PROJECT REPORT

INTRODUCTION:

In this project we will be show how different page replacement algorithm works and will show a general comparison between them. The algorithms which we will be comparing are as under:

* First In First Out. (FIFO)
* Least Recently Used. (LRU)
* Second Chance Algorithm. (SCA)

DESCRIPTION:

FIRST IN FIRST OUT:

The simplest page-replacement algorithm is a FIFO algorithm. The first-in, first-out (FIFO) page replacement algorithm is a low-overhead algorithm that requires little bookkeeping on the part of the [operating system](https://en.wikipedia.org/wiki/Operating_system). The idea is obvious from the name – the operating system keeps track of all the pages in memory in a queue, with the most recent arrival at the back, and the oldest arrival in front. When a page needs to be replaced, the page at the front of the queue (the oldest page) is selected. While FIFO is cheap and intuitive, it performs poorly in practical application. Thus, it is rarely used in its unmodified form. This algorithm experiences [Bélády's anomaly](https://en.wikipedia.org/wiki/B%C3%A9l%C3%A1dy%27s_anomaly" \o "Bélády's anomaly). In simple words, on a page fault, the frame that has been in memory the longest is replaced.

LEAST RECENTLY USED:

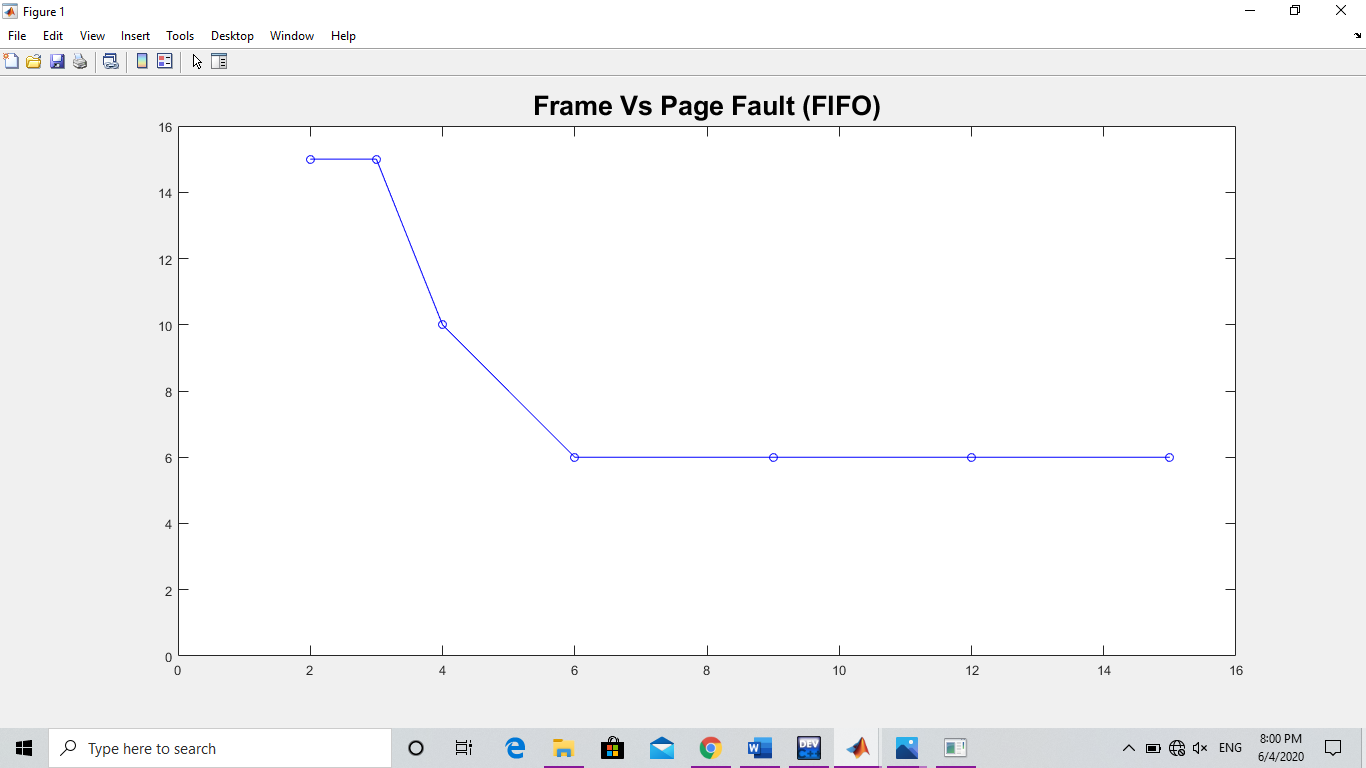
The least recently used (LRU) page replacement algorithm, though similar in name to NRU, differs in the fact that LRU keeps track of page usage over a short period of time, while NRU just looks at the usage in the last clock interval. LRU works on the idea that pages that have been most heavily used in the past few instructions are most likely to be used heavily in the next few instructions too. While LRU can provide near-optimal performance in theory (almost as good as [adaptive replacement cache](https://en.wikipedia.org/wiki/Adaptive_replacement_cache)), it is rather expensive to implement in practice. There are a few implementation methods for this algorithm that try to reduce the cost yet keep as much of the performance as possible.

