PDPU

PANDIT DEENDAYAL PETROLEUM UNIVERSITY

Operating System Lab Project Report

Team 34: Page Replacement Algorithm with Blady's Anomaly GUI

Leader: Ronak Makwana (18BCP093)
Team Members: Parv Chauhan (18BCP077)
Prachi Doshi (18BCP079)
Pratvi Shah (18BCP083)

Submission Date: 11th December, 2020
Submitted to: Professor Chintan Patel

Table of Contents

Introduction:

Page Replacements algorithm is one of the most important concepts of the Operating System. The major drawback of any operating system is its speed and memory allocation process. The number of memory calls done to access frequently used pages, is tiresome and affects the speed, accuracy and durability of the computer hardware and the other software present in the system.

To overcome this issue, operating system came up with the concept of virtual memory management. Using the concept of storing the pages in the virtual memory, page replacement algorithms is introduced as a solution to paging issues faced in the operating systems.

In this project, we are going to focus on the three most important algorithms of the Page Replacement Topic. We are going to show how the algorithms are implemented, we are going to provide you with algorithm calculator, and a comparison graph simulator, that helps you to compare between the all the algorithm for the similar kind of input.

Theoretical Explanation:

In this project, we are going to explain three basic algorithms of the Page Replacement Algorithm, which are:

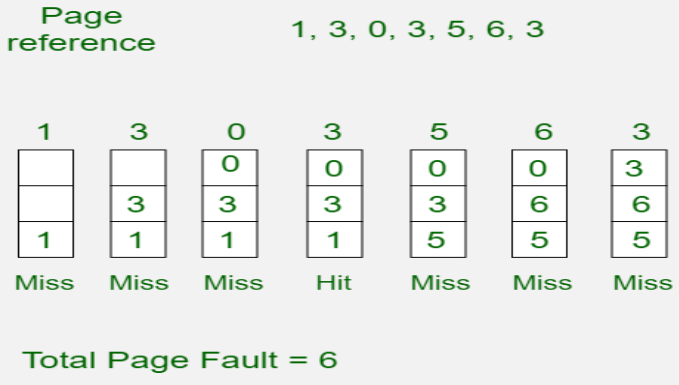
- FIFO(First In First Out) Page Replacement Algorithm
- LRU(Least Recently Used) Page Replacement Algorithm
- OPR (Optimal Page Replacement) Page Replacement Algorithm.

01 FIFO (First in First Out) Page Replacement Algorithm:

First In First out Page Replacement Algorithm, is the simplest algorithm. Page Replacement Algorithm follows a simple queue of pages entering the page frame. The oldest page is at the head of the queues. Thus, when the need arises the page at the head of the queue is removed and replaced with another set of frame.

FIFO Page Replacement Algorithms suffers from Baldy's Anomaly that says that when the number of page frames increase, the page fault should decrease but if the algorithm suffers from Blady's Anomaly, and then the page fault will increase.

Example-1: Consider page reference string 1, 3, 0, 3, 5, 6 with 3 page frames. Find number of page faults.



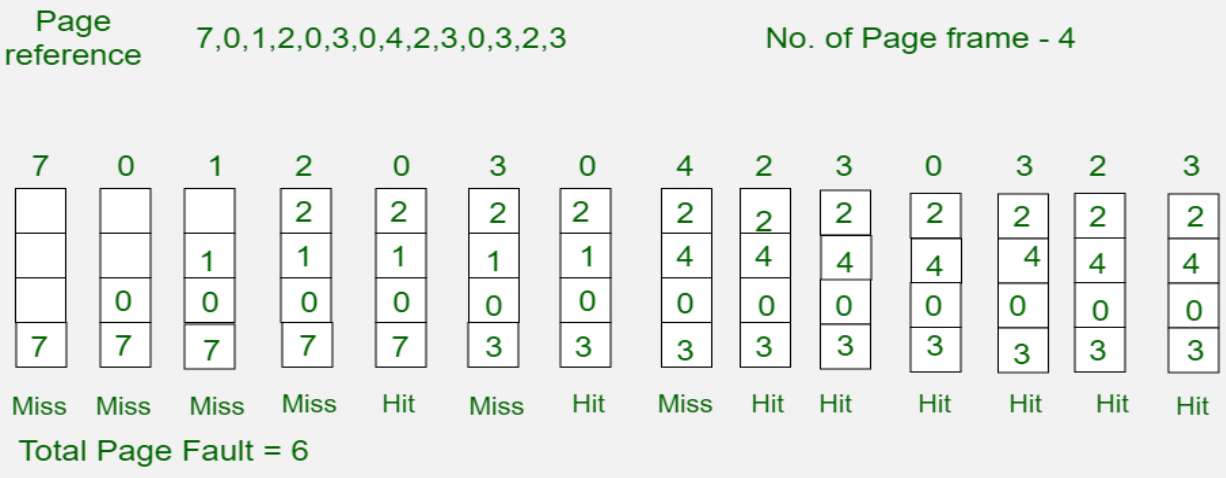
02 LRU (Least Recently Used) Page Replacement Algorithm:

