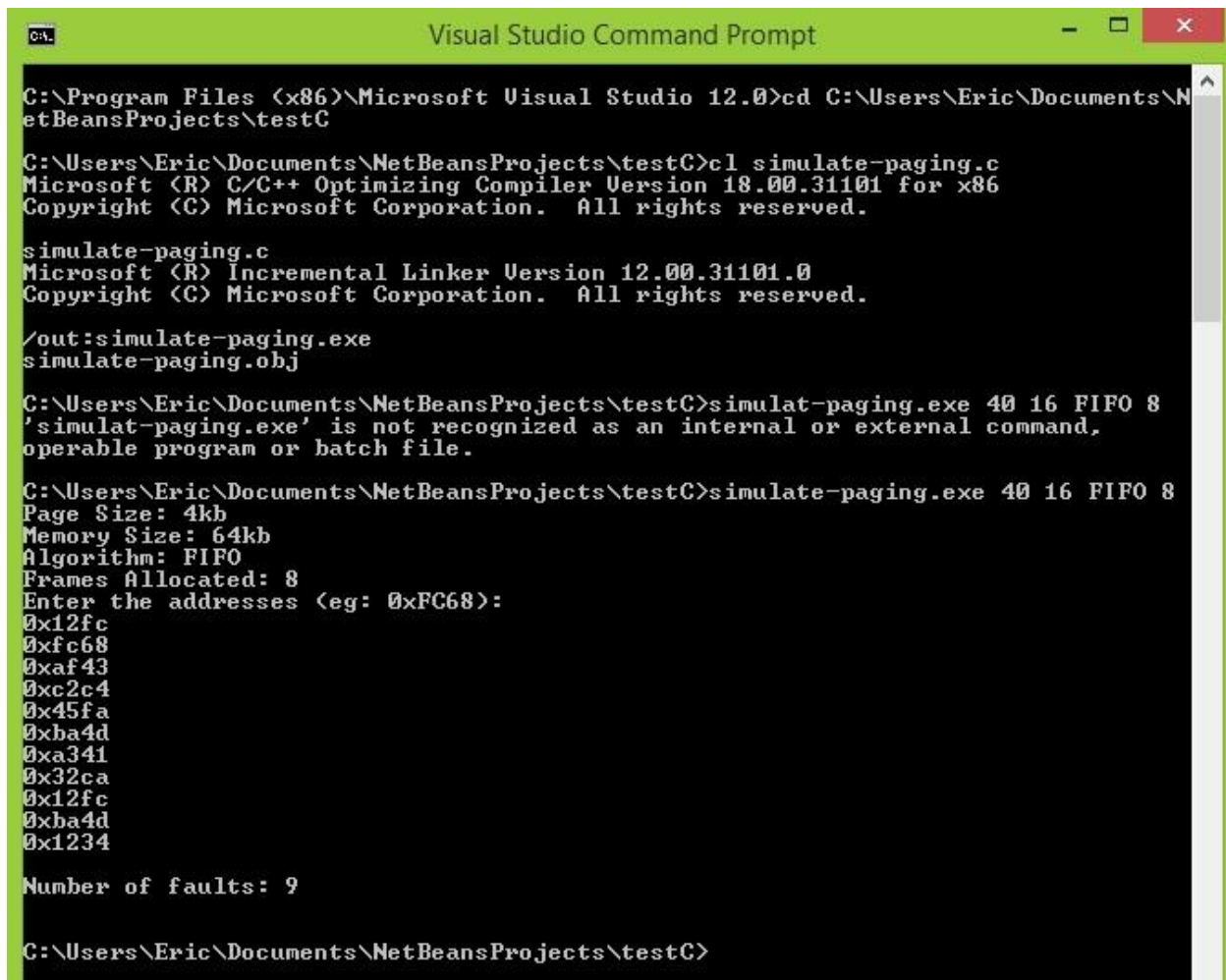


## Homework #6

For this assignment we were supposed to write a program that simulates the FIFO and LRU replacement algorithms. As indicated in the description we were supposed to accept four command line arguments that specify the page size, logical memory size, page replacement algorithm, and the number of frame allocated. This was a fairly straightforward program to right and didn't involve s much as the assignments before.

-First I compiled my program and tested with 8 frames and a couple of memory references.



```
C:\Program Files (x86)\Microsoft Visual Studio 12.0>cd C:\Users\Eric\Documents\NetBeansProjects\testC

C:\Users\Eric\Documents\NetBeansProjects\testC>cl simulate-paging.c
Microsoft (R) C/C++ Optimizing Compiler Version 18.00.31101 for x86
Copyright (C) Microsoft Corporation. All rights reserved.

simulate-paging.c
Microsoft (R) Incremental Linker Version 12.00.31101.0
Copyright (C) Microsoft Corporation. All rights reserved.

/out:simulate-paging.exe
simulate-paging.obj

C:\Users\Eric\Documents\NetBeansProjects\testC>simulat-paging.exe 40 16 FIFO 8
'simulat-paging.exe' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\Eric\Documents\NetBeansProjects\testC>simulate-paging.exe 40 16 FIFO 8
Page Size: 4kb
Memory Size: 64kb
Algorithm: FIFO
Frames Allocated: 8
Enter the addresses (eg: 0xFC68):
0x12fc
0xfc68
0xaf43
0xc2c4
0x45fa
0xba4d
0xa341
0x32ca
0x12fc
0xba4d
0x1234

Number of faults: 9

C:\Users\Eric\Documents\NetBeansProjects\testC>
```

