/// spos practical 7   
//// fifo code   
def fifo\_page\_replacement():

    # Input for number of pages

    noofpages = int(input("Enter the number of pages you want to enter: "))

    pages = list(map(int, input("Enter the pages (space-separated): ").split()))

    # Input for frame capacity

    capacity = int(input("Enter the capacity of frame: "))

    frame = [-1] \* capacity

    table = [[-1] \* capacity for \_ in range(noofpages)]

    hit = 0

    fault = 0

    index = 0

    print("\n----------------------------------------------------------------------")

    # Process each page

    for i in range(noofpages):

        search = -1

        # Check if page is already in frame (hit)

        for j in range(capacity):

            if frame[j] == pages[i]:

                search = j

                hit += 1

                print(" H ", end="")

                break

        # If page is not found (fault)

        if search == -1:

            frame[index] = pages[i]  # Replace page in FIFO order

            fault += 1

            print(" F ", end="")

            index += 1

            if index == capacity:

                index = 0  # Reset index if it reaches capacity

        # Store current frame status

        for j in range(capacity):

            table[i][j] = frame[j]

    print("\n----------------------------------------------------------------------")

    # Print the page frames at each step

    for i in range(capacity):

        for j in range(noofpages):

            if table[j][i] == -1:

                print("  - ", end="")

            else:

                print(f"{table[j][i]:3d}", end=" ")

        print()

    print("----------------------------------------------------------------------")

    # Calculate and display hit and fault ratios

    fault\_ratio = (fault / noofpages) \* 100

    hit\_ratio = (hit / noofpages) \* 100

    print(f"Page Fault: {fault}\nPage Hit: {hit}")

    print(f"Hit Ratio: {hit\_ratio:.2f}% \nFault Ratio: {fault\_ratio:.2f}%")

# Run the function

fifo\_page\_replacement()

// fifo output  
Enter the number of pages you want to enter: 7

Enter the pages (space-separated): 1 3 0 3 5 6 3

Enter the capacity of frame: 3

----------------------------------------------------------------------

F F F H F F F

----------------------------------------------------------------------

1 1 1 1 5 5 5

- 3 3 3 3 6 6

- - 0 0 0 0 3

----------------------------------------------------------------------

Page Fault: 6

Page Hit: 1

Hit Ratio: 14.29%

Fault Ratio: 85.71%

PS C:\Users\HP> === Code Execution Successful ===  
  
  
//// least recently used code   
def least\_recently\_used():

    # Input for number of pages

    noofpages = int(input("Enter the number of pages you want to enter: "))

    pages = list(map(int, input("Enter the pages (space-separated): ").split()))

    # Input for frame capacity

    capacity = int(input("Enter the capacity of frame: "))

    frame = [-1] \* capacity

    table = [[-1] \* capacity for \_ in range(noofpages)]

    arr = []  # To keep track of the order of pages for LRU

    hit = 0

    fault = 0

    index = 0

    isFull = False

    print("----------------------------------------------------------------------")

    # Process each page

    for i in range(noofpages):

        if pages[i] in arr:

            arr.remove(pages[i])

        arr.append(pages[i])

        search = -1

        # Check if page is already in frame (hit)

        for j in range(capacity):

            if frame[j] == pages[i]:

                search = j

                hit += 1

                print(" H ", end="")

                break

        # If page is not found (fault)

        if search == -1:

            if isFull:

                # Find the least recently used page

                min\_loc = noofpages

                for j in range(capacity):

                    if frame[j] in arr:

                        temp = arr.index(frame[j])

                        if temp < min\_loc:

                            min\_loc = temp

                            index = j

            frame[index] = pages[i]

            fault += 1

            print(" F ", end="")

            index += 1

            if index == capacity:

                index = 0

                isFull = True

        # Store current frame status

        for j in range(capacity):

            table[i][j] = frame[j]

    print("\n----------------------------------------------------------------------")

    # Print the page frames at each step

    for i in range(capacity):

        for j in range(noofpages):

            if table[j][i] == -1:

                print("  - ", end="")

            else:

                print(f"{table[j][i]:3d}", end=" ")

        print()

    print("----------------------------------------------------------------------")

    # Calculate and display hit and fault ratios

    hit\_ratio = (hit / noofpages) \* 100

    fault\_ratio = (fault / noofpages) \* 100

    print(f"Page Fault: {fault}\nPage Hit: {hit}")

    print(f"Hit Ratio: {hit\_ratio:.2f}% \nFault Ratio: {fault\_ratio:.2f}%")

