

Nama : Yosua Satria Bara Harmoni

NPM : 21083010029

MATA KULIAH SISTEM OPERASI

KELAS A

PERTEMUAN 8

Soal latihan :

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

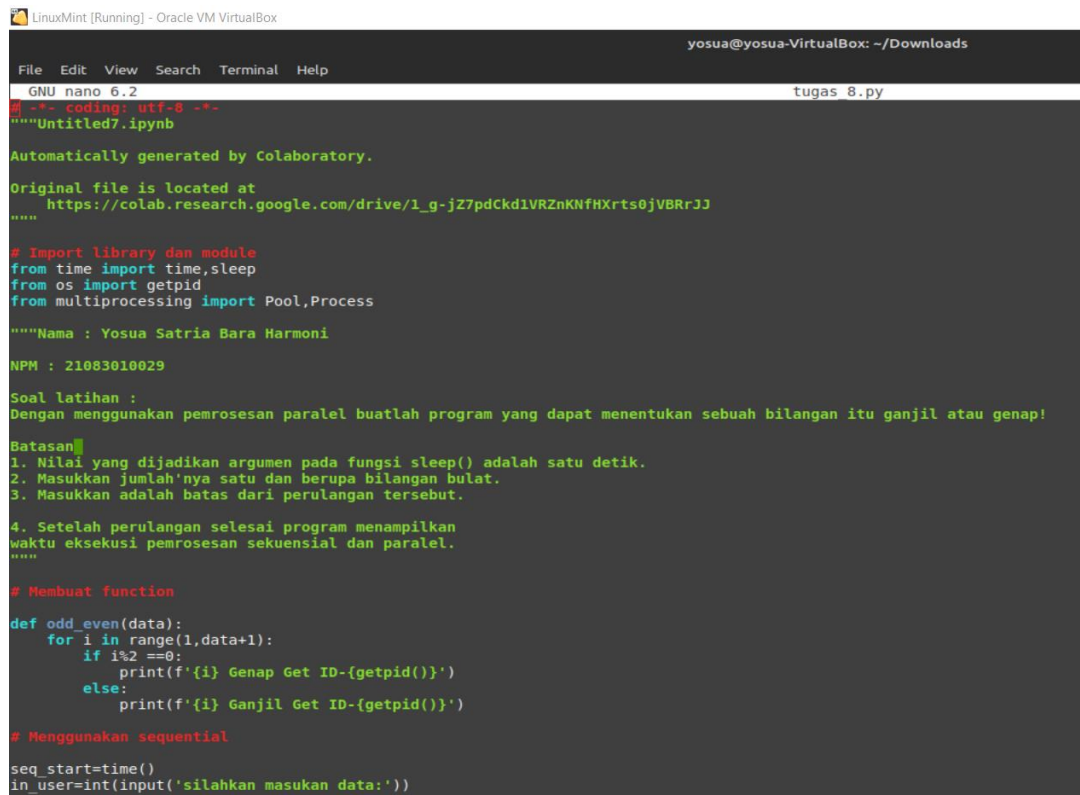
Batasan :

1. Nilai yang dijadikan argumen pada fungsi sleep() adalah satu detik.
2. Masukkan jumlah'nya satu dan berupa bilangan bulat.
3. Masukkan adalah batas dari perulangan tersebut.
4. Setelah perulangan selesai program menampilkan waktu eksekusi pemrosesan sekuensial dan paralel.

Jawab :

Pada script yang sudah saya buat (ipynb) akan berisi program yang berfungsi untuk mencetak bilangan bulat positif beserta tipe (ganjil/genap) berdasarkan batasan bilangan yang ditentukan oleh user dengan pemrosesan sekuensial, multiprocessing dengan kelas Process, dan multiprocessing dengan kelas Pool.