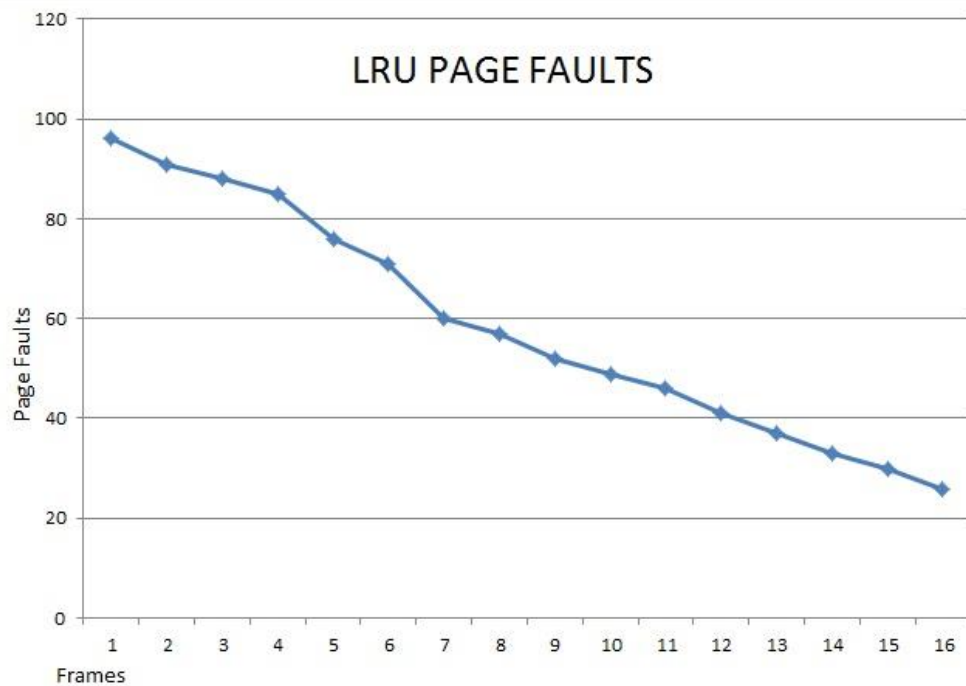
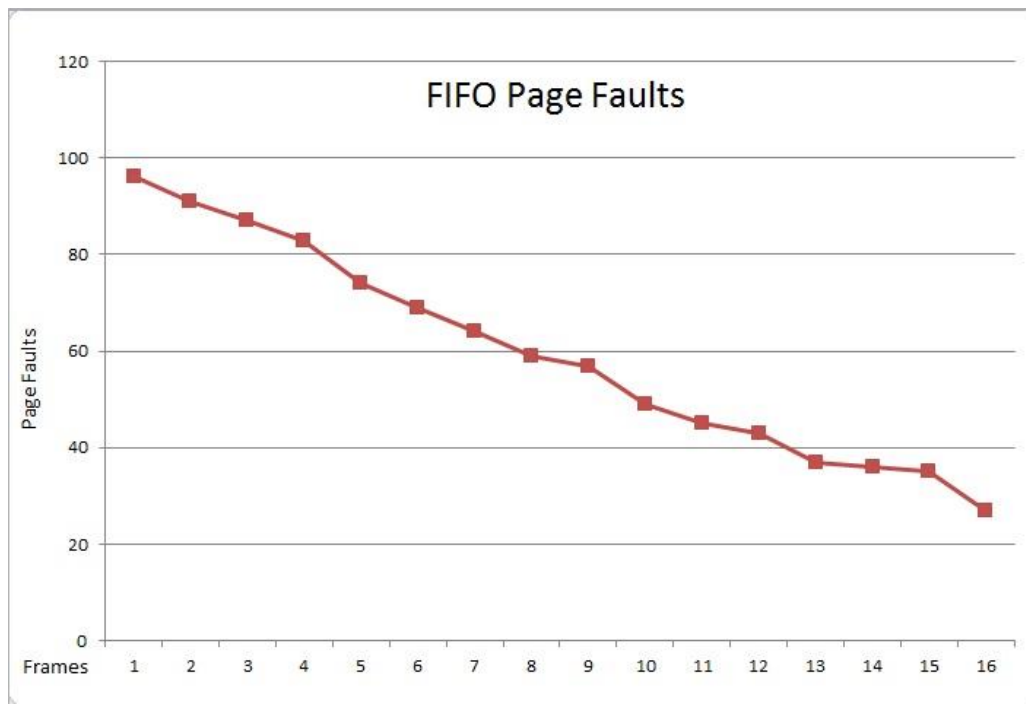
As you can see the program reads lines in hexadecimal representing the virtual address. It keeps reading the memory references until there's a blank line. It then prints out the number of faults.

-I then proceeded to test both the FIFO and LRU page-replacement algorithms with 100 random memory references using a page size of 4KB and a total logical memory size of 64KB, with each of the number of frames allocated ranging from 1 to 16. Here are the results:

### Page-replacement results:

Frame Alloc	FIFO	LRU
1	96	96
2	91	91
3	87	88
4	83	85
5	74	76
6	69	71
7	64	60
8	59	57
9	57	52
10	49	49
11	45	46
12	43	41
13	37	37
14	36	33
15	35	30
16	27	26

## Graphs



As you can see the results are very similar when put into a graph. The LRU algorithm had less page faults in some particular frames such as 12, 14, 15 but nothing exaggerated. Belady's anomaly does not occur during these test because the number of page faults does not go up as number of frames does, it goes down.