Introduction:

In this exercise, I modified the code search.cpp and made it palatalized on distributed memory system. MPI was used for implementation. After successfully implemented the code, the efficiency of the code was tested by setting the process number to be 1, 2, 4 and 8. Time was recorded separately.

Methodology:
The MPI was used to paralyze the code. The process rank 0 generate an array, and send the chunks of this array to worker process(including process 0). The worker processes and main processes then search the assigned chunks for the specific number. Then the results was reduced together and printed on the screen.

To make the code and timing work, several changes has been made on initial code:
1. I have reduce the size of b to 1/10 of original size, because the original size is too big for an array(more than 10GB).
2. Correspondingly, I need to change a test cases to get positive result, c=60 or c=3081 are positive test cases.
3. I also changed the search function so that it use array as a parameter. Also **to make sure that the timing process is meaningful I removed the break inside the for loop**. So that we can do the worst case analysis of this parallel programming.

Result:
The result was shown as Table 1:

Processor number	1	2	4	8
Execution time(s)	14.9446	7.48943	3.76658	2.74732

Table1: The execution time of the program with processor number 1, 2, 4, 8

Conclusion:
From the result above, we can see that as the process number doubled, the execution time decrease approximately by half, which indicate the successfully implementation of the code.
The decreasing of execution time is less significant when process number change from 4 to 8. This phenomenon may caused by the lagging of communication between nodes.