Least Recently Used page replacement algorithm is the second most optimal algorithm and can be implemented easily. According to the LRU algorithm, when the need arises to replace the page in the frame stack, the least recently referred page is replaced.

This assumption made is taking into consideration, since the page is not referred frequently until date it will not be referenced in the future too. Thus, whenever the need arises to replace the page a pointer starts from the head of the page replacement queue and runs until the page to be replaced with and checks which page present currently in frame is least recently used. After checking the frequency it suggests the replacement.

LRU Page Replacement does not suffer from Blady's Anomaly.

Example - : Consider the page reference string 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2 with 4 page frames. Find number of page faults.



03 OPR (Optimal Page Replacement) Algorithm:

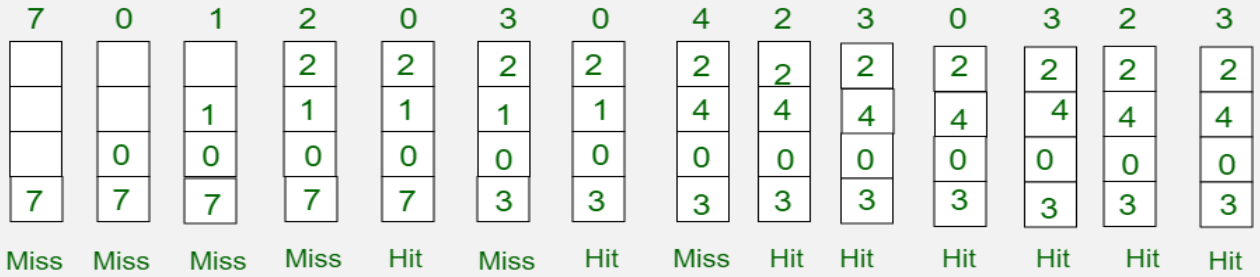
Optimal Page Replacement is the most efficient algorithm, but cannot be implemented practically in an operating system. As, in the operating system we do not have finite number of page references and hence not possible in practice as the operating system cannot know future requests.

In this algorithm, when the need arises to replace the page, the page are replaced which would not be used/needed for the longest duration of time in future. This strategy is applicable to finite number of known pages, not infinite number of unknown pages that might not be referenced in future.

Optimal Page Replacement Algorithm does not suffer from Blady's Anomaly. It set a benchmark so that the other replacement algorithm can be analyzed against it.

Example-3: Consider the page references 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, with 4-page frame. Find number of page fault.

Page reference 7,0,1,2,0,3,0,4,2,3,0,3,2,3 No. of Page frame - 4



Total Page Fault = 6

GUI (Graphical User Interface):

To implement all the three algorithms and explain the working of the algorithm visually our team has created a GUI. The GUI is a Desktop App developed using Java Programming Language. Whole code have been developed on the Eclipse IDE Java Editor. All the methods of downloading the IDE and external jar files, also setting up the IDE and jar files and running the code has been mentioned in the README file attached with this file. The README file will guide the student to set up the editor and external jar file and help them to run our code snippet effortlessly.

To develop the Desktop App, we have used the following java packages:

1. **Javax Swing Package** for creating and viewing desktop app
2. **Java AWT Package** for drawing and setting dimension and external layout in the Desktop App
3. **Javax Windows Builder** method to create the visuals in the GUI
4. **Java language Exception package** to handle all the possible exceptions in the code.
5. **Java language Thread class** to change between multiple JFrames smoothly and effortlessly.
6. Used **JFreeChart** library**,** which is an external jar file to show graph and comparisons graph.
7. Used **vlcj** library, which is an external jar file to attach the video animation and the voice over created by the team.

