MATA KULIAH SISTEM OPERASI

MULTIPROCESSING

Nama : Lisya Septyo Ningrum

NPM : 21083010003

Kelas : Sistem Operasi B

MATERI 8

Buat File .py di terminal linux dengan cara ketik nano nama_file.py.

```
mint@mint:~/Tugas8$ nano Materi8.py
```

Built-in Libraries yang digunakan :

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

Beberapa libraries yang digunakan bertujuan untuk :

- **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 library multiprocessing yang digunakan untuk melakukan pemrosesan parallel dengan menggunakan proses sebanyak jumlah CPU pada computer.
- **Process** adalah sebuah class library multiprocessing yang digunakan untuk melakukan pemrosesan parallel dengan menggunakan proses secara beruntun pada computer.

Inisialisasi fungsi yang akan digunakan :

```
#Inisialisasi fungsi yang digunakan
def cetak(i):
    print("Cetak angka", i+1, "punya ID Proses", getpid())
    sleep(1)
```

Fungsi diatas digunakan untuk mencetak angka dari variabel i beserta IP proses sejumlah parameter yang diberikan. Kita panggil fungsi sleep untuk memberi jeda waktu (detik) sebanyak parameter yang diberikan.

Jenis multiprocessing dan implementasi script-nya.

1. Pemrosesan Sekuensial

```
print("------------------------------------------------------------")

print("1. Pemrosesan sekuensial")
#Untuk mendapatkan waktu sebelum eksekusi
sekuensial_awal = time()

#Proses Berlangsung
for i in range(10):
    cetak(i)

#Untuk mendapatkan waktu setelah eksekusi
sekuensial_akhir = time()
```

Output pemrosesan sekuensial :

```
mint@mint:~/Tugas8$ python3 Materi8.py
------------------------------------------------------------
1. Pemrosesan sekuensial
Cetak angka 1 punya ID Proses 2965
Cetak angka 2 punya ID Proses 2965
Cetak angka 3 punya ID Proses 2965
Cetak angka 4 punya ID Proses 2965
Cetak angka 5 punya ID Proses 2965
Cetak angka 6 punya ID Proses 2965
Cetak angka 7 punya ID Proses 2965
Cetak angka 8 punya ID Proses 2965
Cetak angka 9 punya ID Proses 2965
Cetak angka 10 punya ID Proses 2965
------------------------------------------------------------
```

2. Multiprocessing dengan kelas Process

```
print("------------------------------------------------------------")

print("2. Multiprocessing dengan kelas Process")
#Untuk menampung proses-proses
kumpulan_proses = []

#Untuk mendapatkan waktu sebelum eksekusi
process_awal = time()

#Proses berlangsung
for i in range(10):
    p = Process(target=cetak, args=(i,))
    kumpulan_proses.append(p)
    p.start()

#Untuk menggabungkan proses-proses agar tidak loncat ke proses sebelumnya
for i in kumpulan_proses:
    p.join()

#Untuk mendapatkan waku setelah eksekusi
process_akhir = time()
```

Output Multiprocessing dengan kelas Process :

```
------------------------------------------------------------
2. Multiprocessing dengan kelas Process
Cetak angka 1 punya ID Proses 2966
Cetak angka 2 punya ID Proses 2967
Cetak angka 4 punya ID Proses 2969
Cetak angka 5 punya ID Proses 2970
Cetak angka 6 punya ID Proses 2971
Cetak angka 3 punya ID Proses 2968
Cetak angka 7 punya ID Proses 2972
Cetak angka 8 punya ID Proses 2973
Cetak angka 9 punya ID Proses 2974
Cetak angka 10 punya ID Proses 2975
```

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 selanjutnya ditangani oleh proses yang lain.
Kumpulan proses harus ditampung dan digabung menjadi satu(p.join())agar tidak merambah ke proses selanjutnya. Silahkan eksekusi file berikut pada terminal anda, maka anda akan paham apa yang saya maksudkan.

