - punya ID proses 2352
3 Ganjil - punya ID proses 2353

multiprocessing.pool
1 Ganjil - punya ID proses 2354
2 Genap - punya ID proses 2355
3 Ganjil - punya ID proses 2355

Sekuensial : 3.005563735961914 detik
multiprocessing.Process : 1.0339486598968506 detik
multiprocessing.pool : 2.2878825664520264 detik
adhisa@adhisa-VirtualBox:~$
```

**Source Code :**

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

x = int(input())

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

print('\nSekuensial')
# UNTUK MENDAPATKAN WAKTU SEBELUM EKSEKUSI
sekuensial_awal = time()

# PROSES BERLANGSUNG
for i in range(x):
    cetak(i)

# UNTUK MENDAPATKAN WAKTU SETELAH EKSEKUSI
sekuensial_akhir = time()

print('\nmultiprocessing.Process')
# UNTUK MENAMPUNG PROSES-PROSES
kumpulan_proses = []

# UNTUK MENDAPATKAN WAKTU SEBELUM EKSEKUSI
process_awal = time()

# PROSES BERLANGSUNG
```

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

# UNTUK MENGGABUNGKAN PROSES-
# PROSES AGAR TIDAK LONCAT KE PROSES SEBELUM'NYA
for i in kumpulan_proses:
    p.join()

# UNTUK MENDAPATKAN WAKTU SETELAH EKSEKUSI
process_akhir = time()

print('\nmultiprocessing.pool')
# UNTUK MENDAPATKAN WAKTU SEBELUM EKSEKUSI
pool_awal = time()

# PROSES BERLANGSUNG
pool = Pool()
pool.map(cetak, range(0, x))
pool.close()

# UNTUK MENDAPATKAN WAKTU SEBELUM EKSEKUSI
pool_akhir = time()

# Banding Waktu Eksekusi
print("\nSekuensial :", sekuensial_akhir -
    sekuensial_awal, "detik")
print("multiprocessing.Process :", process_akhir -
    process_awal, "detik")
print("multiprocessing.pool :", pool_akhir -
    pool_awal, "detik")
```

### Penjelasan :

Pertama buat terlebih dahulu file dengan menggunakan nano, pada praktikum kali ini saya memberikan nama tugas8.py. Pada script bash, pertama kita melakukan import library terlebih dahulu.

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

**Get id** digunakan untuk mengambil ID Proses

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

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

**Cpu\_count** digunakan untuk melihat 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 secara beruntun pada komputer

Selanjutnya, yaitu **membuat fungsi** yang akan digunakan yaitu dengan menggunakan script berikut

```
x = int(input())

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

Pertama yaitu menginput nilai looping yang ingin dilakukan. Pada praktikum ini saya mendeklarasikannya sebagai variabel x. Kemudian, def cetak untuk membuat fungsi yang ingin dijalankan. Dimana ia akan mencetak i, yang jika i tersebut merupakan modulus 2 maka akan dikatakan bilangan genap dan jika bukan maka akan dikatakan sebagai bilangan ganjil. Fungsi sleep diberikan untuk memberikan jeda waktu sebanyak parameter yang diberikan.

Pemrosesan **Sekuensial** digunakan dengan menggunakan script berikut

```

print('\nSekuensial')
# UNTUK MENDAPATKAN WAKTU SEBELUM EKSEKUSI
sekuensial_awal = time()

# PROSES BERLANGSUNG
for i in range(x):
    cetak(i)

# UNTUK MENDAPATKAN WAKTU SETELAH EKSEKUSI
sekuensial_akhir = time()

```

Pertama, yaitu membuat awal waktunya sebelum mengeksekusinya, kemudian pada proses berlangsung merupakan i yang akan terus melakukan looping selama bilangan i tersebut masih berada dalam range x/nilai yang diinputkan kemudian jika masih terdapat dalam range x maka akan dimenghasilkan output pada fungsi cetak(i) yang telah kita buat.

Multipreprocessing dalam **multiprocessing.Process** dapat dilakukan dengan menjalankan script berikut ini.

```

print('\nmultiprocessing.Process')
# UNTUK MENAMPUNG PROSES-PROSES
kumpulan_proses = []

# UNTUK MENDAPATKAN WAKTU SEBELUM EKSEKUSI
process_awal = time()

# PROSES BERLANGSUNG
for i in range(x):
    p = Process(target=cetak, args=(i, ))
    kumpulan_proses.append(p)
    p.start()

# UNTUK MENGGABUNGKAN PROSES
PROSES AGAR TIDAK LONCAT KE PROSES SEBELUM'NYA
for i in kumpulan_proses:
    p.join()

```

```
# UNTUK MENDAPATKAN WAKTU SETELAH EKSEKUSI
process_akhir = time()
```

Pertama yaitu menampung proses-proses kemudian mendeklarasikan start waktunya. Kemudian pada prosesnya menggunakan perulangan for dengan range yang disesuaikan dengan inputan yang dilakukan user selanjutnya menjalankan fungsi Process yang telah diambil dari sebelumnya dengan target yaitu fungsi cetak dan argumennya adalah i, tak lupa juga menambahkannya dalam list kumpulan\_proses. Lalu melakukan perulangan for untuk menggabungkan prosesnya agar tidak loncat/beralih ke proses sebelumnya. Dan penutupnya yaitu mendeklarasikan waktu berakhirnya eksekusi proses tersebut.

**Multiprocessing Pool** ini akan digunakan dengan menjalankan program sebagai berikut :

```
print('\nmultiprocessing.pool')
# UNTUK MENDAPATKAN WAKTU SEBELUM EKSEKUSI
pool_awal = time()

# PROSES BERLANGSUNG
pool = Pool()
pool.map(cetak, range(0, x))
pool.close()

# UNTUK MENDAPATKAN WAKTU SEBELUM EKSEKUSI
pool_akhir = time()
```

Pertama yaitu mendapatkan waktu sebelum eksekusinya, kemudian masuk pada bagian prosesnya yaitu dengan menyatakan pool dengan fungsi/class library pool selanjutnya yaitu menggunakan fungsi pool.map untuk memanggil panggilan fungsi cetak dalam range 0 hingga inputan bilangan user. Dan terakhir mendeklarasikan end time dalam menjalankan proses tersebut.

**Membandingkan Waktu Eksekusi** dapat dilakukan dengan cara menggunakan script bash berikut

```
# Banding Waktu Eksekusi
print("\nSekuensial :", sekuensial_akhir -
```

```
sekuensial_awal, "detik")
print("multiprocessing.Process :", process_akhir -
      process_awal, "detik")
print("multiprocessing.pool :", pool_akhir -
      pool_awal, "detik")
```

Pada bagian terakhir ini dilakukan perbandingan terhadap lamanya waktu eksekusi pada masing-masing proses dalam hitungan detik. Dilakukan dengan cara mengurangi antara waktu awal/start time dan waktu berakhir/end time-nya suatu proses.

Contoh: proses Sekuensial

Dilakukan dengan cara mengurangi sekuensial\_akhir dengan sekuensial\_awalnya sehingga didapatkan hasil seperti pada output gambar diatas. Dan begitupun seterusnya.