# Lab 3 Output analysis

Least Recently used:

| File No. | Page Hits | Page Faults | First Page Load | Physical Memory pages left | Swap Space pages left |
| --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 13 | 0 | 0 |
| 2 | 2 | 14 | 24 | 5 | 0 |
| 3 | 9 | 13 | 27 | 2 | 0 |
| 4 | 14 | 20 | 31 | 8 | 11 |
| 5 | 16 | 37 | 63 | 10 | 11 |
| 6 | 203 | 32 | 56 | 3 | 2 |

Random:

| File No. | Page Hits | Page Faults | First Page Load | Physical Memory pages left | Swap Space pages left |
| --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 13 | 0 | 0 |
| 2 | 2 | 11 | 24 | 4 | 1 |
| 3 | 9 | 14 | 27 | 2 | 0 |
| 4 | 14 | 20 | 31 | 9 | 10 |
| 5 | 16 | 33 | 63 | 10 | 11 |
| 6 | 203 | 31 | 56 | 3 | 2 |

### Analysis

Starting with the page hits both the least recently used and Random had the same values. As for their page faults, for the smaller data files, 1 - 3. My results between the two algorithms were very similar with random having fewer page faults for file 2 and more for file 3. The case was also the same for the larger files 3 - 6.

Random had to perform fewer swaps from the physical memory to the main memory. It is also not evident but from the values of the Physical memory and swap space memory, we can also see that the Least recently used manages the distribution of pages in memory better and more efficiently compared to Random.

One thing to note and a red flag I noticed was my program was not generating any insufficient memory errors generated, even though I was keeping track of the total used space. But at the same time, no array out-of-bounds error was generated.

In conclusion, both the least recently used and Random page replacement algorithms had a similar performance for smaller data files, but Random had fewer page faults for some larger data files. However, the distribution of pages in memory was more efficient with the least recently used algorithm.