Exercice 1

We consider a virtual memory management system which relies on the following structures:

mem_addr_t PageTable[NB_PAGE];
disk_addr_t SwapTable[NB_PAGE];

mem_addr_t is an address in memory and disk_addr_t is an address on disk.
We assume that a page is either in memory or in the swap (on disk), else there's an addressing error.
PageTable is the page table of the virtual address space we have to manage (we manage only one virtual address space). SwapTable gives for each virtual page its location in the swap if the page was swapped out.
You have to implement the *void page_fault(int npage)* procedure which is invoked by the hardware when the *npage* entry in the page table is null.
To implement this procedure, you can use the following procedures (i.e. they are **available**, you **don't have to implement them**):

- **void memory_error()**: have to be called when a memory error is detected.
- **mem_addr_t memory_alloc()**: allocate a page in main memory. Return the address of the page in memory, null if memory is full.
- **int lru_select()**: select a page for replacement. Return the page number in PageTable. Be carefull, it does not return a page address in memory. The address of the page in memory (which is selected for replacement) has to be fetched from PageTable.
- **disk_addr_t swap_alloc()**: allocate a page in the swap. Return the address of this page on disk.
- **void swap_free(disk_addr_t disk_page)**: free a page from the swap.
- **void load_page(mem_addr_t mem_page, disk_addr_t disk_page)**: load a page from the swap into a page in memory.
- **void store_page(mem_addr_t mem_page, disk_addr_t disk_page)**: store a page from memory into a page in the swap.

Exercice 2

We consider a page replacement system for virtual memory management in an operating system.
We assume that the physical memory is composed of 4 pages.
The first line of the table below gives the numbers of the virtual pages which are consecutively accessed.
The first column of the table gives the physical page numbers associated with each line.
Physical pages are initally empty.
Each column corresponds to an access to a virtual page and gives the state of physical memory after the access.
You have to give such a table in three cases:

- when a FIFO strategy is used
- when a LRU (Least Recently Used) page replacement strategy is ued
- when the optimal strategy is used

For each case, indicate the number of page faults.

Access to page	1	2	3	4	1	2	5	1	2	3	4	5	2	3
Page 1														
Page 2														
Page 3														
Page 4														