# Run the function

least\_recently\_used()

//lru output  
S C:\Users\HP> & "C:/Program Files/Python312/python.exe" "c:/Users/HP/OneDrive/Desktop/spos practical/lexical/fifo.py"

Enter the number of pages you want to enter: 14

Enter the pages (space-separated): 7 0 1 2 0 3 0 4 2 3 0 3 2 3

Enter the capacity of frame: 4

----------------------------------------------------------------------

F F F F H F H F H H H H H H

----------------------------------------------------------------------

7 7 7 7 7 3 3 3 3 3 3 3 3 3

- 0 0 0 0 0 0 0 0 0 0 0 0 0

- - 1 1 1 1 1 4 4 4 4 4 4 4

- - - 2 2 2 2 2 2 2 2 2 2 2

----------------------------------------------------------------------

Page Fault: 6

Page Hit: 8

Hit Ratio: 57.14%

Fault Ratio: 42.86%

PS C:\Users\HP>  
/// optimal page replacement algorithm code  
  
def optimal\_page\_replacement(pages, capacity):

    noofpages = len(pages)

    frame = [-1] \* capacity

    table = [[-1] \* capacity for \_ in range(noofpages)]

    ptr = 0

    hit = 0

    fault = 0

    is\_full = False

    print("----------------------------------------------------------------------")

    for i in range(noofpages):

        search = -1

        # Check if the page is already in the frame

        for j in range(capacity):

            if frame[j] == pages[i]:

                search = j

                hit += 1

                print(f"{'H':>4}", end="")

                break

        if search == -1:

            if is\_full:

                index = [None] \* capacity

                # Find the optimal page to replace

                for j in range(i + 1, noofpages):

                    for k in range(capacity):

                        if pages[j] == frame[k] and index[k] is None:

                            index[k] = j

                            break

                # Find the page in the frame that is either not referenced again or has the farthest reference

                max\_index = -1

                ptr = -1

                for j in range(capacity):

                    if index[j] is None:

                        ptr = j

                        break

                    elif index[j] > max\_index:

                        max\_index = index[j]

                        ptr = j

            # Replace the page in the frame

            frame[ptr] = pages[i]

            fault += 1

            print(f"{'F':>4}", end="")

            if not is\_full:

                ptr += 1

                if ptr == capacity:

                    ptr = 0

                    is\_full = True

        # Store the current frame configuration

        for j in range(capacity):

            table[i][j] = frame[j]

    print("\n----------------------------------------------------------------------")

    for i in range(capacity):

        for j in range(noofpages):

            print(f"{table[j][i]:3d} ", end="")

        print()

    print("----------------------------------------------------------------------")

    hit\_ratio = (hit / noofpages) \* 100

    fault\_ratio = (fault / noofpages) \* 100

    print(f"Page Fault: {fault}\nPage Hit: {hit}")

    print(f"Hit Ratio: {hit\_ratio:.2f} \nFault Ratio: {fault\_ratio:.2f}")

# Input from the user

noofpages = int(input("Enter the number of pages you want to enter: "))

pages = list(map(int, input("Enter the pages separated by space: ").split()))

capacity = int(input("Enter the capacity of frame: "))

# Call the function

optimal\_page\_replacement(pages, capacity)

// code output  
  
PS C:\Users\HP> & "C:/Program Files/Python312/python.exe" "c:/Users/HP/OneDrive/Desktop/spos practical/lexical/fifo.py"

Enter the number of pages you want to enter: 14

Enter the pages separated by space: 7 0 1 2 0 3 0 4 2 3 0 3 2 3

Enter the capacity of frame: 4

----------------------------------------------------------------------

F F F F H F H F H H H H H H

----------------------------------------------------------------------

7 7 7 7 7 3 3 3 3 3 3 3 3 3

-1 0 0 0 0 0 0 0 0 0 0 0 0 0

-1 -1 1 1 1 1 1 4 4 4 4 4 4 4

-1 -1 -1 2 2 2 2 2 2 2 2 2 2 2

----------------------------------------------------------------------

Page Fault: 6

Page Hit: 8

Hit Ratio: 57.14

Fault Ratio: 42.86

=== Code Execution Successful ===