The aim of the laboratory task was to configure and run a memory management simulator. We had to edit two files "commands" and "memory.conf" by setting a proper configuration. We observed the results in a file named "traceline" and on a graphical simulator which allowed to watched memory mapping in real time.

We had to map any 8 pages of physical memory to the first 8 pages of virtual memory such a solution gives as more realistic simulation since in real world physical memory addresses are not mapped in the same addresses as in virtual memory. Than we had read from one virtual memory address on each of the 64 virtual pages, and predict which virtual memory addresses cause page faults and which page replacement algorithm is being used.

First of all it turned out to be impossible to map only 8 and read 64 pages, because the number of pages was always equal to the sum of physical and virtual pages mapped (f.e. for the maximum number of pages set to 32, the simulator always mapped 16 physical to 16 virtual pages).

In command file we set addresses of each virtual page that we will be read. Since I set the size of page to 128 address of each page will be (128*i + 1) where i = 0,1,2...63.

As we set number of virtual memory pages to 64 in simulator number of physical pages was 32 which made sense since if this number was equal we would not observe any page fault.

Page fault occurs when want to map our virtual memory address to physical memory address but there is no more space physical memory. In this situation we have to swap some memory address to our external memory to make space for currently need it memory. In tracefile we can see our observations.

The question is which physical memory to choose, and this where algorithms comes to help.

In our case the algorithm used for pages replacement was FIFO algorithm, which means "First-In First-Out".It replace the frame page which has been in memory for the longest time. In other words the page frames are kept in a queue and the frame that was used lately is moved to the tail and in the next replacement the next page from the queue is swapped.

This algorithm is not very efficient because it doesn't make a difference between pages frequently used and not used at all, therefore the better algorithm for page replacement would be for example a priority queue or LRU (least recently used) algorithm.