TUGAS 8

NAMA : ACH. ARIF SETIAWAN
NPM : 21083010014
KELAS : SISTEM OPERASI (B)

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.

Contoh Program:


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
```

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

Lakukan import beberapa library dan modul yang akan dipakai, seperti:
getpid digunakan untuk mendapatkan id proses,
time digunakan untuk mengambil waktu pada proses dijalankan atau diakhiri,
sleep digunakan untuk memberi jeda waktu eksekusi dalam satuan detik,
cpu_count digunakan untuk melihat jumlah CPU,
Pool digunakan untuk melakukkan pemrosesan pararel dengan menggunakan proses sebanyak jumlah CPU pada komputer,
Process digunakan untuk melakukkan pemrosesan pararel dengan menggunakan process secara beruntun padad komputer.

```
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("Masukkan angka untuk batasan perulangan: "))
```

Selanjutnya, membuat sebuah fungsi dengan nama “cetak” yang akan digunakan untuk melihat angka yang masuk apakah ganjil atau genap. Ketika angka yang dimasukkan oleh user menghasilkan angka 0 setelah dilakukan proses modulo 2, maka angka yang dimasukkan oleh user tersebut adalah genap. Apabila menghasilkan angka lainya, maka angka tersebut adalah ganjil. Angka yang dimasukkan oleh user akan disimpan ke dalam variabel “n”. dimana angka tersebut haruslah berupa bilangan bulat, ditandai dengan perintah “int” untuk mendefinisikan tipe data integer.

```
#SEKUENSIAL
sekuensial_awal = time()
print("Sekuensial".center(50,"="))
for i in range(n):
   cetak(i)
sekuensial_akhir=time()
```

Pertama menggunakan pemrosesan sekuensial. Variabel “sekuensial_awal” dan “sekuensial_akhir” berfungsi untuk menyimpan waktu durasi selama proses sekuensial berlangsung. Perulangan tersebut akan mencetak setiap angka ganjil atau genap dengan proses id nya masing-masing sebanyak angka yang dimasukkan oleh user.

```
#MULTIPROCESSING DENGAN KELAS PROCESS
process_awal=time()
print("Multiprocess.process".center(50,"="))
for i in range(n):
    p=Process(target=cetak, args=(i, ))
    p.start()
    p.join()
process_akhir=time()
```

Pertama menggunakan pemrosesan dengan kelas process. Variabel “process_awal” dan “process_akhir” berfungsi untuk menyimpan waktu durasi selama multiprosessing kelas process berlangsung. Perulangan tersebut akan mencetak setiap angka ganjil atau genap dengan proses id nya masing-masing sebanyak angka yang dimasukkan oleh user. Perintah “P.start()” digunakan untuk mengeksekusi fungsi “cetak” di kelas process. Perintah “P.join()” digunakan untuk menunggu beberapa saat sampai proses sebelumnya selesai. Sehingga akan menghasilkan id proses yang berbeda-beda pada tiap proses.

```
#MULTIPROCESSING DENGAN KELAS POOL
pool_awal=time()
pool = Pool()
print("Multiprocess.pool".center(50,"="))
pool.map(cetak,range(0,n))
pool.close()
pool_akhir=time()
```

Ketiga menggunakan pemrosesan dengan kelas pool. Variabel “pool_awal” dan “pool_akhir” berfungsi untuk menyimpan waktu durasi selama multiprosessing kelas process berlangsung. Mendefinisikkan fungsi “Pool()” sebagai “pool”. Perintah “map” berfungsi untuk memetakan fungsi “cetak” ke dalam setiap CPU yang tersedia sebanyak 0 sampai dengan n kali, dimana n adalah inputan dari user.

```
#BANDIGKAN WAKTU EKSEKUSI
print("Perbandingan waktu".center(50,"="))
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")
```

Kemudian, adalah membandingkan berapa lama setiap jenis pemrosesan tersebut berlangsung.

```
setiawant@setiawant-VirtualBox:~$ python3 Tugas_8.py
Masukkan angka untuk batasan perulangan: 6
====================Sekuensial====================
1 ganjil - ID Process 2765
2 genap - ID Process 2765
3 ganjil - ID Process 2765
4 genap - ID Process 2765
5 ganjil - ID Process 2765
6 genap - ID Process 2765
==============Multiprocess.process==============
1 ganjil - ID Process 2766
2 genap - ID Process 2767
3 ganjil - ID Process 2768
4 genap - ID Process 2769
5 ganjil - ID Process 2770
6 genap - ID Process 2771
================Multiprocess.pool================
1 ganjil - ID Process 2772
2 genap - ID Process 2772
3 ganjil - ID Process 2772
4 genap - ID Process 2772
5 ganjil - ID Process 2772
6 genap - ID Process 2772
================Perbandingan waktu================
Waktu eksekusi Sekuensial: 6.00564169883728 detik
Waktu eksekusi multiprocessing.Process: 6.0911290645599365 detik
Waktu eksekusi multiprocessing.Pool: 6.107711315155029 detik
```

Pada hasil output tersebut user akan diminta untuk memasukkan sebuah angka batasan.

Pada pemrosessan sekuensial, terlihat bahwa setiap id proses nya adalah sama. Hal ini dikarenakan pemrosesan sekuensial akan mengeksekusi pada pemrosesan yang sama. Pada pemrosessan kelas process, terlihat bahwa setiap id prosesnya berbeda dan berurutan. Hal ini menunjukan bahwa setiap pemanggilan fungsi “cetak” ditagani oleh satu proses saja. Pada pemrosessan kelas pool, terlihat bahwa id proses nya dalah sama. Karena hanya terdapat 1 buah CPU dan dipetakan menjadi 1 pemrosessan di tiap CPU yang sama. Angka tidak urut menandakan terjadinya pemrosessan secara pararel. Terlihat dari perbandingan waktunya, pemrosessan sekuensial adalah yang terbaik dengan durasi paling singkat.