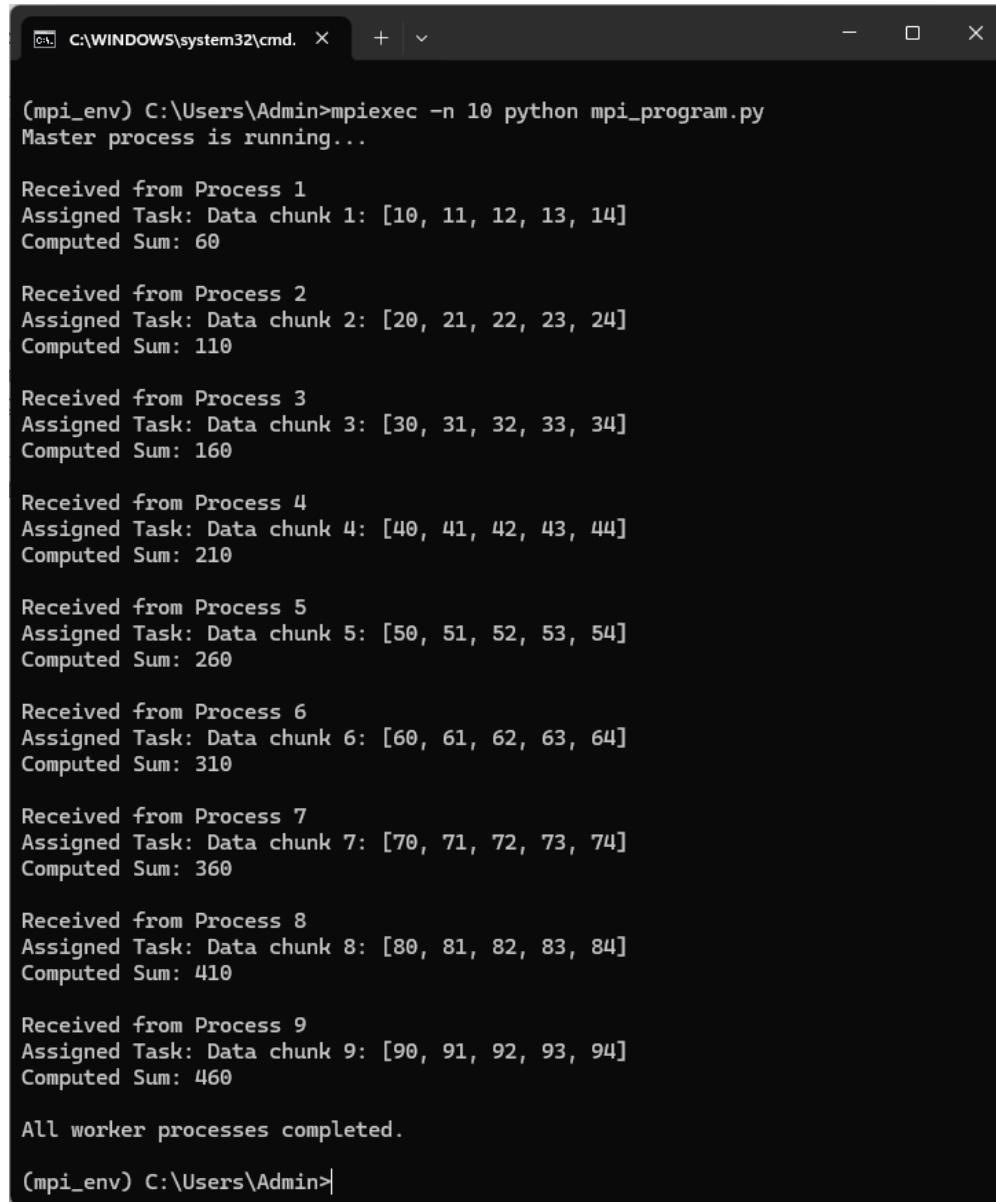


## Reflection

Message passing is needed because processes have their own memory and cannot share data directly. They must send messages to communicate and work together. If a process fails, the master might wait forever for its message, which can stop the program. Unlike shared-memory programming, distributed processes are separate and must use messages to share information. This makes distributed systems easier to run on many computers but a bit harder to manage.



The screenshot shows a Windows Command Prompt window titled 'C:\WINDOWS\system32\cmd.' with the path '(mpi\_env) C:\Users\Admin>'. The window displays the output of a Python script named 'mpi\_program.py' running with 10 processes. The master process starts by announcing its own task and then receives tasks from other processes, printing each task's data and the resulting sum. Finally, it declares that all worker processes have completed.

```
(mpi_env) C:\Users\Admin>mpiexec -n 10 python mpi_program.py
Master process is running...

Received from Process 1
Assigned Task: Data chunk 1: [10, 11, 12, 13, 14]
Computed Sum: 60

Received from Process 2
Assigned Task: Data chunk 2: [20, 21, 22, 23, 24]
Computed Sum: 110

Received from Process 3
Assigned Task: Data chunk 3: [30, 31, 32, 33, 34]
Computed Sum: 160

Received from Process 4
Assigned Task: Data chunk 4: [40, 41, 42, 43, 44]
Computed Sum: 210

Received from Process 5
Assigned Task: Data chunk 5: [50, 51, 52, 53, 54]
Computed Sum: 260

Received from Process 6
Assigned Task: Data chunk 6: [60, 61, 62, 63, 64]
Computed Sum: 310

Received from Process 7
Assigned Task: Data chunk 7: [70, 71, 72, 73, 74]
Computed Sum: 360

Received from Process 8
Assigned Task: Data chunk 8: [80, 81, 82, 83, 84]
Computed Sum: 410

Received from Process 9
Assigned Task: Data chunk 9: [90, 91, 92, 93, 94]
Computed Sum: 460

All worker processes completed.

(mp�_env) C:\Users\Admin>
```