SECOND CHANCE ALGORITHM:

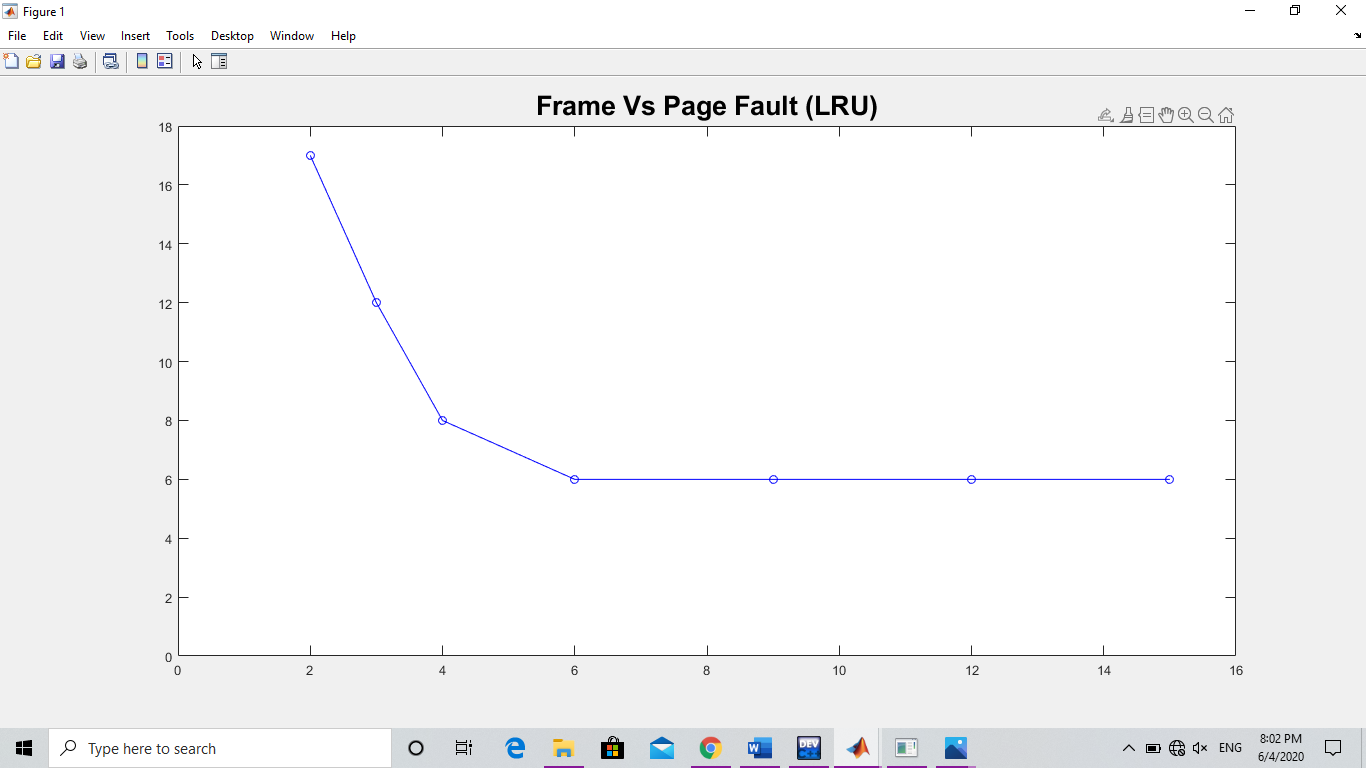
A modified form of the FIFO page replacement algorithm, known as the Second-chance page replacement algorithm, fares relatively better than FIFO at little cost for the improvement. It works by looking at the front of the queue as FIFO does, but instead of immediately paging out that page, it checks to see if its referenced bit is set. If it is not set, the page is swapped out. Otherwise, the referenced bit is cleared, the page is inserted at the back of the queue (as if it were a new page) and this process is repeated. This can also be thought of as a circular queue. If all the pages have their referenced bit set, on the second encounter of the first page in the list, that page will be swapped out, as it now has its referenced bit cleared. If all the pages have their reference bit cleared, then second chance algorithm degenerates into pure FIFO.