1. Nano tugas_8.py



```
LinuxMint [Running] - Oracle VM VirtualBox
yosua@yosua-VirtualBox: ~/Downloads

File Edit View Search Terminal Help
GNU nano 6.2 tugas_8.py
- coding: utf-8 -
Untitled7.ipynb

Automatically generated by Colaboratory.

Original file is located at
https://colab.research.google.com/drive/1_g-jZ7pdCkd1VRZnKNfHXrts0jVBRrJJ

# Import library dan module
from time import time,sleep
from os import getpid
from multiprocessing import Pool,Process

"""Nama : Yosua Satria Bara Harmoni
NPM : 21083010029

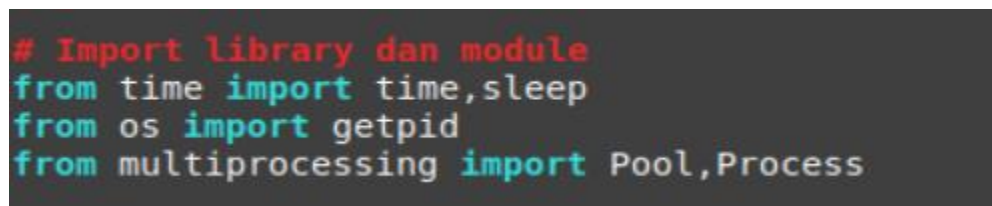
Soal latihan :
Dengan menggunakan pemrosesan paralel buatlah program yang dapat menentukan sebuah bilangan itu ganjil atau genap!

Batasan
1. Nilai yang dijadikan argumen pada fungsi sleep() adalah satu detik.
2. Masukkan jumlah'nya satu dan berupa bilangan bulat.
3. Masukkan adalah batas dari perulangan tersebut.
4. Setelah perulangan selesai program menampilkan waktu eksekusi pemrosesan sekuensial dan paralel.
"""

# Membuat function
def odd_even(data):
    for i in range(1,data+1):
        if i%2 ==0:
            print(f'{i} Genap Get ID-{getpid()}')
        else:
            print(f'{i} Ganjil Get ID-{getpid()}')

# Menggunakan sequential
seq_start=time()
in_user=int(input('silahkan masukan data:'))
```

2. Import library dan module



```
# Import library dan module
from time import time,sleep
from os import getpid
from multiprocessing import Pool,Process
```

1. `getpid` : Merupakan sebuah function yang mengembalikan (return) ID proses yang sedang berjalan.
2. `time` : Merupakan sebuah function yang berfungsi untuk mengambil waktu (detik).
3. `sleep` : Merupakan sebuah function yang berfungsi untuk menangguhkan eksekusi perintah dalam jumlah waktu (detik) yang diberikan.
4. `Process` : Merupakan sebuah class pada library multiprocessing yang digunakan untuk melakukan pemrosesan paralel dengan menggunakan proses secara beruntun pada komputer
5. `Pool` : Merupakan sebuah class pada library multiprocessing yang digunakan untuk melakukan pemrosesan paralel dengan menggunakan proses sebanyak jumlah CPU yang terdapat pada komputer.

3. Membuat function

```
# Membuat function

def odd_even(data):
    for i in range(1,data+1):
        if i%2 ==0:
            print(f'{i} Genap Get ID-{getpid()}')
        else:
            print(f'{i} Ganjil Get ID-{getpid()}')
```

4. Menggunakan sequential

```
# Menggunakan sequential

seq_start=time()
in_user=int(input('silahkan masukan data:'))
odd_even(in_user)
seq_finish=time()

seq_finish-seq_start
```

Dilakukan perintah inputan 'silahkan masukkan data' pada syntax sequential agar user bisa memasukkan angka sesuai kemauan user. Setelah itu, keluar data ganjil genap sesuai pilihan angka user. Misalnya user memasukkan angka 10, maka akan keluar 10 data dengan 5 ganjil dan 5 genap.

```
silahkan masukan data:10
1 Ganjil Get ID-2718
2 Genap Get ID-2718
3 Ganjil Get ID-2718
4 Genap Get ID-2718
5 Ganjil Get ID-2718
6 Genap Get ID-2718
7 Ganjil Get ID-2718
8 Genap Get ID-2718
9 Ganjil Get ID-2718
10 Genap Get ID-2718
```

a. Seq_start (sekuensial awal)

Digunakan function `time()` untuk mengambil waktu (detik) yang disimpan dalam variabel `seq_start`.

- b. Seq_finish (sekuensial akhir)

Digunakan function time() untuk mengambil waktu (detik) yang disimpan dalam variabel seq_finish.