All the information regarding the installation, the version issues and setting up the library to, successfully run the code is mentioned in the README file.

Now, attaching the snapshots of GUI and briefly explaining all the components of our GUI.

01 Homepage of the Desktop App

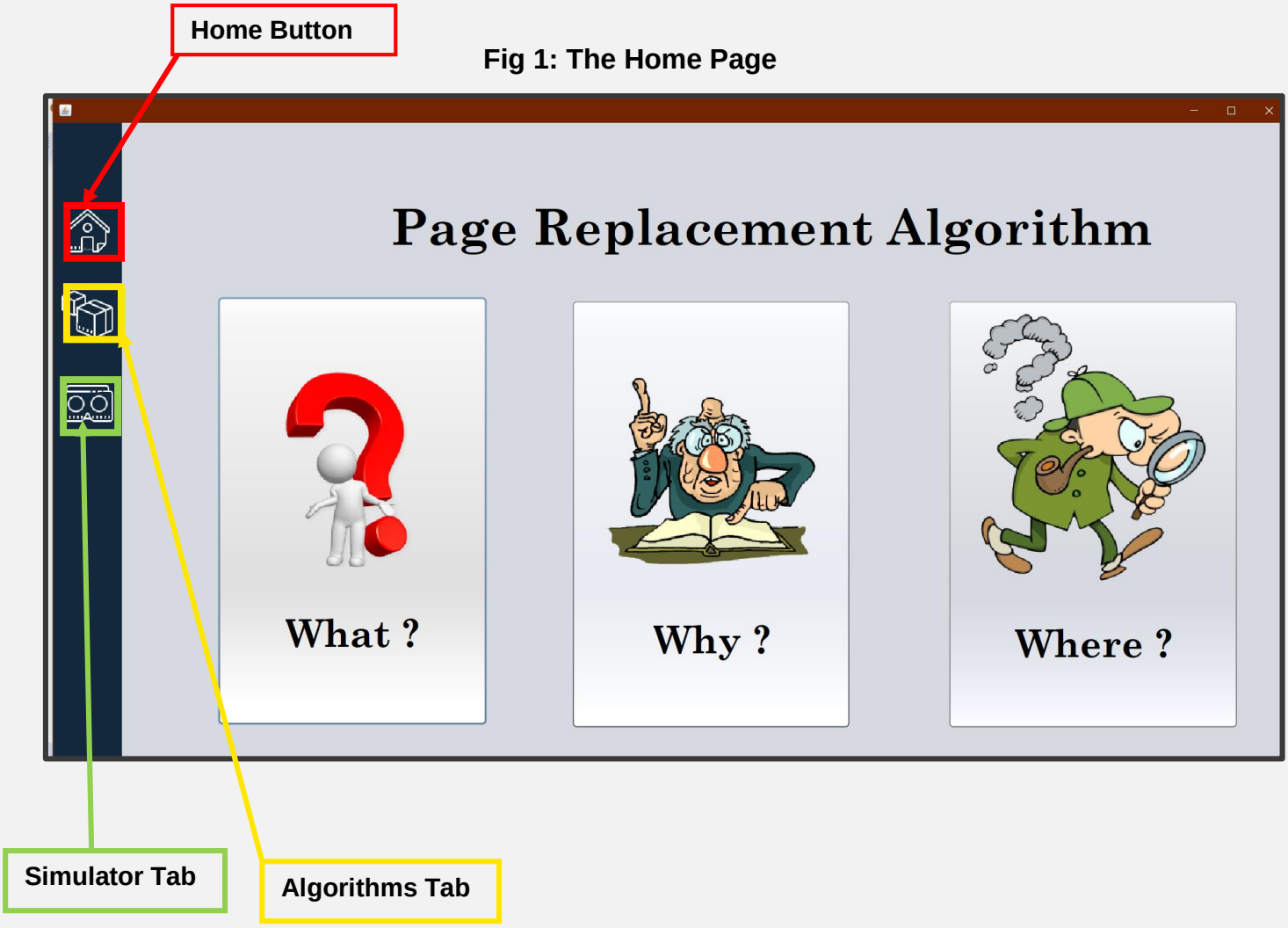


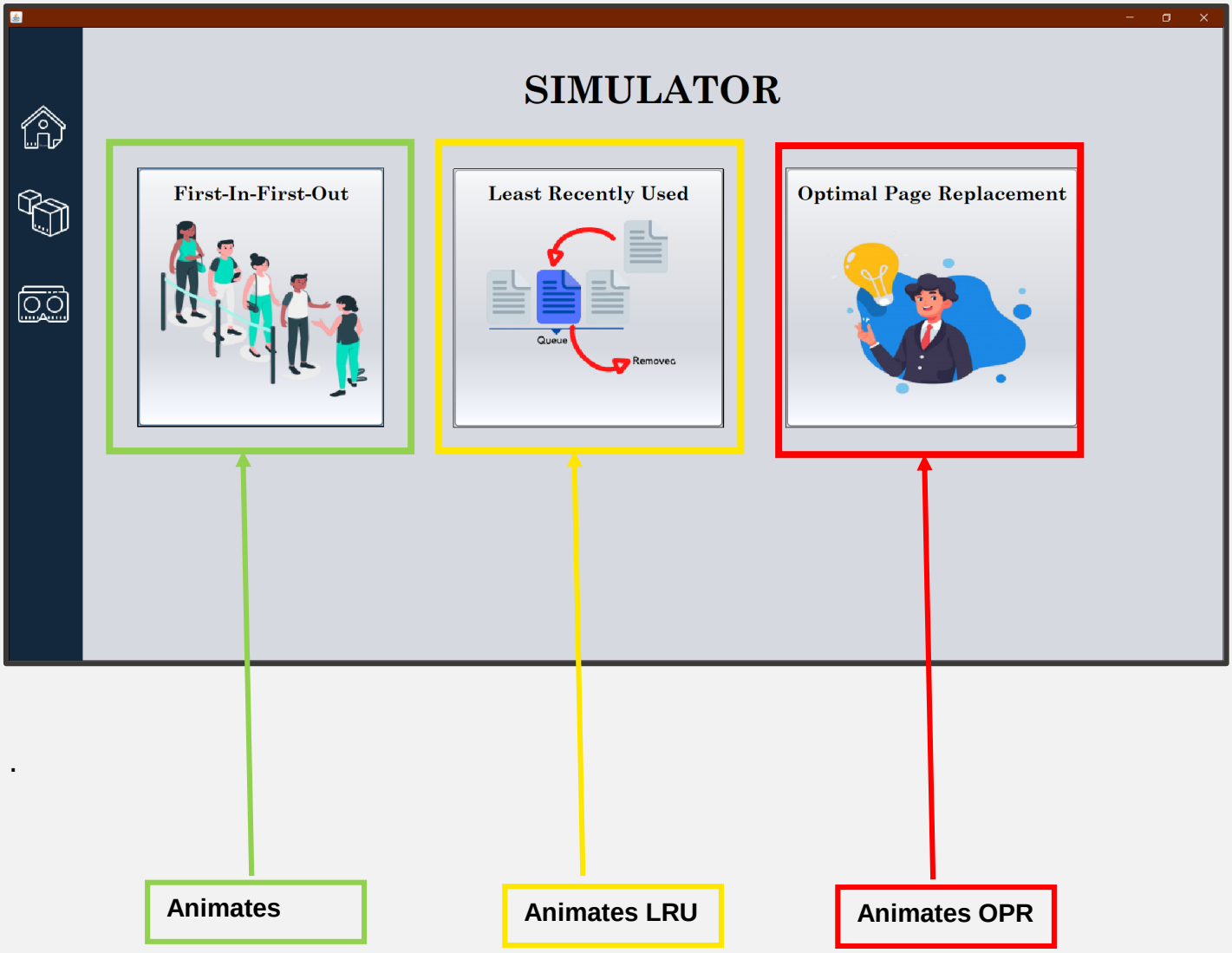
Fig 1: The Home Page

The **What?** Tab explains about the page replacement algorithm. The **Why?** Tab explains about the need of the page replacement algorithms and which kind so algorithms we use. The **Where?** Tab tells us the alternative applications of the page replacement algorithm apart from the Operating System.

02 Simulator Tab of the Desktop App

The **Simulator Tab,** visually shows the video animation with an attached voiceover on how the algorithm is solved and implemented in the computer and how the algorithm should be implemented.

Fig 2: The Simulator Tab



.

The side bar navigation remains similar as mentioned in the Fig 1.

