1) Primos

O algoritmo abaixo, escrito em C, realiza a contagem de quantos primos existem entre 2 e o número inserido. Modifique o código para utilizar funções de comunicação ponto-a-ponto MPI (send e receive) e paralelizar a contagem. Utilize apenas troca de mensagens ponto-a-ponto.

Dica: Não esqueça de utilizar -lm na compilação.

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/* função que decide se n é primo, sendo n maior do que 1
retorna 1 caso o número seja primo, e 0 caso não seja */
int isprime(int n)
{
        int i, squareroot;
        if (n > 1) {
                squareroot = (int) sqrt((double) n);
                for (i = 3; i <= squareroot; i = i + 2) {
                        if ((n % i) == 0) {
                                return 0;
                        }
                }
                return 1;
        } else {
                return 0;
        }
}

int main(int argc, char **argv)
{
        int n,                  /* numero a ser testado */
         pc,                    /* prime counter */
         limit;                 /* maior número a ser testado */

        if(argc < 2) {
            printf("Digite o numero a ser testado\n");
            return 0;
        } else {
            sscanf(argv[1], "%d", &limit);
        }
        printf
("\nEncontrar numeros primos entre 2 e %d (inclusive).\nCalculando...", limit);
        pc = 0;                 /* Inicializa o contar de primos */
        for (n = 1; n <= limit; n = n + 2) {  /* testa sequencialmente se um
número é primo ou nao */
                    if (isprime(n)) {
                        pc++;
                }
        }
        printf("\nFinalizado.\nTotal de primos encontrados %d\n", pc);
        return 0;
}
```

2) Primos V2.0

Modifique o código do exercício 1 (a versão sequencial) para que utilize apenas comunicação coletiva.

3) Comunicação coletiva

O algoritmo abaixo lê um vetor de 1600 inteiros, e um valor inteiro X a ser procurado no vetor. O resultado da execução é a quantidade de ocorrências de X no vetor. Modifique o código para utilizar funções de comunicação coletiva MPI (bcast, gather, scatter) para realizar a contagem de ocorrências do valor lido.

Dica: Utilize redirecionamento de entrada para facilitar os testes.
Atenção: Considere que apenas o processo de rank 0 aloca console (i.e. Pode ler da entrada padrão).

```c
#include <stdio.h>
#include <stdlib.h>
#define SIZE 1600   /* tamanho do vetor */

int main ( int argc, char **argv ){
    int occurrences;    /* número de ocorrências do número alvo */
    int target;         /* número a ser buscado */
    int vec[SIZE];      /* vetor no qual o número será buscado */

    int i;              /* iterador */

    printf("Digite o vetor a ser analisado separando cada elemento em uma linha diferente: ");
    for(i=0; i<SIZE; i++){
        scanf("%d", &vec[i]);   /* leitura do vetor */
    }

    printf("\nDigite o valor a ser buscado: ");
    scanf("%d", &target);   /* leitura do número a ser buscado */

    occurrences = 0;
    for(i=0; i<SIZE; i++){      /* busca sequencial pelo número */
        if(vec[i] == target){   /* ocorrência encontrada */
            occurrences++;
        }
    }

    printf("\nFinalizado\n\nNumero de vezes em que %d aparece no vetor e: %d\n", target, occurrences);

    return 0;
}
```