3. Multiprocessing dengan kelas Pool

```
print("------------------------------------------------------------")

print("3. Multiprocessing dengan kelas Pool")
#Untuk mendapatkan waktu sebelum eksekusi
pool_awal = time()

#Proses berlangsung
pool = Pool()
pool.map(cetak, range(0,10))
pool.close()

#Untuk mendapatkan waktu sebelum eksekusi
pool_akhir = time()
```

Output Multiprocessing dengan kelas Pool :

```
------------------------------------------------------------
3. Multiprocessing dengan kelas Pool
Cetak angka 1 punya ID Proses 2976
Cetak angka 2 punya ID Proses 2976
Cetak angka 3 punya ID Proses 2976
Cetak angka 4 punya ID Proses 2976
Cetak angka 5 punya ID Proses 2976
Cetak angka 6 punya ID Proses 2976
Cetak angka 7 punya ID Proses 2976
Cetak angka 8 punya ID Proses 2976
Cetak angka 9 punya ID Proses 2976
Cetak angka 10 punya ID Proses 2976
```

Jumlah ID proses terbatas pada empat saja karena jumlah CPU pada komputer saya hanyalah 4. Jangan risaukan urutan angka yang dicetak jika tidak berurutan, kan emang ini pemrosesan paralel. Fungsi map() itu memetakan pemanggilan fungsi cetak ke dalam 4 CPU sebanyak 10 kali.

Perbandingan waktu eksekusi

```
print("------------------------------------------------------------")

print("Perbandingan waktu eksekusi")
print("Sekuensial :", sekuensial_akhir - sekuensial_awal, "detik")
print("Kelas Process :", process_akhir - process_awal, "detik")
print("Kelas Pool :", pool_akhir - pool_awal, "detik")
```

```
------------------------------------------------------------
Perbandingan waktu eksekusi
Sekuensial : 10.019890308380127 detik
Kelas Process : 1.8526279926300049 detik
Kelas Pool : 10.615742444992065 detik
```

Sudah sewajarnya proses sekuensial lebih lambat dibanding multiprocessing namun bukan berarti kita harus melakukan multiprocessing terus menerus, gunakan metode sesuai kebutuhan. Nah apabila barisan kode di atas dikumpulkan jadi satu maka jadinya akan seperti ini.

LATIHAN SOAL

Dengam menggunakan pemrosesan parallel, buatlah program yang dapat menentukan sebuah bilangan itu ganjil atau genap!

Batasan :

- Nilai yang dijadikan argument pada fungsi sleep() adalah 1 detik.
- Masukkan jumlahnya satu dan berupa bilangan bulat.
- Masukkan adalah batas dari perulangan tersebut.
- Setelah perulangan selesai, program menampilkan waktu eksekusi pemrosesan sekuensial dan parallel.

Contoh input dan output dari program yang akan dibuat

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
```

PENYELESAIAN

Buat File .py di terminal linux dengan cara ketik nano nama_file.py.

```
mint@mint:~/Tugas8$ nano Tugas8.py
```

Built-in Libraries yang digunakan.

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

Beberapa libraries yang digunakan bertujuan untuk :

- **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 library multiprocessing yang digunakan untuk melakukan pemrosesan parallel dengan menggunakan proses sebanyak jumlah CPU pada computer.
- **Process** adalah sebuah class library multiprocessing yang digunakan untuk melakukan pemrosesan parallel dengan menggunakan proses secara beruntun pada computer.

Inisialisasi fungsi yang akan digunakan.

```
#inisialisasi fungsi yang akan digunakan
def cetak(i) :
    end = i + 1
    x = (end % 2)
    if x == 0 :
        print(end, "Genap - ID proses", getpid())
    else :
        print(end, "Ganjil - ID proses", getpid())
    sleep(1)

print("Tugas 8 Sistem Operasi")
print("By : Lisya")

print("----------------------------------------------------------------------")
a = int(input("Masukkan batas angka : "))
```

Fungsi diatas digunakan untuk mencetak angka dari variabel i beserta IP proses sejumlah parameter yang diberikan. Oleh karena program yang akan dibuat yaitu menentukan bilangan ganji genap. Maka dalam fungsi ini juga akan dibuat sebuah percabangan untuk menentukan ganjil genap suatu bilangan.

Dibuat sebuah variabel end, dimana end ini menampung i + 1. Kemudian, dibuat variabel baru, yaitu x. Dimana, x ini adalah variabel yang menampung hasil sisa bagi dari i mod 2. Dari hasil sisa bagi itulah, dapat dibuat sebuah percabangan untuk membuat keputusan yang menentukan bilangan ini ganjil atau genap. Jika x == 0, maka bilangan i ini adalah bilangan genap. Maka, akan dicetak bahwa bilangan ini genap beserta ID prosesnya. Lain, jika x ≠ 0, maka bilangan i ini adalah bilangan ganjil. Maka, akan dicetak bawah bilangan ini adalah bilangan ganjil beserta ID prosesnya. Dalam fungsi ini, variabel i kita panggil fungsi sleep untuk memberi jeda waktu (detik) sebanyak parameter yang diberikan.

Setelah dideklarasikan, dibuat heading dari program. Selain itu, dibuat juga variabel a. Yang mana variabel ini menampung inputan angka atau integer yang menjadi batas atas.

Jenis multiprocessing dan implementasi script-nya.

1. Pemrosesan Sekuensial