02.1 Showing snapshots of all the algorithms

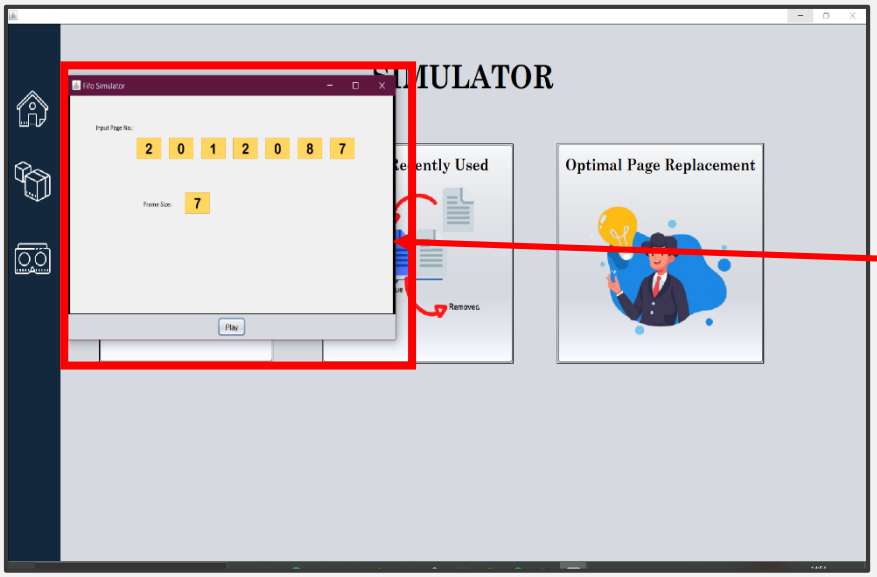
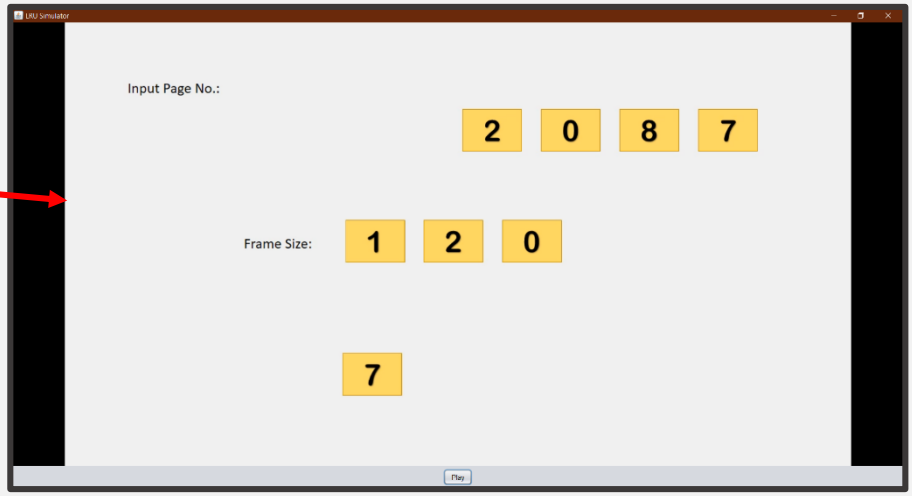


