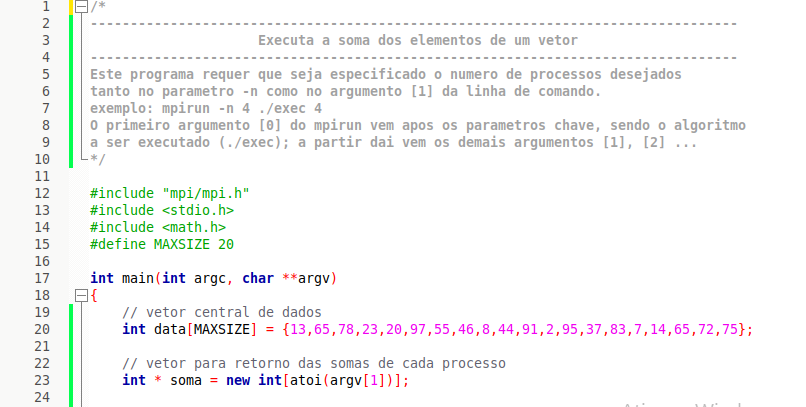
**MPI - COMUNICAÇÃO ENTRE PROCESSOS (continuação)**

Continuando o conteúdo que vínhamos estudando, vamos ver agora o MPI\_Scatter. É semelhante ao MPI\_Bcast, mas com a seguinte diferença - MPI\_Bcast envia o mesmo conteúdo a todos os processos e MPI\_Scatter separa as partes e envia a parte que cada um deve trabalhar.

|  |  |
| --- | --- |
| Figura 1 - MPI\_Bcast | Figura 2 - MPI\_Scatter |

Vamos ver o exemplo a seguir:



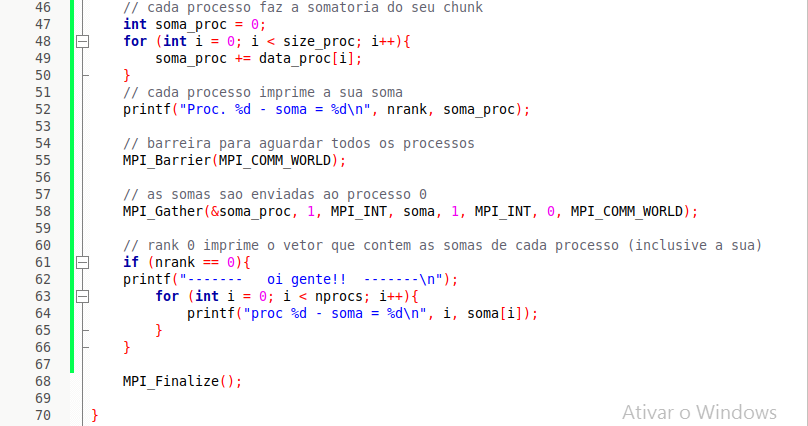
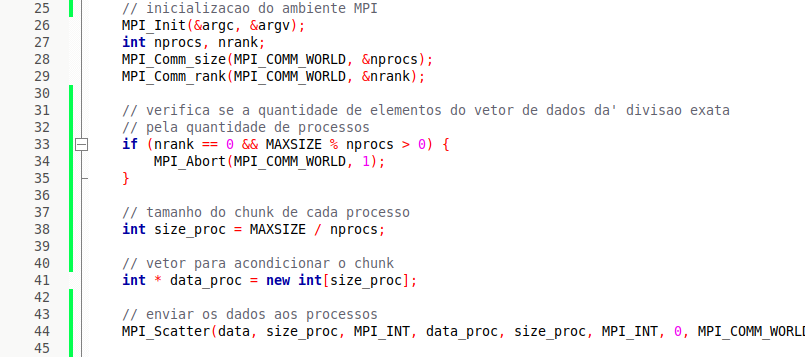


Figura 3

A saída deste programa é a seguinte:

