Jordan Taranto
CS431
Homework 3

main.py

main.py FIFO.py LRU.py OR.py

```python
# Author: Jordan Taranto
# CS431

from OR import optimal_page_replacement
from FIFO import fifo_page_replacement
from LRU import lru_page_replacement

REFERENCE_STRING = [0, 1, 4, 0, 2, 3, 0, 1, 0, 2, 3, 4, 2, 3]
NUM_FRAMES = 4

# Question 1 Optimal algorithm
optimal_page_replacement(REFERENCE_STRING, NUM_FRAMES)
# Question 2 FIFO algorithm
fifo_page_replacement(REFERENCE_STRING, NUM_FRAMES)
# Question 3 LRU algorithm
lru_page_replacement(REFERENCE_STRING, NUM_FRAMES)
# Question 4 Ranking of page algorithm performance
print("Q4 Ranking of algos")
print("1. Optimal Page Replacement Algorithm")
print("2. LRU Page Replacement Algorithm")
print("3. FIFO Page Replacement Algorithm")
```

Optimal Page Replacement

1. Show which pages are resident under the optimal page replacement algorithm. Indicate when page faults happen. *(6 pts)*

OR.py

main.py FIFO.py LRU.py OR.py

```python
# Author: Jordan Taranto
# CS431
# references:
# https://stackoverflow.com/questions/74307591/is-there-a-more-elegant-way-of-finding-minimum-in-array-in-this-case
    # used this for float('inf') to find min value in array which is phenonemnal and fairly fast

# These algos are pretty stratight forward since we have been doing FIFO all semester just different size matrices etc
def fifo_page_replacement(reference_string, num_frames):
    print("\nQ2 FIFO PAGE REPLACEMENT ALGO:")
    frames = [-1] * num_frames
    page_faults = 0
    queue = []

    # simulate page access
    for page in reference_string:
        if page not in frames:
            if -1 in frames:
                frames[frames.index(-1)] = page
            else:
                # pop element in queue and put new page
                frames[queue.pop(0)] = page
            queue.append(frames.index(page))
            page_faults += 1
            print(f"page fault: Page {page} loaded into frames: {frames}")
        else:
            print(f"page {page} already in frames: {frames}")

    print(f"Total page faults: {page_faults}\n")
```

codetoimg.com

Output

```
Q1 OPTIMAL PAGE REPLACEMENT ALGO:
page fault: Page 0 loaded into frames: [0, -1, -1, -1]
page fault: Page 1 loaded into frames: [0, 1, -1, -1]
page fault: Page 4 loaded into frames: [0, 1, 4, -1]
page 0 already in frames: [0, 1, 4, -1]
page fault: Page 2 loaded into frames: [0, 1, 4, 2]
page fault: Page 3 loaded into frames: [0, 1, 3, 2]
page 0 already in frames: [0, 1, 3, 2]
page 1 already in frames: [0, 1, 3, 2]
page 0 already in frames: [0, 1, 3, 2]
page 2 already in frames: [0, 1, 3, 2]
page 3 already in frames: [0, 1, 3, 2]
page fault: Page 4 loaded into frames: [4, 1, 3, 2]
page 2 already in frames: [4, 1, 3, 2]
page 3 already in frames: [4, 1, 3, 2]
Total page faults: 6
```

FIFO

2. Show which pages are resident under the FIFO page replacement algorithm. Indicate when page faults happen. *(6 pts)*

FIFO.py

main.py FIFO.py LRU.py OR.py

```python
# Author: Jordan Taranto
# CS431
# references:
# https://stackoverflow.com/questions/74307591/is-there-a-more-elegant-way-of-finding-minimum-in-array-in-this-case
    # used this for float('inf') to find min value in array which is phenonemnal and fairly fast

# These algos are pretty stratight forward since we have been doing FIFO all semester just different size matrices etc
def fifo_page_replacement(reference_string, num_frames):
    print("\nQ2 FIFO PAGE REPLACEMENT ALGO:")
    frames = [-1] * num_frames
    page_faults = 0
    queue = []

    # simulate page access
    for page in reference_string:
        if page not in frames:
            if -1 in frames:
                frames[frames.index(-1)] = page
            else:
                # pop element in queue and put new page
                frames[queue.pop(0)] = page
            queue.append(frames.index(page))
            page_faults += 1
            print(f"page fault: Page {page} loaded into frames: {frames}")
        else:
            print(f"page {page} already in frames: {frames}")

    print(f"Total page faults: {page_faults}\n")
```


Output

```
Q2 FIFO PAGE REPLACEMENT ALGO:
page fault: Page 0 loaded into frames: [0, -1, -1, -1]
page fault: Page 1 loaded into frames: [0, 1, -1, -1]
page fault: Page 4 loaded into frames: [0, 1, 4, -1]
page 0 already in frames: [0, 1, 4, -1]
page fault: Page 2 loaded into frames: [0, 1, 4, 2]
page fault: Page 3 loaded into frames: [3, 1, 4, 2]
page fault: Page 0 loaded into frames: [3, 0, 4, 2]
page fault: Page 1 loaded into frames: [3, 0, 1, 2]
page 0 already in frames: [3, 0, 1, 2]
page 2 already in frames: [3, 0, 1, 2]
page 3 already in frames: [3, 0, 1, 2]
page fault: Page 4 loaded into frames: [3, 0, 1, 4]
page fault: Page 2 loaded into frames: [2, 0, 1, 4]
page fault: Page 3 loaded into frames: [2, 3, 1, 4]
Total page faults: 10
```

LRU

3. Show which pages are resident under the LRU page replacement algorithm. Indicate when page faults happen. *(6 pts)*

LRU.py

main.py FIFO.py LRU.py

```python
# Author: Jordan Taranto
# CS431

def lru_page_replacement(reference_string, num_frames):
    print("\nQ3 LRU PAGE REPLACEMENT ALGO:")
    # init frames with -1 being the indicator that they are empty frames
    frames = [-1] * num_frames
    page_faults = 0
    # used a dict here to keep track of the last time a page was used
    last_used = {}

    # simulate page replacement
    for i, page in enumerate(reference_string):
        # check if frame is in page
        if page not in frames:
            # increment page fault because page isn't present in frame
            page_faults += 1
            # fill first empy frame if exists
            if -1 in frames:
                frames[frames.index(-1)] = page
            # find leaset recentl used page and replace LRU page with current page
            else:
                lru_page = min(last_used, key=last_used.get)
                # check if lru page exists otherwise errors out
                if lru_page in frames:
                    frames[frames.index(lru_page)] = page
                else:
                    frames[0] = page
        # if no page fault print status of frame
        else:
            print(f"Page {page} already in frames: {frames}")
        last_used[page] = i

    print(f"Total page faults: {page_faults}\n")
```


Output

```
Q3 LRU PAGE REPLACEMENT ALGO:
Page 0 already in frames: [0, 1, 4, -1]
Page 0 already in frames: [0, 3, 4, 2]
Page 2 already in frames: [1, 3, 0, 2]
Page 3 already in frames: [1, 3, 0, 2]
Page 2 already in frames: [4, 3, 0, 2]
Page 3 already in frames: [4, 3, 0, 2]
Total page faults: 8
```

Comparison

4. Compare and rank all three algorithms in order of best performance. *(2 pts)*

Ranking

```
Q4 Ranking of algos
1. Optimal Page Replacement Algorithm
2. LRU Page Replacement Algorithm
3. FIFO Page Replacement Algorithm
taranto@taranto cs431 %
```