**Pigeonhole Sort MPI Program**

**Algorithm**

* Initialize:

Import necessary libraries.

Define the pigeonhole sort function.

Main Function:

* Initialize MPI.

Get process rank and size.

If rank is 0, input total number of elements.

Broadcast total elements to all processes.

Divide data among processes using Scatter.

* Local Sorting:

Find local minimum and maximum values.

Reduce to get global minimum and maximum values.

Calculate range size for pigeonhole sorting.

* Pigeonhole Sort:

Perform pigeonhole sorting on local arrays.

* Gather and Print:

Gather sorted sub-arrays from all processes.

If rank is 0, print the sorted array.

* Finalize:

Finalize MPI.

* Return:

Return from main function.

**pseudocode**

* Import necessary libraries

Define pigeonhole\_sort function

Initialize local\_pigeonholes array

For each element in local\_arr

Increment corresponding pigeonhole

Construct sorted\_arr from pigeonholes

* Main Function

Initialize MPI

Get process rank and size

If rank is 0

Input total number of elements

Broadcast total elements to all processes

Divide data among processes using Scatter

* Find local minimum and maximum values

Reduce to get global minimum and maximum values

Calculate range size for pigeonhole sorting

* Perform pigeonhole sort on local arrays

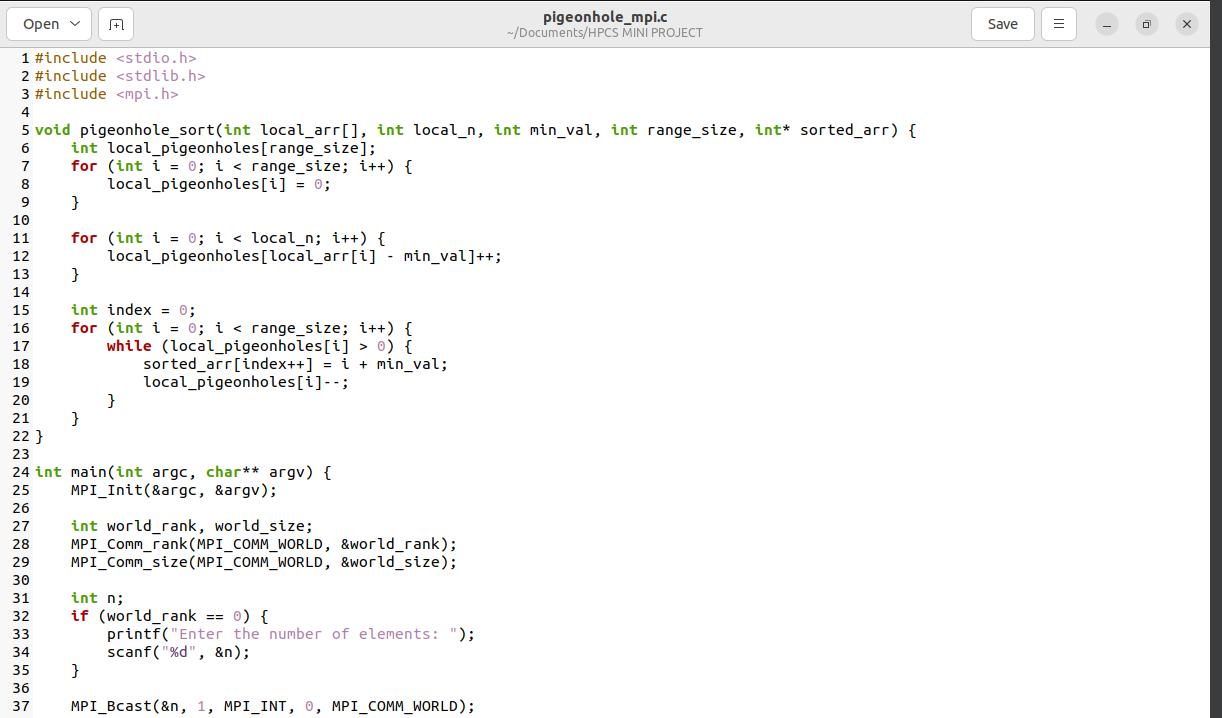
Gather sorted sub-arrays from all processes

If rank is 0

Print the sorted array

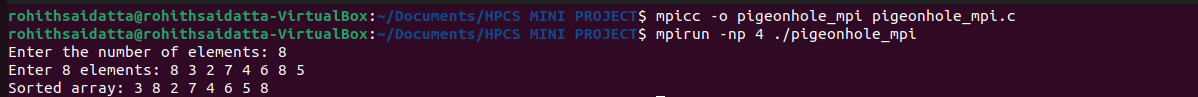
* Finalize MPI

Return from main function

A screenshot of a computer program

Description automatically generatedA computer code on a white background

Description automatically generated



* The compiled the code using mpicc and created an executable named pigeonhole\_mpi.
* user used mpirun to run the program with 4 processes.
* The program asked you to enter the number of elements, and user typed 8.
* Then, user entered the array elements: 8 3 2 7 4 6 8 5.
* The program sorted the array in parallel and displayed the sorted result: 3 8 2 7 4 6 5 8.
* The program finished execution.

So, the output shows that the program sorted the input array, and the sorted array is displayed at the end. Keep in mind that the order of equal elements might vary due to the sorting algorithm and parallel processing.