Nama: Dwinggrit Oktaviani Putri
NPM: 21083010012
Mata Kuliah: Sistem Operasi / B

Multiprocessing

Multiprocessing atau Pemrograman Paralel adalah sebuah Teknik eksekusi yang dilakukan secara bersamaan pada CPU. Multiprocessing terdiri datri 2 jenis, yaitu Paralel Processing dan Serial Processing.

Manfaat Multiprocessing antara lain:

1. Menggunakan CPU untuk komputasi.
2. Tidak berbagi sumber daya memory.
3. Memerlukan sumber daya memori daya waktu yang tidak sedikit.
4. Tidak memerlukan sinkronisasi memori.

Soal Latihan:

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

Batasan:

a. Nilai yang dijadikan argument pada fungsi sleep() adalah satu detik.
b. Masukkan jumlahnya satu dan berupa bilangan bulat.
c. Masukkan adalah batas dari perulangan tersebut.
d. Setelah perulangan selesai, program dapat menampilkan waktu eksekusi pemrosesan sekuensial paralel.

Jawab:

- Membuat file .py dengan perintah “nano.Tugas_8.py”
- Membuat syntax/codding sesuai dengan perintah pada soal Latihan diatas.
- Untuk mengetahui outputnya, kitab isa menggunakan perintah “python3 Tugas_8.py”.
- Setelah itu nantinya output yang keluar akan sesuai dengan perintah dari soal Latihan. Yaitu, terdapat jeda 1 detik untuk menuju ke nilai argument selanjutnya.

Syntax:

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

print("Masukkan bilangan :")
bilangan = int(input())

def cetak(i):
    for i in range(bilangan):
        if i % 2 == 0:
           print(f"{i+1} Ganjil","- ID proses", getpid())
        else:
           print(f"{i+1} Genap","- ID proses", getpid())
    sleep(1)
print(" ")
```

Memasukkan library yaitu

1. getpid" untuk mengambil ID proses.
2. time untuk mengambil waktu (detik)
3. sleep untuk mendeklarasikan jeda waktu (detik)
4. cpu_count untuk melihat jumlah cpu
5. pool untuk melakukan pemrosesan paralel menggunakan proses sebanyak jumlah cpu
6. process untuk melakukan pemrosesan paralel menggunakan proses secara beruntun pada computer.

Mendeklarasikan fungsi print dan syntax bilangan untuk memasukkan bilangan yang diinginkan nantinya. Syntax def cetak digunakan untuk mendeklarasikan angka yang termasuk bilangan ganjil/genap beserta ID proses sejumlah parameter yang digunakan dan sleep(1) untuk jeda 1 detik Ketika menuju ke nilai argument selanjutnya.

Multiprocessing sekuensial:

```
#multiprocessing sekuensial
print("Sekuensial")
sekuensial_awal = time()

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

Menggunakan looping for dan if else.

1. Syntax **sekuensial_awal** untuk mendapatkan waktu sebelum di eksekusi.
2. Syntax **for** digunakan untuk berlangsungnya proses sekuensial.
3. **Range(1)** digunakan untuk mengeluarkan output hanya 1x.
4. Syntax **sekuensial_akhir = time()** digunakan untuk mendapatkan waktu setelah dieksekusi.
5. **Print(" ")** untuk memberikan spasi pada output yang akan dieksekusi.

Multiprocessing process:

```
#multiprocessing process
print("multiprocessing.Process")
kumpulan_proses = []
process_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()
process_akhir = time()
print(" ")
```

1. Syntax **process_awal = time()** untuk mendapatkan waktu sebelum di eksekusi.
2. Syntax **for** dan **range(1)** untuk proses berlangsungnya.
3. Syntax **for i in kumpulan-proses:** digunakan untuk menghubungkan proses-proses secara berurutan.

Multipocessing pool:

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

Syntax yang digunakan hamper sama dengan syntax multiprocessing sekuensial, ataupun multiprocessing process. Hanya saja ditambah dengan syntax **pool.map(cetak, range(0,1)** yang digunakan untuk memetakkan panggilan fungsi cetak kedalam sebanyak 1x.

```
#membandingkan waktu eksekusi
print("waktu eksekusi sekuensial :", sekuensial_akhir - sekuensial_awal, "detik")
print("waktu eksekusi multiprocessing.Process :", process_akhir - process_awal, "detik")
print("waktu eksekusi multiprocessing.Pool :", pool_akhir - pool_awal, "detik")
```

Syntax …._akhir-…_awal untuk mendapatkan waktu eksekusi yang nantinya akan dibandingkan dengan 3 argumen yaitu multiprocessing sekuensial, multiprocessing process, multiprocessing pool

Output:

```
oktaviani@oktaviani-VirtualBox:~$ nano Tugas_8.py
oktaviani@oktaviani-VirtualBox:~$ python3 Tugas_8.py
Masukkan bilangan :
3

Sekuensial
1 Ganjil - ID proses 2294
2 Genap - ID proses 2294
3 Ganjil - ID proses 2294

multiprocessing.Process
1 Ganjil - ID proses 2296
2 Genap - ID proses 2296
3 Ganjil - ID proses 2296

multiprocessing.Pool
1 Ganjil - ID proses 2297
2 Genap - ID proses 2297
3 Ganjil - ID proses 2297

waktu eksekusi sekuensial : 1.0008478164672852 detik
waktu eksekusi multiprocessing.Process : 1.0118920803070068 detik
waktu eksekusi multiprocessing.Pool : 1.0656790733337402 detik
```