GRAPHS:

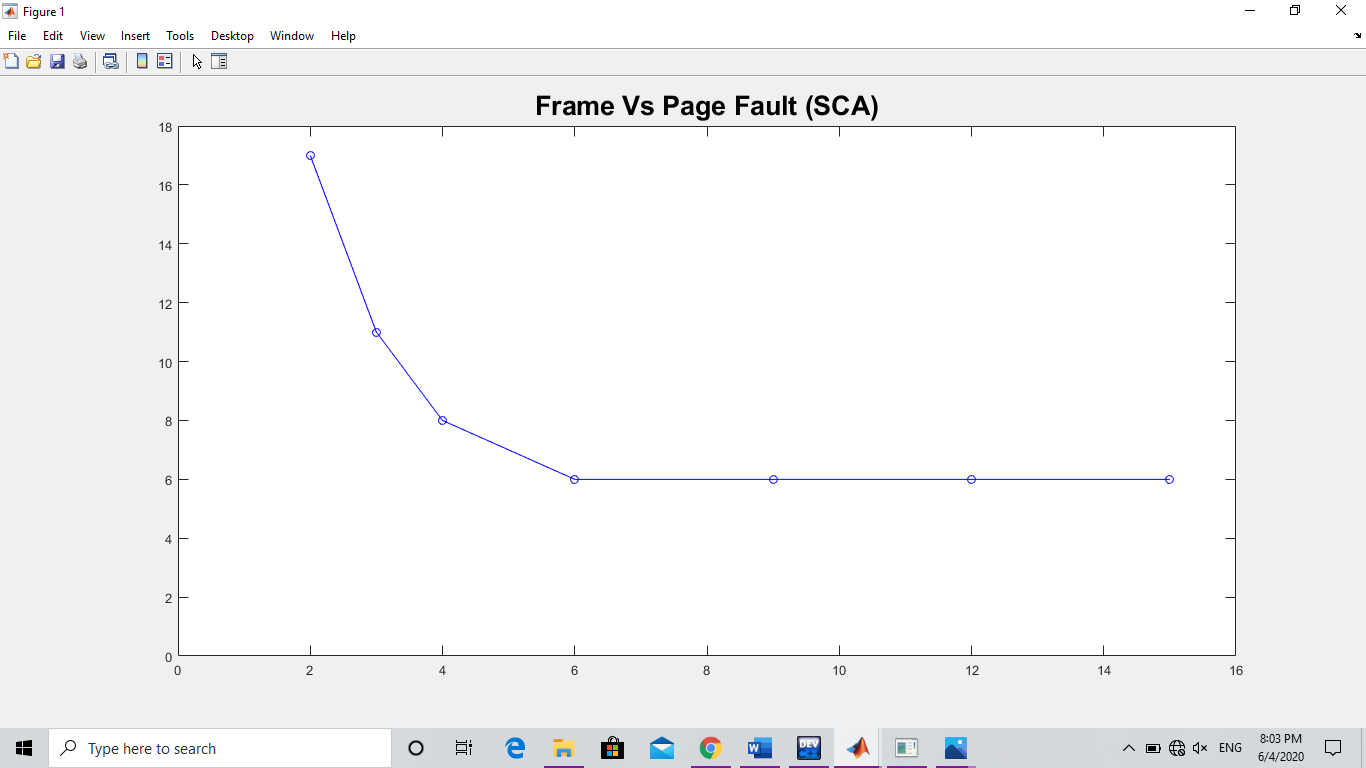
FIFO:



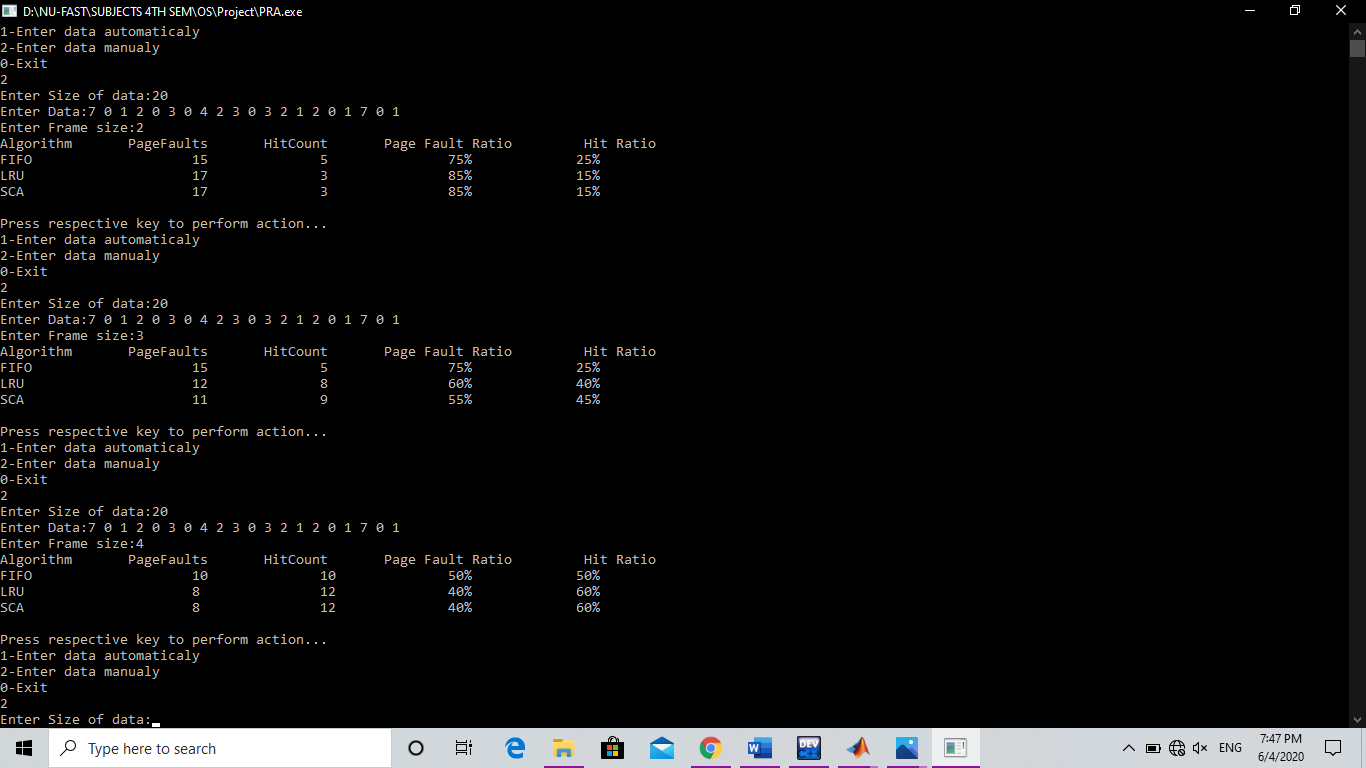
LRU:

