Faculty of Computers and Artificial Intelligence
Cairo University

HPC/ Parallel Processing
Assignment 3

Problem1: Encrypt a Message (Caesar Cipher)

[Using BCast, Scatter and Gather ONLY]

You are required to take a message from a file and encrypt the message using caesar cipher and store the output in a new file. The encryption can be represented using modular arithmetic by first transforming the letters into numbers, according to the scheme, A = 0, B = 1,..., Z = 25. Encryption of a letter x by a shift n can be described mathematically as:

$$E_n(x) = (x + n) \bmod 26$$

Any other character outside the range from A to Z (or a to z) is transferred to the output file as it is.

Input: filename and key/shift value.
Output: new file that contains the encrypted message.

Master Scenario:

- Read the entire message in a char array from the given file.
- Read the key (shift) value from the user.
- Broadcast the key value to all slaves
- Scatter message on all slaves
- Handle remaining part of a message *(if exists)*
- Gather encrypted message parts from slaves.
- Write the entire encrypted message to a file.

Slave Scenario:

- Receive the key (shift) value from master.
- Get the part of the message to be encrypted using scatter.
- Do the encryption on the received part.
- Send the encrypted part to the master using gather.

Sample input/output:

```
KEY/SHIFT: 23
Plaintext.txt:  THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG
Ciphertext.txt: QEB NRFZH YOLTK CLU GRJMP LSBO QEB IXWV ALD
```

Problem2: Value of PI

[Using Bcast and Reduce ONLY]

The following piece of code is a serial program to calculate the value of PI using integration, give it a try and run it to see the output.

```c
#include <stdio.h>
int main()
{
    static long num_steps = 1000000;
    double step;
    int i;
    double x, pi, sum = 0.0;
    step = 1.0/(double)num_steps;

    for(i = 0; i < num_steps; i++)
    {
        x= ((double)(i + 0.5))*step;
        sum += 4.0/(1.0+x*x);
    }
    pi = step*sum;
    printf("%.20f", pi);
    return 0;
}
```

You are required to parallelize the program by distributing the number of steps on multiple processes and broadcast the step value to all slaves:

- Each slave process will receive the step and the range it will work on, then calculate its partial sum.
- Master process will divide the number of steps into ranges and broadcast the value of step, then perform reduce to output the value of PI using the reduced sum.

Submission Rules (Read Carefully)

1. You can work with team members of any lab.
2. Max number of members per team is 2.
3. Your code should run on any number of processes and make sure to deliver a running code.
4. Deliver only your ".c" files in a zipped folder and don't include any ".exe" files.
5. Follow the naming convention: ID1_ID2_Group to name your folder.
6. You are required to deliver your own effort, cheating any part of your code from your colleagues or from the internet will lead to serious consequences.