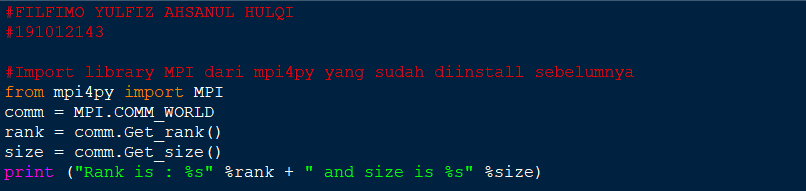
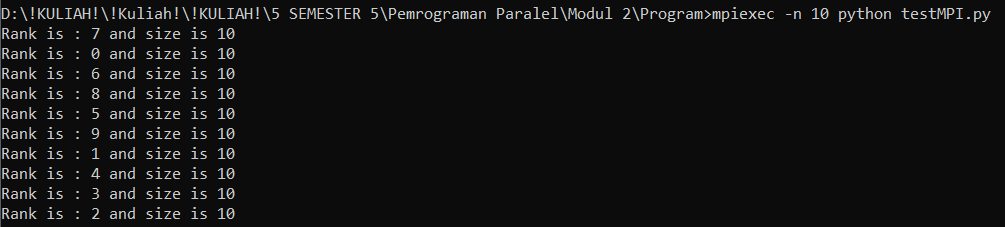
1. Setting MPI



Gambar 1 MPI Program



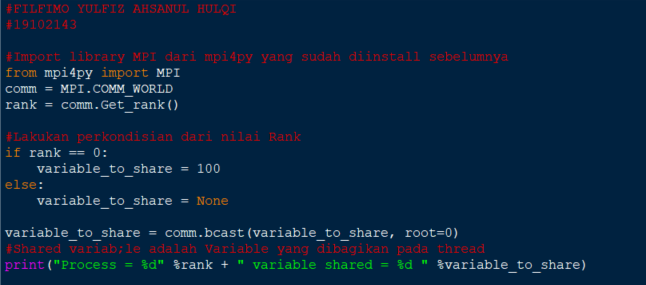
Gambar 2 Output MPI

Penjelasan Program :

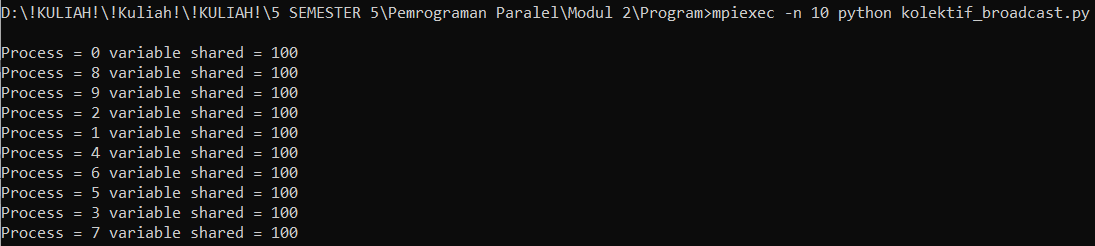
Program diatas adalah contoh progarm untuk melakukan MPI, pertama lakukan *import library* dari *mpi4py* yaitu *import MPI*, selanjutnya buat sebuah objek bernama *comm* yang memuat MPI\_COMM\_WORLD, yang berfungsi sebagai komunikator yang mendefinisikan jangkauan operasi. Selanjutnya ada *rank, comm.Get\_rank()*, yang berfungsi untuk mengembalikan *rank* dari suatu proses. Kemudian ada *size*, *comm.Get\_size()*, yang berfungsi untuk mencari banyaknya prosesor.

Setelah itu buat fungsi untuk melakukan *print*, didalamnya terdapat “*%rank*” dan “*%size*” yang merupakan inisialisasi dari *Get\_rank()*, dan *Get\_size()*. Terakhir lakukan eksekusi program dengan menjalankan perintah **mpiexec -n 10 python “nama program”.py** pada *command prompt* atau terminal.

1. Kolektif Broadcast



Gambar 3 Program Kolektif Broadcast



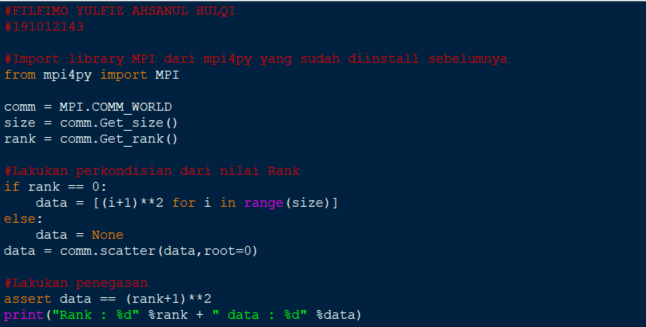
Gambar 4 Output Program Kolektif Broadcast

