1. Find errors in the given code fragment, correct them.
2. Complete the code fragment to solve the problem:
   1 create objects: list, class object, numpy array
   2 the host should send messages to three Workers with the 1 - list, 2 - class object and 3 - numpy array.
   3 the workers have to display the obtained objects in the console.
3. Define the time between sending a message from a worker and receiving it at the host. This assumes that multiple workers can be run. The result of the program should be printed to the console in milliseconds.
4. Using the sleep function write a program in which worker sends a message to host and continues to run (outputs some message to console) while host is idle and gets the message only when sleep is over then outputs it to console.
5. A program to calculate dot-product in distributed mode: two vectors of size 100 000 must be initialized on host with values 1 (first vector) and 2 (second vector), each worker must receive their fragment of both vectors, perform element-by-element multiplication and sum the result, host then must collect the results from all vectors and calculate the final result by displaying it on the console
6. Determine the bandwidth when forwarding MPI messages depending on the size of the data.
   1 Create a list object of one int type element
   2 Measure the size L of the object using the sys library
   3 host should send a message to the worker, then the worker should send it back
   4 repeat point (3) N=10 times
   5 measure the time T from sending the first message to receiving the last one at the host
   6 bandwidth is measured by the formula: R=(2*N*L) / T
   7 output the result to the console in the format "object_size (bytes): R (MB/s)
   8 increase the number of sheet elements by 1000
   9 repeat from point (2) until the number of elements is equal to 50 000 elements

7. Write a program that implements a circular message exchange for the number of workers N=10:
   1 the host sends a message to the worker with rank 1
   2 the worker receives the message, displays his rank and the message in the console, and forwards it to the next rank
   3 the process stops when host receives the original message and outputs the "DONE" message

8. Develop a program that demonstrates the asynchronous communication mechanism:
   1 host sends a message in non-blocking mode
   2 host must spend 25 seconds in sleep mode before it receives the message
   3 after sending the message, the host should send a "WAITING" message to the console every 5 seconds until it receives confirmation of receiving the message.
9. Write a program which implements master-worker mode:
   1 host must create N worker processes by itself

2 when worker is created, he writes his rank in the console

3 each worker sends his rank to host

4 host shows the messages received in the console

For this program you need to write two scripts - host.py and worker.py, each will implement its own logic.