Nama: Yasmin Ulayya

NPM: 21083010033

Kelas: A

Tugas 8

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.

Di sini pertama kita import terlebih dahulu built-in libraries yang akan digunakan.

1. getpid digunakan untuk mengambil ID proses
2. time digunakan untuk mengambil waktu(detik)
3. sleep digunakan untuk memberi jeda waktu(detik)
4. cpu_count digunakan untuk melihat jumlah CPU
5. Pool adalah sebuah class pada library multiprocessing yang digunakan untuk melakukan pemrosesan paralel dengan menggunakan proses sebanyak jumlah CPU pada komputer
6. Process adalah sebuah class pada library multiprocessing yang digunakan untuk melakukan pemrosesan paralel dengan menggunakan proses secara beruntun pada komputer

Selanjutnya kita buat command input untuk memasukkan angka. Setelah itu kita meninisialisasi fungsi yang akan digunakan. Di sini kita membuat fungsi bernama cetak, dimana didalamnya memuat looping for. Untuk if jika i(angka yang diinputkan user) dimodulo 2 sama dengan 0 akan memprint i+1 ialah bilangan ganjil dengan id prosesnya memanfaatkan library getpid, jika inputan dimodulo 2 tidak sama dengan 0 akan masuk ke else dan memprint i+1 ialah bilangan genap dengan id prosesnya memanfaatkan library getpid. Setelah itu kita manfaatkan library sleep untuk mengambil jeda waktu 1 detik.

```
  GNU nano 6.2                          Tugas_8.py
from os import getpid
from time import time, sleep
from multiprocessing import cpu_count, Pool, Process

# Inisialisasi fungsi yang akan digunakan
angka = int(input("Masukkan angka:"))

def cetak(i):
    if i  % 2 == 0:
       print(i+1, "Ganjil", "- punya ID proses", getpid())
    else:
       print(i+1, "Genap", "- punya ID proses", getpid())
    sleep(i)
```

Pada sekuensial, pertama membuat variabel sekuensoal_awal yang digunakan iuntuk mendapatkan waktu sebelum eksekusi. Selanjutnya proses berlangsung dengan rangenya adalah variabel angka yang pertama kali kita input. Terakhir, membuat variabel sekuensil_akhir yang digunakan untuk mendapatkan waktu setelah eksekusi.

```
  GNU nano 6.2                          Tugas_8.py
# Sekuensial
print("Sekuensial")
sekuensial_awal = time()

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

sekuensial_akhir = time()
```

Pada multiprocessing process, pertama membuat variabel kumpulan_proses yang digunakan untuk menampung proses-proses. Lalu, membuat variabel process+awal untuk mendapatkan waktu sebelum eksekusi. Lalu, proses berlangsung. Membuat perulangan for i in kumpulan proses: utnuk menggabungkan proses-proses agar tidak loncat ke proses sebelumnya. Terakhir, membuat variabe process_akhir untuk mendapatkan waktu setelah eksekusi.

```
# Multiprocessing process
print("Multiprocessing.process")
kumpulan_proses = []
process_awal = time()

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

for i in kumpulan_proses:
    p.join()

process_akhir = time()
```

Pada multuprocessing pool, pertama mebuat variabel pool_awal yng digunakan utnuk mendapatkan waktu sebelum eksekusi. Selanjutnya proses berlangsung dengan rangenya adalah (0,angka) variabel angka yang pertama kali kita inputkan. Terakhir, membuat variabel pool_akhir yang digunakan untuk mendapatkan waktu setelah eksekusi,

```
# Multiprocessing pool
print("Multiprocessing.pool")
pool_awal = time()

pool = Pool()
pool.map(cetak, range(0,angka))
pool.close()

pool_akhir = time()
```

Di sini kita akan membandingkan waktu eksekusi. Untuk caranya ialah mengurangi proses akhir dengan proses awal, ini berlaku untuk sekuensial, multiprocessing process dan juga multiprocessing poll tinggal mengikuti nama variabel yang telah kita buat sebelumnya.

```
print("Waktu eksekusi Sekuansial :", 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")
```