EOPSY – Task 4
Memory Management
https://github.com/TimeRamen/EOPSY/tree/master/LAB4

Taimur Rahman
289009

Initial tracefile – `tracefile`

```
READ 4 ... okay
READ 13 ... okay
WRITE cc32 ... okay
READ 4000 ... okay
READ 4000 ... okay
WRITE 6001 ... okay
WRITE f34aa ... page fault
```

Intro

At the beginning, I ran the files out of the box, and the tracefile looked like that. I later allowed it to run further by changing the commands file. It is impossible to map only 8 pages and read from one virtual memory address on each of the 64 pages since the number of pages is the sum of physical and virtual pages mapped together. "numpages" was set to 64 on memory.conf and will map 32 physical pages to 32 virtual ones.

Memory configuration – `memory.conf`

Configurations

```
// memset  virt page #  physical page #  R (read from)  M (modified)
// inMemTime (ns) lastTouchTime (ns)
memset 0 0 0 0 0 0
memset 1 1 0 0 0 0
memset 2 2 0 0 0 0
memset 3 3 0 0 0 0
memset 4 4 0 0 0 0
memset 5 5 0 0 0 0
memset 6 6 0 0 0 0
memset 7 7 0 0 0 0


// enable_logging 'true' or 'false'
// When true specify a log_file or leave blank for stdout
enable_logging true

// log_file <FILENAME>
// Where <FILENAME> is the name of the file you want output
// to be print to.
log_file tracefile

// page size, defaults to 2^14 and cannot be greater than 2^26
// pagesize <single page size (base 10)> or <'power' num (base 2)>
pagesize 16384

// addressradix sets the radix in which numerical values are displayed
// 2 is the default value
// addressradix <radix>
addressradix 16
```

```
// numpages sets the number of pages (physical and virtual)
// 64 is the default value
// numpages must be at least 2 and no more than 64
// numpages <num>
numpages 64
```

Commands – `commands`

Configurations

```
// Enter READ/WRITE commands into this file
// READ <OPTIONAL number type: bin/hex/oct> <virtual memory address or random>
// WRITE <OPTIONAL number type: bin/hex/oct> <virtual memory address or random>

READ 16000
READ 32000
READ 48000
READ 64000
READ 80000
READ 96000
READ 112000
READ 128000
READ 144000
READ 160000
READ 176000
READ 192000
READ 208000
READ 224000
READ 240000
READ 256000
READ 272000
READ 288000
READ 304000
READ 320000
READ 336000
READ 352000
READ 368000
READ 384000
READ 400000
READ 416000
READ 432000
READ 448000
READ 464000
READ 480000
READ 496000
READ 512000
READ 528000
READ 544000
READ 560000
READ 576000
READ 592000
READ 608000
READ 624000
READ 640000
READ 656000
READ 672000
READ 688000
READ 704000
READ 720000
READ 736000
READ 752000
```

```
 READ 768000
 READ 784000
 READ 800000
 READ 816000
 READ 832000
 READ 848000
 READ 864000
 READ 880000
 READ 896000
 READ 912000
 READ 928000
 READ 944000
 READ 960000
 READ 976000
 READ 992000
 READ 1008000
 READ 1024000
```

Conclusion

Page faults happen after 31 pages, because from 32 onwards pages are mapped into virtual address spaces and not loaded into physical memory. The virtual pages are still allocated sequentially in the physical one.

The algorithm used is FIFO algorithm (first in first out). FIFO is a page replacement algorithm that decides which page has to be replaced when new one is made and the algorithm keeps track of all pages in memory in a queue. The oldest in front of the queue gets removed and the new page can go in. We see this happen in our tracefile.