Penjelasan Program :

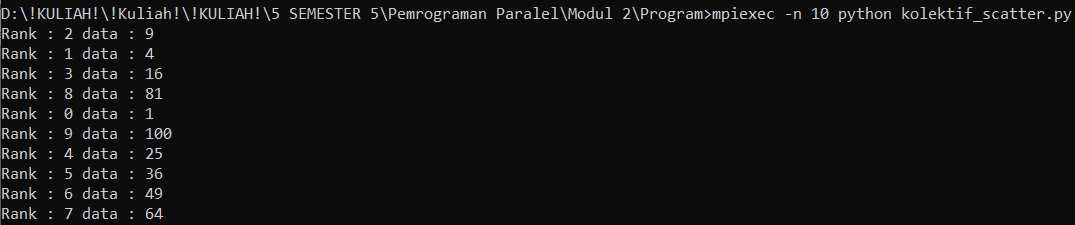
Program diatas adalah progarm untuk melakukan kolektif *broadcast*, pertama lakukan *import library* dari *mpi4py* yaitu *import MPI*, selanjutnya sama seperti program pertama, terdapat MPI\_COMM\_WORLD dan *comm.Get\_rank()*, pada program ini akan mencoba untuk melakukan pembagian variabel. Pada awalnya dilakukan perkondisian terhadap nilai variabel yang akan dibagikan, setelah itu lakukan *bcast* (mem-*broadcast*) sebuah pesan ke semua prosesor yang tergabung dalah sebuah *communicator*.

Terakhir lakukan eksekusi program dengan menjalankan perintah **mpiexec -n 10 python “nama program”.py** pada *command prompt* atau terminal.

1. Kolektif Scatter



Gambar 5 Program Kolektif Scatter

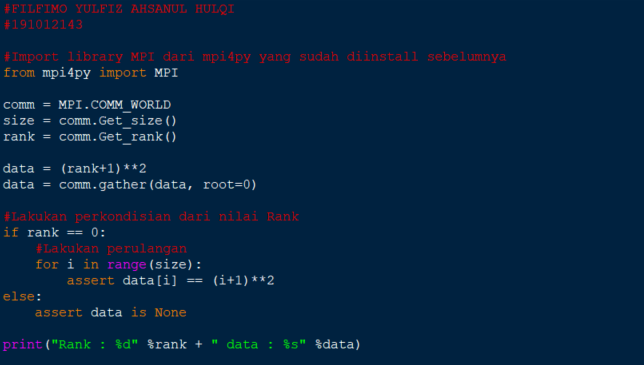


Gambar 6 Output Program Kolektif Scatter

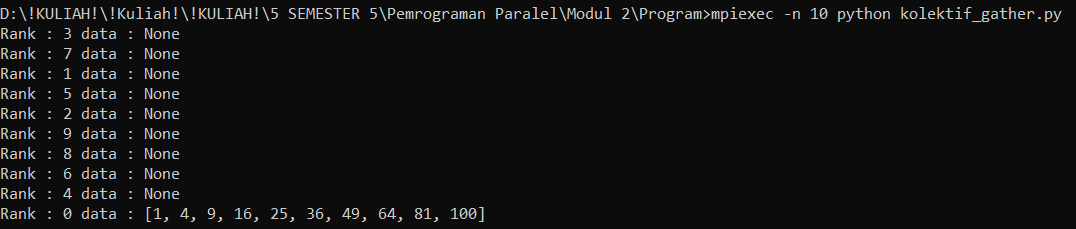
Penjelasan Program :

Program diatas adalah program untuk melakukan kolektif *scatter*, pertama lakukan *import library* dari *mpi4py* yaitu *import MPI*, selanjutnya sama seperti program pertama, terdapat MPI\_COMM\_WORLD, *comm.Get\_size()* dan *comm.Get\_rank()*. Pada program kali ini hampir sama seperti program sebelumnya (*Broadcast*) yang mengirimkan beberapa data ke proses, berbeda dengan MPI\_Bcast yang membagi data sama rata ke setiap proses, MPI\_Scatter mengirimkan *chunks* dari array ke beberapa proses yang berbeda.

1. Kolektif Gather



Gambar 7 Program Kolektif Gather

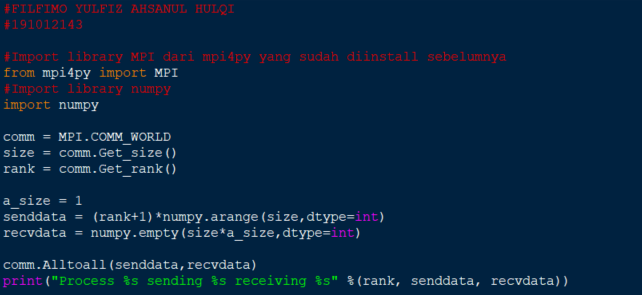


Gambar 8 Output Program Kolektif Gather

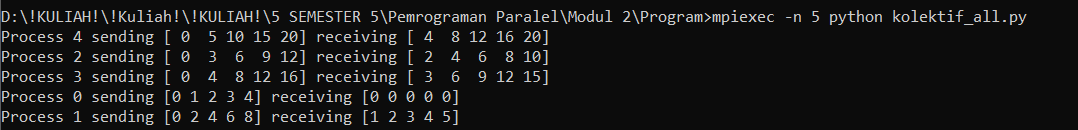
Penjelasan Program :