5. Menggunakan process

```
# Menggunakan process

process_start=time()
memo=[]
for i in range(in_user):
    p=Process(target=odd_even,args=(i,))
    memo.append(p)
    p.start()
for sub in memo:
    p.join()

process_end=time()
```

Dilakukan deklarasi multiprocessing dengan perintah process untuk menandakan bahwa program yang berjalan di bawah kalimat ini merupakan multiprocessing dengan kelas Process dan digunakan untuk mencetak baris baru.

- a. process_start = time()
Digunakan function time() untuk mengambil waktu (detik) yang disimpan dalam variabel
- b. Dideklarasikan sebuah list kosong ([])
- c. For i in range (in_user)
Dilakukan perulangan for untuk i dalam range yang di dalamnya dipanggil dan digunakan kelas Process.
- d. For sub in memo
Digunakan perintah join() terhadap variabel p p.join() untuk menggabungkan kumpulan proses yang telah ditampung agar tidak merambah ke proses selanjutnya.
- e. process_end = time()
Digunakan function time() untuk mengambil waktu (detik) yang disimpan

6. Menggunakan pool

```
# Menggunakan pool

pool_start=time()
pool=Pool()
pool.map(odd_even,range(1,in_user+1))
```

Dilakukan deklarasi multiprocessing dengan perintah pool untuk menandakan bahwa program yang berjalan di bawah kalimat ini merupakan multiprocessing dengan kelas Pool serta digunakan untuk mencetak baris baru.

a. `pool_start = time()`

Digunakan function `time()` untuk mengambil waktu (detik) yang disimpan dalam variabel `pool_start`.

b. Dideklarasikan kelas `pool.map`

Digunakan method `map()` terhadap variabel `pool` (`pool.map()`) untuk memetakan pemanggilan function cetak

7. Python3 tugas_8.py

```
yosua@yosua-VirtualBox:~/Downloads$ python3 tugas_8.py
silahkan masukan data:10
1 Ganjil Get ID-2718
2 Genap Get ID-2718
3 Ganjil Get ID-2718
4 Genap Get ID-2718
5 Ganjil Get ID-2718
6 Genap Get ID-2718
7 Ganjil Get ID-2718
8 Genap Get ID-2718
9 Ganjil Get ID-2718
10 Genap Get ID-2718
1 Ganjil Get ID-2720
1 Ganjil Get ID-2721
2 Genap Get ID-2721
1 Ganjil Get ID-2722
2 Genap Get ID-2722
3 Ganjil Get ID-2722
1 Ganjil Get ID-2725
1 Ganjil Get ID-2724
1 Ganjil Get ID-2723
2 Genap Get ID-2723
3 Ganjil Get ID-2723
4 Genap Get ID-2723
2 Genap Get ID-2724
3 Ganjil Get ID-2724
4 Genap Get ID-2724
5 Ganjil Get ID-2724
2 Genap Get ID-2725
3 Ganjil Get ID-2725
4 Genap Get ID-2725
5 Ganjil Get ID-2725
6 Genap Get ID-2725
1 Ganjil Get ID-2726
2 Genap Get ID-2726
3 Ganjil Get ID-2726
4 Genap Get ID-2726
5 Ganjil Get ID-2726
1 Ganjil Get ID-2727
2 Genap Get ID-2727
3 Ganjil Get ID-2727
4 Genap Get ID-2727
5 Ganjil Get ID-2727
```

Input user : 10

- Pada pemrosesan sekuensial dapat dilihat bahwa untuk setiap bilangan yang dicetak melalui function cetak memiliki ID proses yang sama. Hal tersebut menandakan bahwa untuk semua pemanggilan function cetak ditangani oleh satu proses yang sama.
- Pada multiprocessing dengan kelas Process dapat dilihat bahwa untuk setiap bilangan yang dicetak melalui function cetak memiliki ID proses yang berbeda dan urut. Hal ini menandakan bahwa untuk setiap pemanggilan function cetak ditangani oleh satu proses saja.

- c. Pada multiprocessing dengan kelas Pool dapat dilihat bahwa untuk setiap bilangan yang dicetak melalui function cetak memiliki ID proses yang berbeda, tetapi perbedaan ID proses tersebut hanya terbatas pada 4 ID proses yang berulang dengan tidak urut. Hal tersebut terjadi karena pada komputer yang digunakan untuk menjalankan script tersebut hanya memiliki 4 CPU.