Nama	**Imam Aprido Simarmata**
NIM	**11S18034**

Python Pipe and Queue

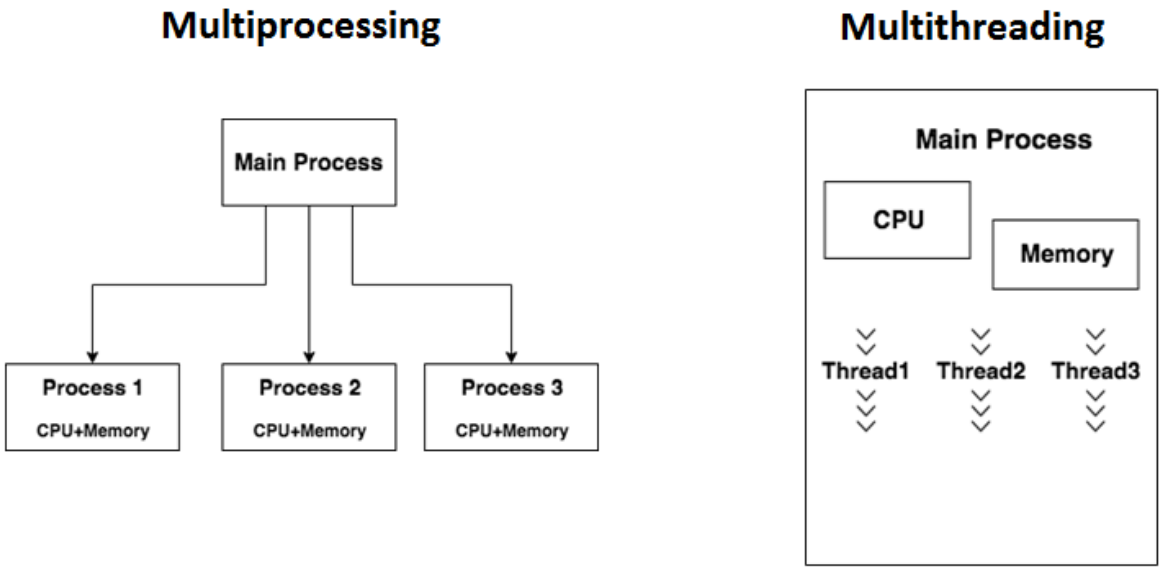
Simple multiprocessing

```
from concurrent.futures import process
from multiprocessing import Process

def say_hi():
    print("Hi from pid: ", end = "")


if __name__ == "__main__":
    p1 = Process(target=say_hi, args = ())
    p2 = Process(target=say_hi, args = ())

    p1.start()

    p2.start()

    p1.join()
    print(p1.pid)
    p2.join()
    print(p2.pid)
```

output

```
PS C:\Users\aprid\Documents\Lit> python3 .\PrakSPT.py
Hi from pid: 21496
Hi from pid: 18596
PS C:\Users\aprid\Documents\Lit>
```

Process means a program is in execution, whereas thread means a segment of a process. A Process is not Lightweight, whereas Threads are Lightweight. A Process takes more time to terminate, and the thread takes less time to terminate. Process takes more time for creation, whereas Thread takes less time for creation. – guru99



Dilihat dari gambar di atas, *thread* dinaungi oleh sebuah proses. Oleh karena itu, thread 1, thread 2, dan thread 3 berbagi ruang memori yang sama. Sebuah variabel x yang ada pada main process dapat mereka akses bersama. Lalu, bagaimana dengan multiprocessing? Setiap process memiliki ruang memori dan processor yang berbeda. Apa yang dapat dilakukan ketika process 1 perlu bertukar data dengan process 2?

Python Pipe

The Pipe() function returns a pair of connection objects connected by a pipe which by default is duplex (two-way). – docs.python.org

Sederhananya, Pipe memanglah pipa

A Pipe() can only have two endpoints. - https://stackoverflow.com/questions/8463008/multiprocessing-pipe-vs-queue

Multiprocessing with Pipe

```python
from multiprocessing import Process, Pipe

def say_hi(no, conn):
    if no == 1:
        print(f"Dari process {no}: Hi, aku Imam")
        while True:
            msg = conn.recv()
            if msg == "OK":
                print(f"Dari process {no}: OK, bye")
                break
    if no == 2:
        print(f"Dari process {no}: OK, Imam")
        conn.send("OK")



if __name__ == "__main__":
    p1_conn, p2_conn = Pipe()
    p1 = Process(target=say_hi, args = (1, p1_conn))
    p2 = Process(target=say_hi, args = (2, p2_conn))

    p1.start()

    p2.start()

    p1.join()
    p2.join()
```

Output

```
PS C:\Users\aprid\Documents\Lit> python3 .\PrakSPT.py
Dari process 1: Hi, aku Imam
Dari process 2: OK, Imam
Dari process 1: OK, bye
PS C:\Users\aprid\Documents\Lit>
```

Python Queue

Returns a process shared queue implemented using a pipe and a few locks/semaphores. When a process first puts an item on the queue a feeder thread is started which transfers objects from a buffer into the pipe. -- docs.python.org

Disebutkan bahwa queue juga mengimplementasikan Pipe(?) dengan beberapa lock/semaphore.

Mungkin Lock yang dimaksud adalah digunakan untuk menghindari *race condition* antar *process* dalam mengakses queue pada waktu yang sama.

Di bawah adalah contoh process synchronization yang tidak ada kaitannya dengan lock/semaphore yang diimplementasikan oleh kelas Queue.

Queue with custom lock

```
from multiprocessing import Process, Queue, Lock
import queue
from time import sleep

def insert(no, queue):
    print(f"Aku insert{no} akan menginsert {no} ke queue")
    queue.put(no)

def consumer(no, queue, lock):
    lock.acquire()
    if no == 1:
        sleep(1)
    print(f"aku consumer {no} harusnya mendapatkan angka {no} dari queue :
{queue.get()}")
    lock.release()

if __name__ == "__main__":
    lock = Lock()
```

```
    q = Queue(5)
    p1 = Process(target=insert, args = (1, q))
    p2 = Process(target=insert, args = (2, q))

    p1.start()

    p2.start()

    p1.join()
    p2.join()

    consumer1 = Process(target=consumer, args = (1, q, lock))
    consumer2 = Process(target=consumer, args = (2, q, lock))
    consumer1.start()
    consumer2.start()
    consumer1.join()
    consumer2.join()
```

Output tanpa lock

```
PS C:\Users\aprid\Documents\Lit> python3 .\PrakSPT.py
Aku insert1 akan menginsert 1 ke queue
Aku insert2 akan menginsert 2 ke queue
aku consumer 2 harusnya mendapatkan angka 2 dari queue : 1
aku consumer 1 harusnya mendapatkan angka 1 dari queue : 2
```

Output dengan lock

```
PS C:\Users\aprid\Documents\Lit> python3 .\PrakSPT.py
Aku insert1 akan menginsert 1 ke queue
Aku insert2 akan menginsert 2 ke queue
aku consumer 1 harusnya mendapatkan angka 1 dari queue : 1
aku consumer 2 harusnya mendapatkan angka 2 dari queue : 2
PS C:\Users\aprid\Documents\Lit>
```