Program diatas adalah contoh program untuk melakukan kolektif *gather*, pertama lakukan *import library* dari *mpi4py* yaitu *import MPI*, selanjutnya sama seperti program pertama, terdapat MPI\_COMM\_WORLD, *comm.Get\_size()* dan *comm.Get\_rank().* Pada program kali ini adalah kebalikan dari kolektif *scatter*, dimana MPI\_Gather ini daripada menyebarkan elemen dari satu proses ke proses lainnya, MPI\_Gather mengambil elemen dari banyak proses dan mengumpulkannya dalam *single* proses, sehingga dapat terlihat seperti *output* program diatas dimana hanya satu proses saja yang memiliki elemen.

1. Kolektif all to all



Gambar 9 Program Kolektif All to All

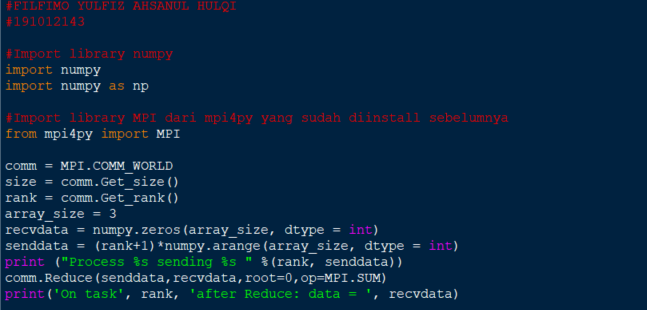


Gambar 10 Output Program Kolektif All to All

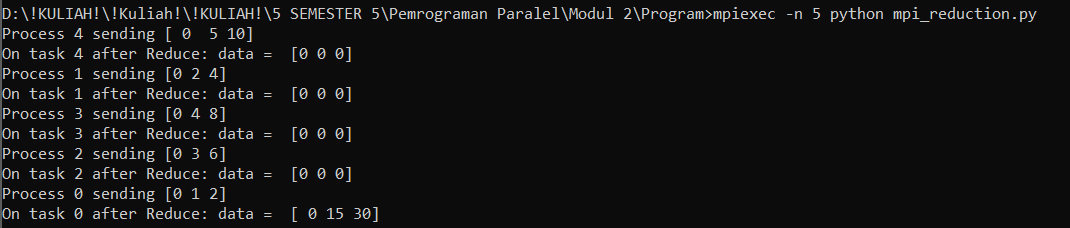
Penjelasan Program :

Program diatas adalah program untuk melakukan kolektif *all to all*, pertama lakukan *import library* dari *mpi4py* yaitu *import MPI*, selanjutnya sama seperti program pertama, terdapat MPI\_COMM\_WORLD, *comm.Get\_size()* dan *comm.Get\_rank()*, *import* juga *library numpy*. MPI\_Alltoall merupakan kombinasi dari *scatter* dan *gather*, yang mana setiap proses mempunyai *buffer* yang berisi elemen yang tersedia pada setiap proses, sebagai *buffer* didalam setiap elemen tersimpan akan dikumpulkan pada semua proses, hasil *output* akan seperti gambar diatas, dimana *array* elemen yang horizontal menjadi vertikal.

1. MPI Reduction



Gambar 11 Program MPI Reduction



Gambar 12 Output Program MPI Reduction

Penjelasan Program :

Program diatas adalah program untuk melakukan MPI *Reduction*, pertama lakukan *import library* dari *mpi4py* yaitu *import MPI*, selanjutnya sama seperti program pertama, terdapat MPI\_COMM\_WORLD, *comm.Get\_size()* dan *comm.Get\_rank()*. *Reduce* sesuai namanya berarti melakukan pengurangan, disini mengurangi nilai elemen array tetapi juga dapat digunakan untuk menjumlahkan juga. MPI\_Reduce mirip seperti MPI\_Gather, mengambil *array* dari elemen pada setiap proses dan mengembalikan *array* dari *output* elemen ke *root* proses. *Output* dari elemen akan berisi hasil yang telah berkurang. *senddata* adalah parameter dari elemen yang bertipe *datatype* yang setiap prosesnya akan di *reduce*. *recvdata* berisi hasil yang telah dilakuka*n reduce* dan memiliki *size*. Untuk *ouput* dapat dilihat pada gambar diatas.