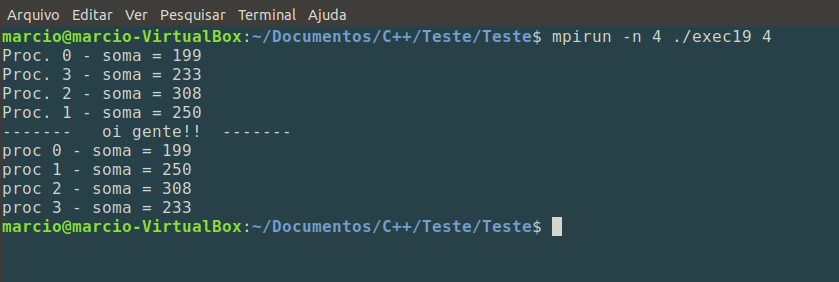
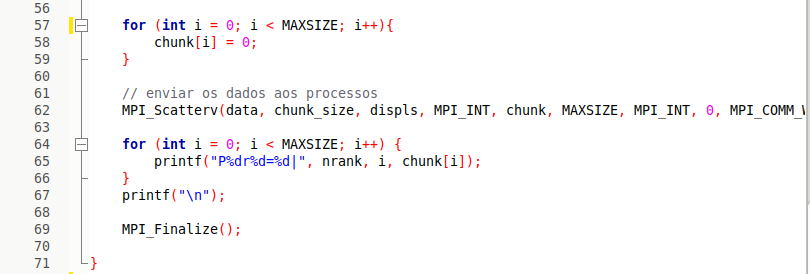
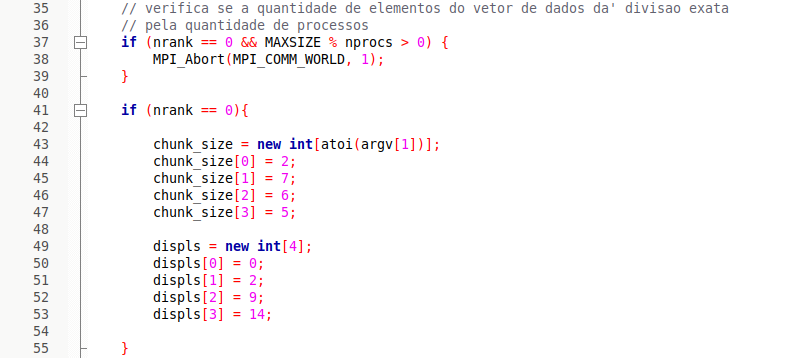
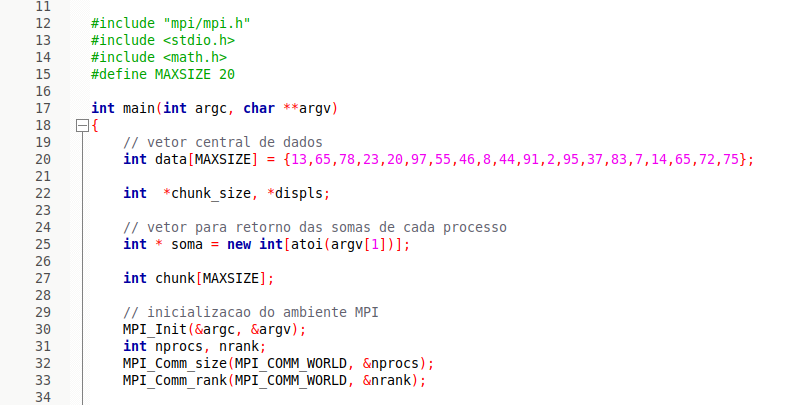


Figura 4

Vamos ver agora uma variação da MPI\_Scatter que é a MPI\_Scatterv. A ideia é a mesma, porém nesta última pode-se especificar o tamanho de cada chunk individualmente.



Este programa tem que ser executado como o anterior, informando, após o nome do executável, a quantidade de processos.

A variável (vetor) *chunk\_size* informa o tamanho do chunk de cada processo, uma vez que serão diferentes[[1]](#footnote-1). A variável (vetor) *displs* informa o início de cada *chunk* no vetor base.

Nas linhas 44 - 47 são especificados os tamanhos de cada *chunk*. O índice do vetor corresponde ao *rank* do processo.

Nas linhas 50 - 53 são especificados os inícios dos *chunks* de cada processo. Aqui é importante observar que os *chunks* são independentes e, tanto podem ter sobreposições de dados, como também "buracos". Ou seja, um chunk não tem relação nenhuma com os outros. Esta relação quem define é o programador e, portanto, deve estar coerente com as regras do negócio. No nosso exemplo é necessário tomar os chunks sequenciais, sem sobreposições e sem buracos.

Na linha 62 vem a instrução MPI\_Gatterv. Os quatro primeiros parâmetros são significativos apenas para o processo remetente, no caso o processo base (0). O quinto parâmetro "chunck" é o elemento que o processo destinatário recebe e deve trabalhar, que é o vetor com os seus dados. Nesse exemplo, foi reservado um espaço igual ao tamanho do vetor base para cada chunk, o que possibilita que cada processo receba o vetor base na íntegra (neste caso é melhor usar a MPI\_Bcast).

Na linha 64, cada processo imprime o seu *chunk*, A saída pode ser vista a seguir (Figura 5).

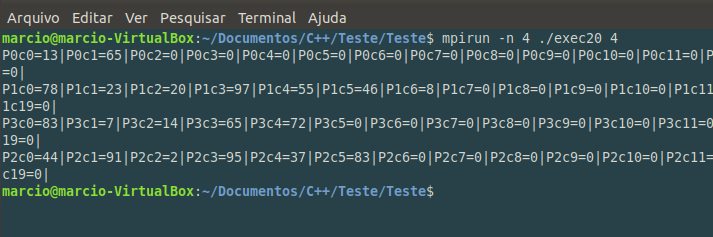


Figura 5

Essas linhas impressas vêm da instrução da linha 65.

onde:

* P = **P**rocesso
* X = rank do processo
* c = **c**hunk
* Y = índice no vetor do chunk
* N = valor a ser computado
* | = separador

Observe que na linha 57 cada chunk foi preenchido com 0, de forma que os valores significativos vão até antes do primeiro zero.

**Exercício:**

Pesquise a instrução MPI\_Gatterv e complete o programa para que seja obtida a somatória dos elementos do vetor.

1. Nos exemplos encontrados na internet, a divisão costuma ser feita de forma equitativa. Aqui vamos especificar individualmente para entender o mecanismo. [↑](#footnote-ref-1)