Fig 3 The FIFO Animation Tab

Fig 4 The LRU Animation Tab



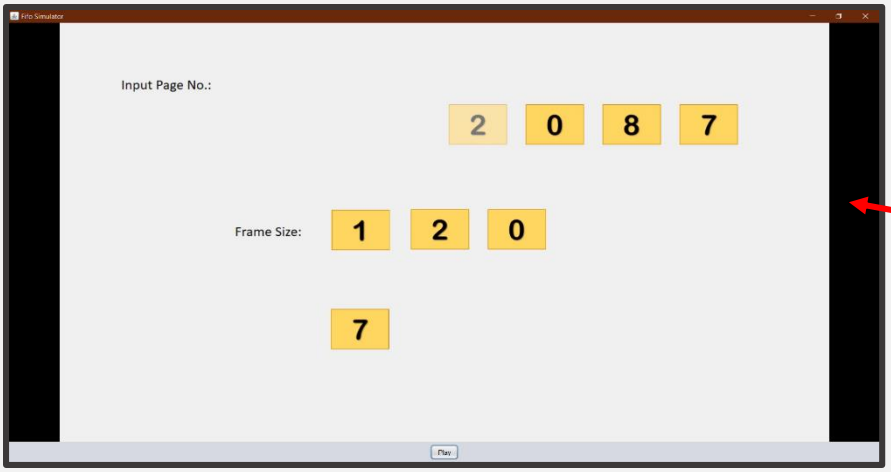
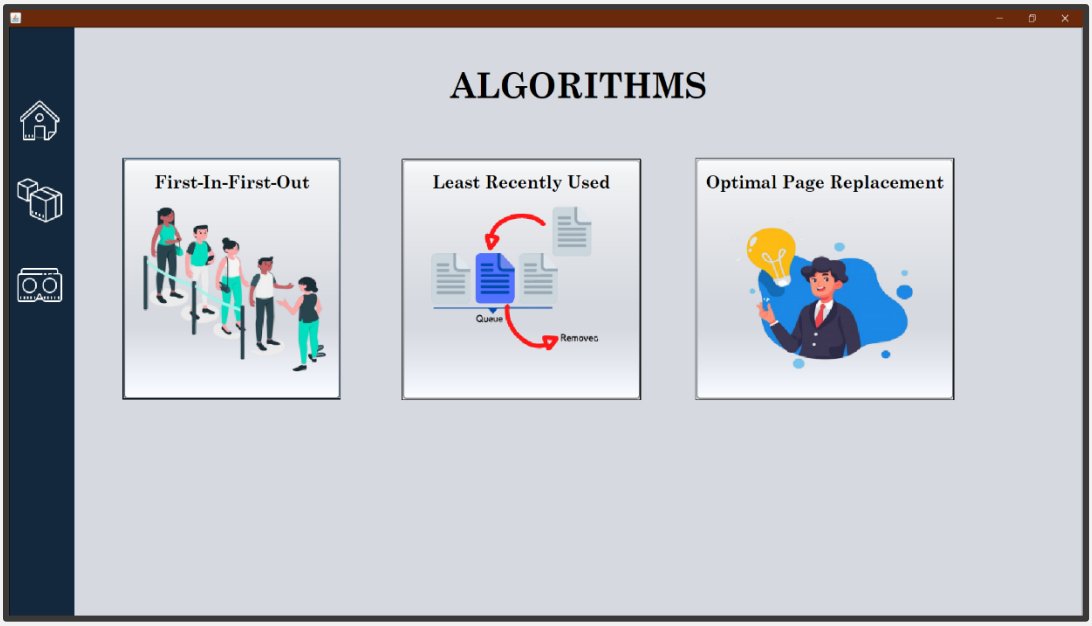


Fig 5 The OPR Animation Tab

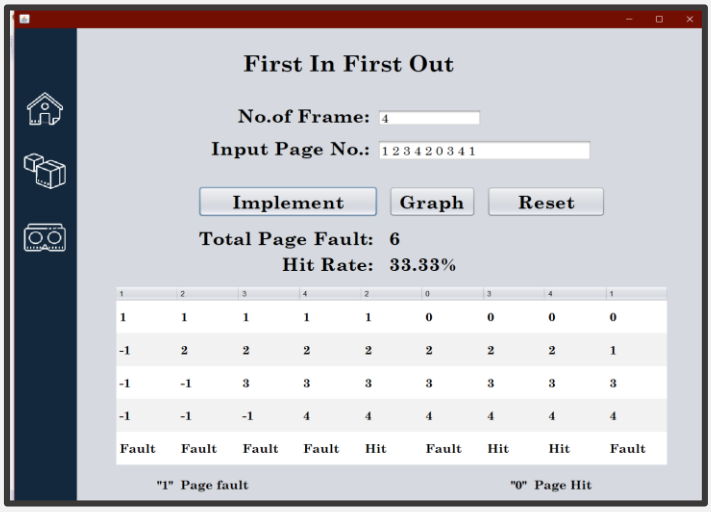
Algorithm Implementation:

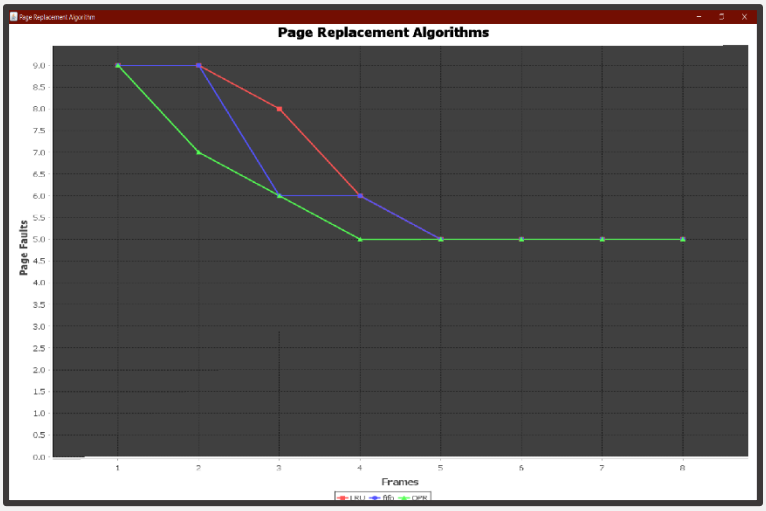
The Algorithms tabs implement all the three algorithms dynamically and provides the graphical output where we compare all the three algorithms for the similar input. The Algorithms also contains a reset button that helps to reset all the values of implemented and thus helps the students to execute examples or sums multiple amount of times without lag.

