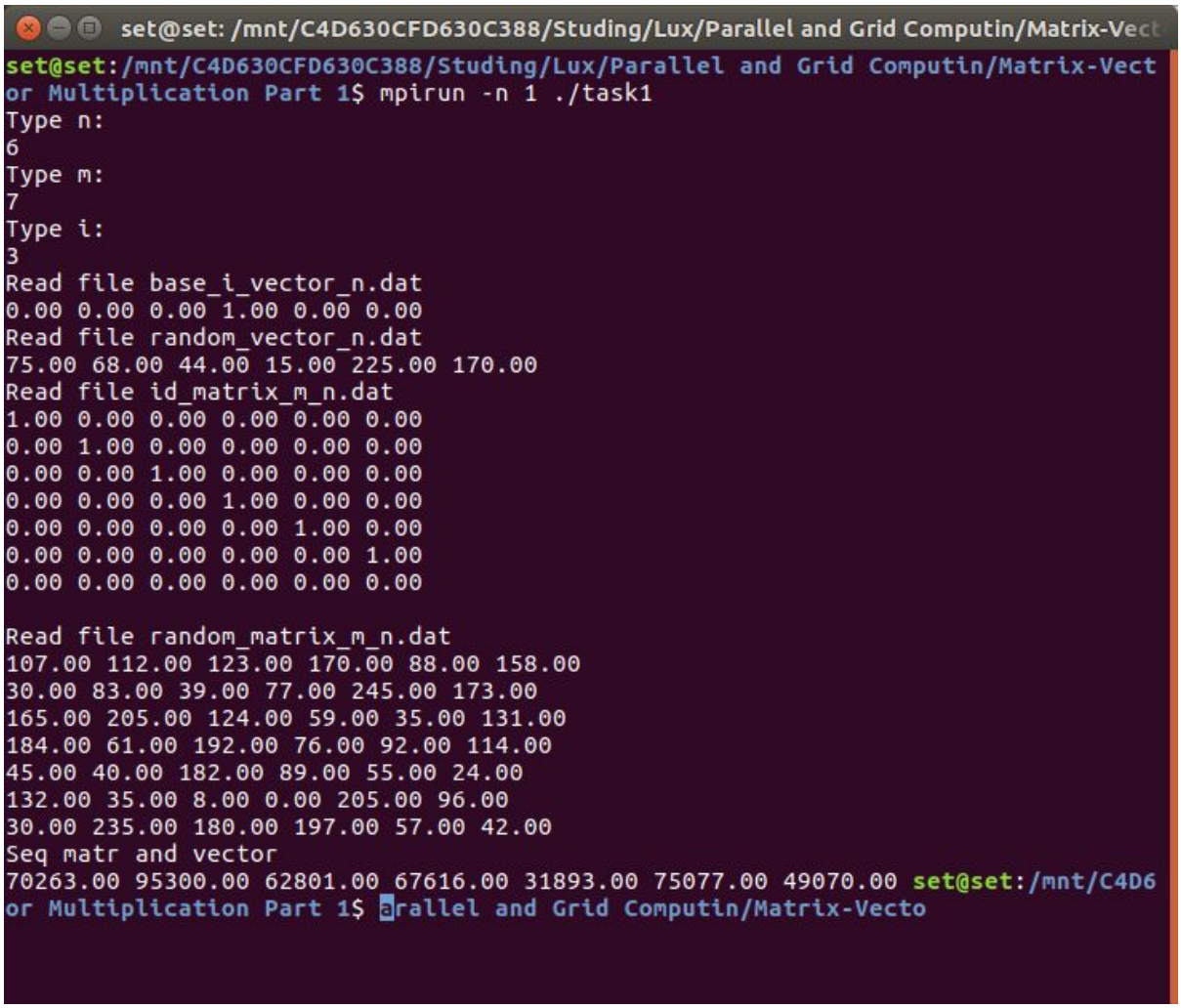
Exercise 1
Implementation of tasks 1-6 you can find in file task1.c
to execute program you need to type next commands in terminal
mpicc task1.c -o task1 - compile
mpirun -n 1 ./task1 - execute

Result

Task 7
The complexity of the sequential matrix-vector multiplication, in the general case is m*n The complexity of the sequential matrix-vector multiplication, when m = n is n^2



Exercise 2
Task1

Task2
To find number of elements of the last processor we need to subtract from k the number of elements of other processors(0..p-2). The number of elements of processor we can find from next formula $\lfloor(i+1) * \frac{k}{p}\rfloor - \lfloor i * \frac{k}{p}\rfloor$
k - ($\lfloor(0+1) * \frac{k}{p}\rfloor - \lfloor 0 * \frac{k}{p}\rfloor$) - ($\lfloor(1 + 1) * \frac{k}{p}\rfloor - \lfloor 1 * \frac{k}{p}\rfloor$) - ($\lfloor(2+1) * \frac{k}{p}\rfloor - \lfloor 2 * \frac{k}{p}\rfloor$) - ....- ($\lfloor(p-m+1) * \frac{k}{p}\rfloor - \lfloor(p-m) * \frac{k}{p}\rfloor$)- ....- ($\lfloor(p-2+1) * \frac{k}{p}\rfloor - \lfloor(p-2) * \frac{k}{p}\rfloor$)
We can see that formula can be simplified to
k - $\lfloor(p-2+1) * \frac{k}{p}\rfloor$ = $k - \lfloor(p-1) * \frac{k}{p}\rfloor$= We need to take upper round value because the are no more processors to process remaining elements => $\lceil k - (p-1) * \frac{k}{p}\rceil = \lceil k - k + \frac{k}{p}\rceil = \lceil\frac{k}{p}\rceil$

Task3
Implementation of task you can find in file task1.c

```
void create_mixed_count_disp_arrays(int p, size_t k, int **count, int **disp) {
   (*count)[0] = BLOCK_SIZE(0, p, k);
   (*disp)[0] = 0;

   for (int i = 1; i < p; ++i) {
         (*disp)[i] = (*disp)[i-1] + (*count)[i-1];
         (*count)[i] = BLOCK_SIZE(i, p, k);
   }
}
```

Taks4
Implementation of task you can find in file task1.c

```
void create_count_disp_arrays(int id, int p, size_t k, int **count, int **disp) {
   int size = BLOCK_SIZE(id, p, k);
   (*disp)[i] = 0;
   (*count)[i] = size;
   for (int i = 1; i < p; ++i) {
         (*disp)[i] = (*disp)[i-1] + (*count)[i-1];
         (*count)[i] = size;
   }
}
```

Exercise 3

The second approach allows to decrease number of communication between processors. This allows to increase speed of calculation (The bottlenecks in parallel algorithms are often connected with communication delay between processors)

Also two parallel algorithms of matrix multiplication required full vector and row of matrix to find complete element of resulting matrix (Rowwise block-striped decomposition and Columnwise block-striped)