```
print("-------------------------------------------------------------------------")
print("1. Pemrosesan Sekuensial")
#untuk mendapatkan waktu sebelum eksekusi
sekuensial_awal = time()

#proses berlangsung
for i in range(a):
    cetak(i)

#untuk mendapatkan waktu setelah eksekusi
sekuensial_akhir = time()
```

2. Multiprocessing dengan kelas Process

```
print("-------------------------------------------------------------------------")
print("2. Multiprocessing dengan kelas Process")
#untuk menampung proses-proses
kumpulan_proses = []

#untuk mendapatkan waktu sebelum eksekusi
process_awal = time()

#proses berlangsung
for i in range(a):
    p = Process(target=cetak, args=(i,))
    kumpulan_proses.append(p)
    p.start()

#untuk menggabungkan proses-proses agar tidak loncat ke proses sebelumnya
for i in kumpulan_proses:
    p.join()

#untuk mendapatkan waktu setelah eksekusi
process_akhir = time()
```

3. Multiprocessing dengan kelas Pool

```
print("-------------------------------------------------------------------------")
print("3. Multiprocessing dengan kelas Pool")
#untuk mendapatkan waktu sebelum eksekusi
pool_awal = time()

#proses berlangsung
pool = Pool()
pool.map(cetak, range(0, a))
pool.close()

#untuk mendapatkan waktu sebelum eksekusi
pool_akhir = time()

print("-------------------------------------------------------------------------")
print("Perbandingan waktu eksekusi")
print("Sekuensial : ", sekuensial_akhir - sekuensial_awal, "detik")
print("Kelas Process : ", process_akhir - process_awal, "detik")
print("Kelas Pool :", pool_akhir - pool_awal, "detik")
```

Perbandingan waktu eksekusi

```
print("-------------------------------------------------------------------------")
print("Perbandingan waktu eksekusi")
print("Sekuensial : ", sekuensial_akhir - sekuensial_awal, "detik")
print("Kelas Process : ", process_akhir - process_awal, "detik")
print("Kelas Pool :", pool_akhir - pool_awal, "detik")
```

Setelah script dibuat, jalankan dengan mengetikkan python nama_file.py atau python3 nama_file.py.

```
mint@mint:~/Tugas8$ python3 Tugas8.py
```

Setelah program berjalan. User diminta menginputkan angka sebagai batasan. Misal diinputkan angka 3. Kemudian klik enter.

```
mint@mint:~/Tugas8$ python3 Tugas8.py
Tugas 8 Sistem Operasi
By : Lisya
------------------------------------------------------------------
Masukkan batas angka : 3
```

Maka, program akan berjalan dan mengeluarkan output seperti dibawah ini.

```
mint@mint:~/Tugas8$ python3 Tugas8.py
Tugas 8 Sistem Operasi
By : Lisya
------------------------------------------------------------------
Masukkan batas angka : 3
------------------------------------------------------------------
1. Pemrosesan Sekuensial
1 Ganjil - ID proses 3381
2 Genap - ID proses 3381
3 Ganjil - ID proses 3381
------------------------------------------------------------------
2. Multiprocessing dengan kelas Process
1 Ganjil - ID proses 3382
2 Genap - ID proses 3383
3 Ganjil - ID proses 3384
------------------------------------------------------------------
3. Multiprocessing dengan kelas Pool
1 Ganjil - ID proses 3385
2 Genap - ID proses 3385
3 Ganjil - ID proses 3385
------------------------------------------------------------------
Perbandingan waktu eksekusi
Sekuensial :  3.0079445838928223 detik
Kelas Process :  1.4257490634918213 detik
Kelas Pool : 3.563615560531616 detik
mint@mint:~/Tugas8$
```