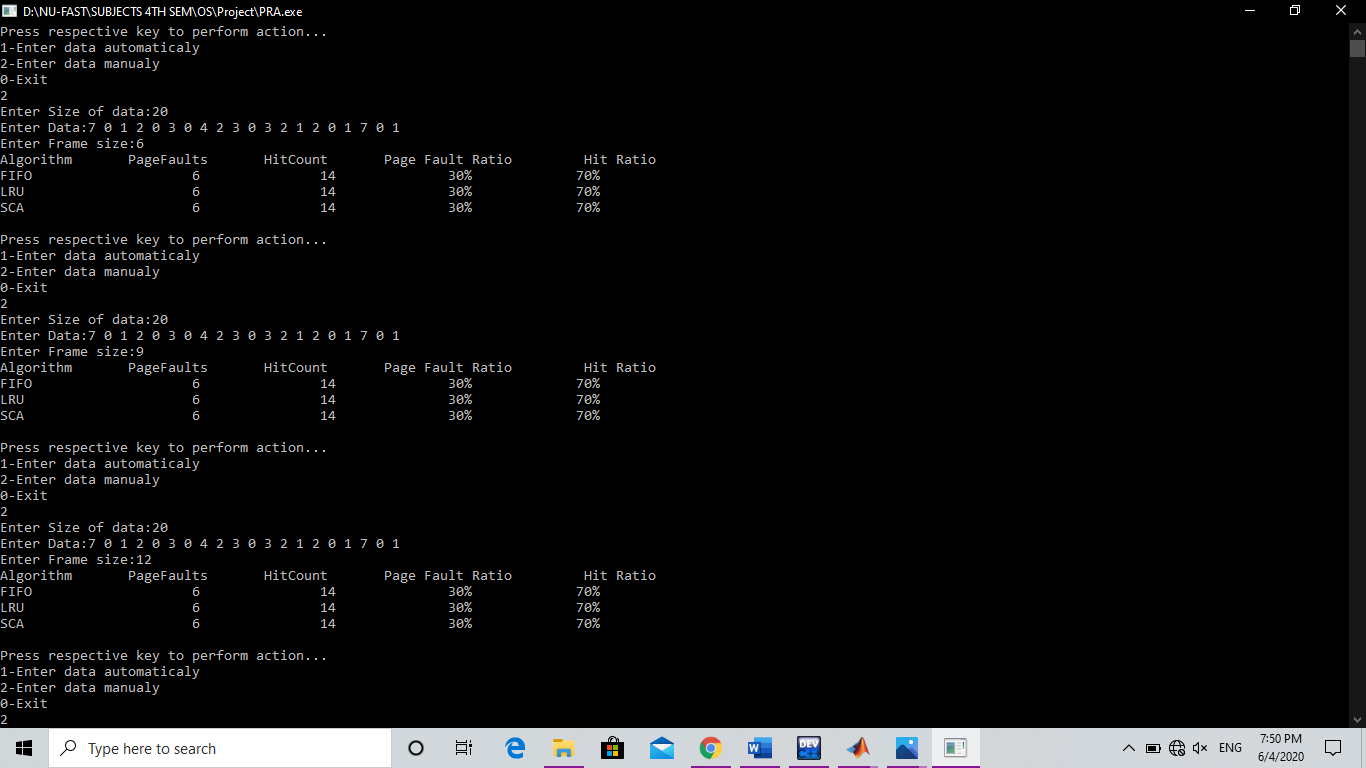


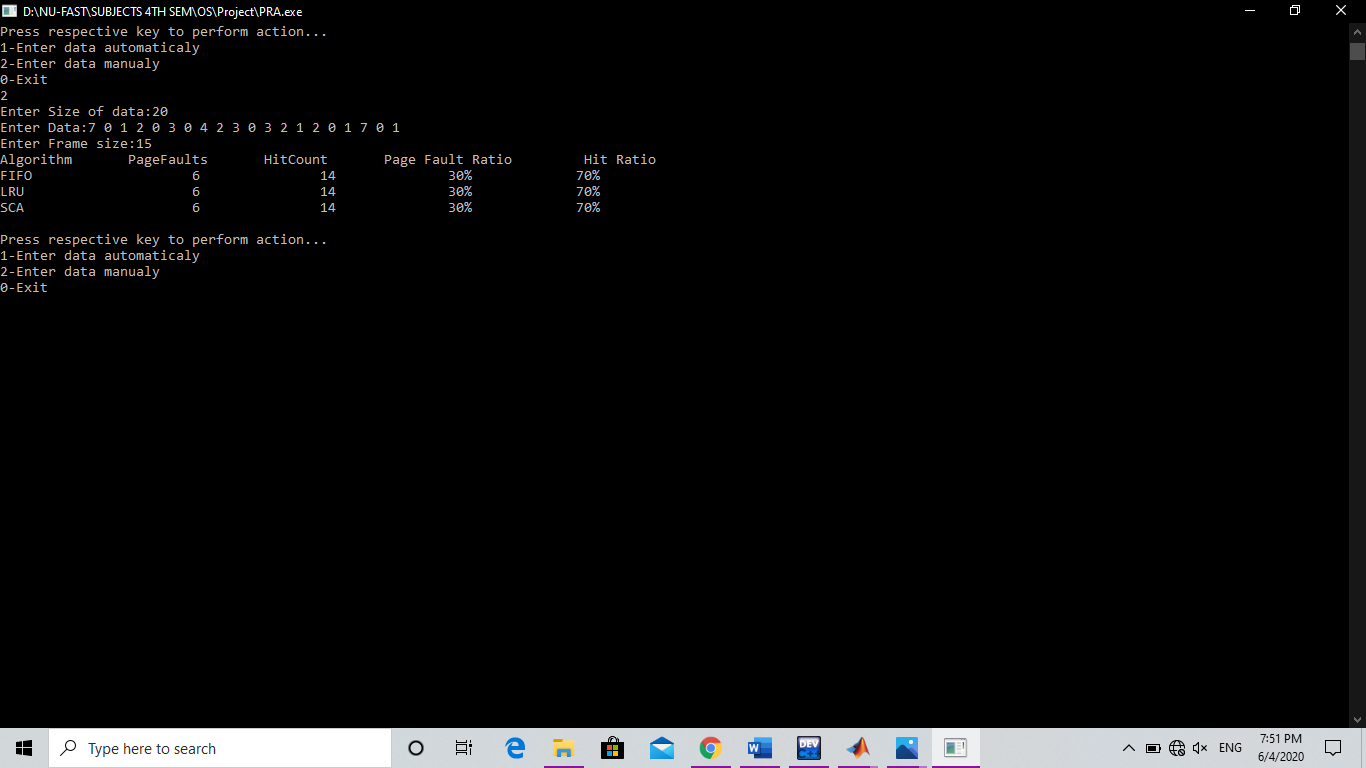
SCA:



SCREENSHOTS:







REFERENCES:

* <https://www.geeksforgeeks.org/page-replacement-algorithms-in-operating-systems/>
* <https://en.wikipedia.org/wiki/Page_replacement_algorithm#:~:text=In%20a%20computer%20operating%20system,memory%20needs%20to%20be%20allocated.>