Fig 6: The Algorithm Tab

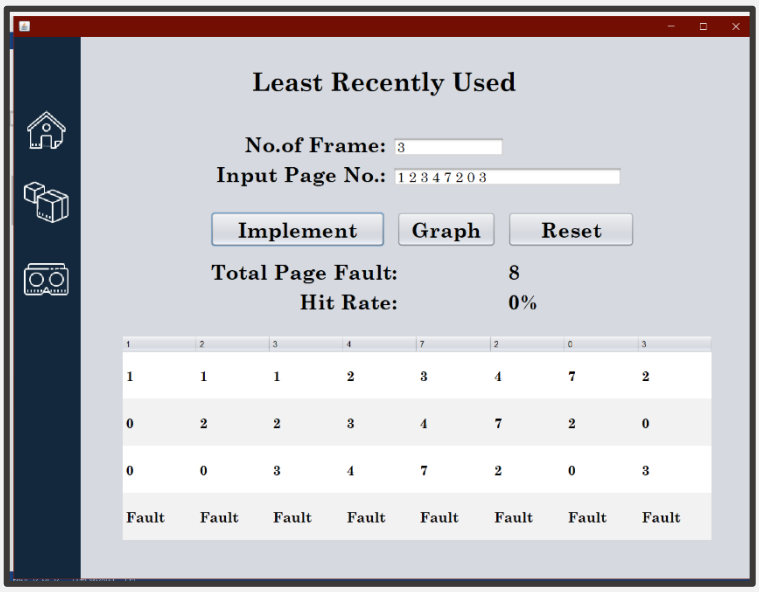


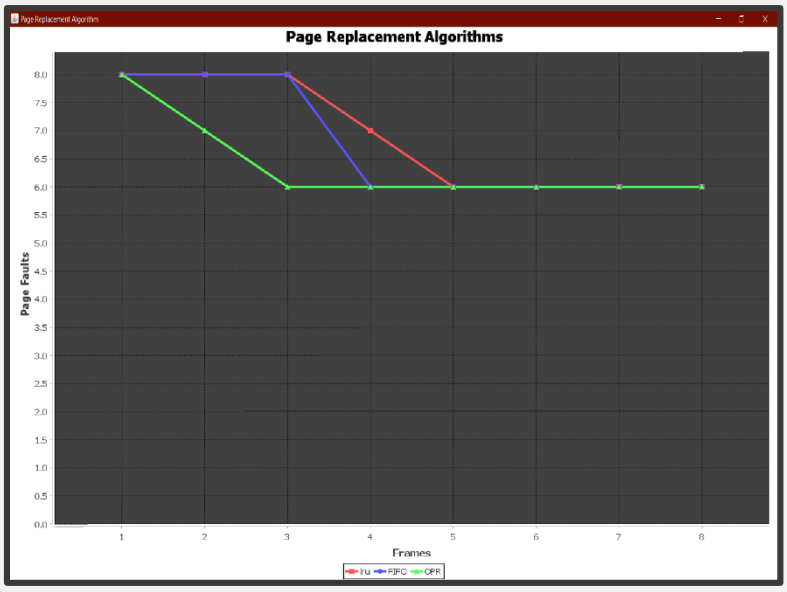
FIFO Algorithm Implementation



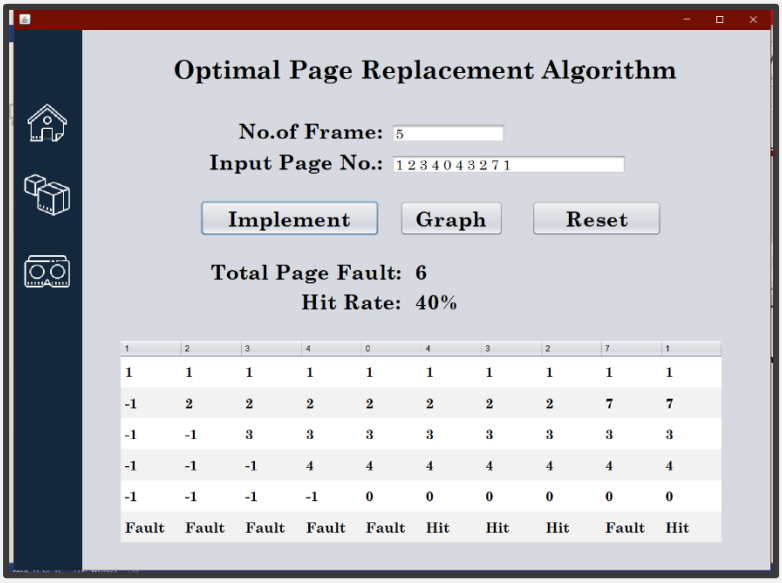


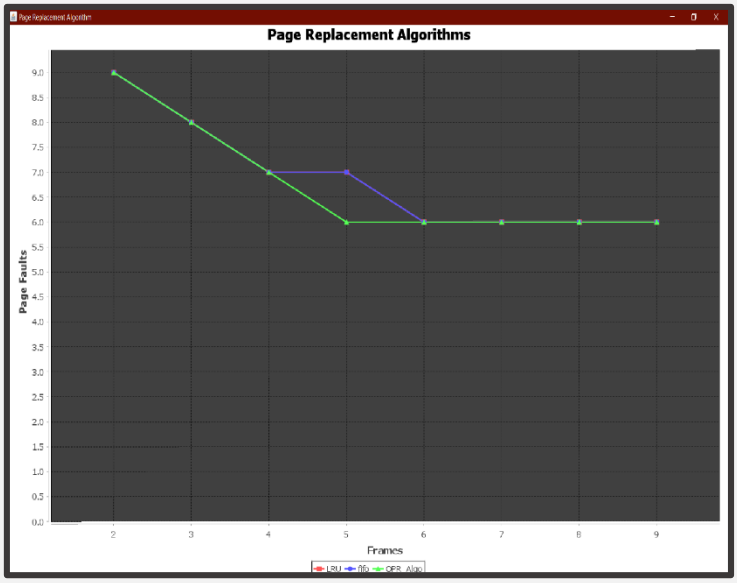
LRU Algorithm Implementation





OPR Algorithm Implementation





Conclusion :

All the above algorithms and tabs are running properly with no errors in the code. Through this project our team got the chance to learn following things given below:

- We learnt that with team work we ca achieve our goal in limited time
- We learnt how to implement different libraries of java and connect all the frames to each other for seamless user experience
- We learned the concept of page replacement algorithm thoroughly.
- We learned to implement animationa and Desktop App with video, voicevers, charts, graphs and content.

Hence, team 34 have implemented all the algorithms successfully within the given time frame and have submitted their video potraying the working of the code properly.

Thanking Chitan sir and Samir Sir for giving us this eminent opportunity to create this GUI and guiding us throughout out project. We are pleased to announce that we have completed our work and will be waiting for your suggestions and constant support in the future.

References :

1. https://www.geeksforgeeks.org/page-replacement-algorithms-in-operating-systems/#:~:text=Optimal%20page%20replacement%20is%20perfect,can%20be%20analyzed%20against%20it.&text=In%20this%20algorithm%20page%20will%20be%20replaced%20which%20is%20least%20recently%20used. (For the content in the report)
2. https://www.flaticon.com/authors/flat-icons (For all the transparent icons used in GUI)
3. https://www.jfree.org/jfreechart/ (For graph in GUI)
4. https://capricasoftware.co.uk/projects/vlcj (For video and voice over animation ins GUI)
5. https://www.eclipse.org/eclipseide/ (For Eclipse IDE 2020)