Rajalakshmi Engineering College

Name: karthieswaran L
Email: 241801117@rajalakshmi.edu.in
Roll no: 241801117
Phone: 9840695531
Branch: REC
Department: I AI & DS FB
Batch: 2028
Degree: B.E - AI & DS

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 4_COD_Question 4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In an office setting, a print job management system is used to efficiently handle and process print jobs. The system is implemented using a queue data structure with an array.

The program provides the following operations:

Enqueue Print Job: Add a print job with a specified number of pages to the end of the queue.Dequeue Print Job: Remove and process the next print job in the queue.Display Queue: Display the print jobs in the queue

The program should ensure that print jobs are processed in the order they are received.

Input Format

The input consists of integers corresponding to the operation that needs to be performed:

Choice 1: Enqueue the print job into the queue. If the choice is 1, the following input is a space-separated integer, representing the pages to be enqueued into the queue.

Choice 2: Dequeue a print job from the queue.

Choice 3: Display the print jobs in the queue.

Choice 4: Exit the program.

Output Format

The output displays messages according to the choice and the status of the queue:

If the choice is 1:

1. Insert the given page into the queue and display "Print job with [page] pages is enqueued." where [page] is the number of pages that are inserted.
2. If the queue is full, print "Queue is full. Cannot enqueue."

If the choice is 2:

1. Dequeue a page from the queue and display "Processing print job: [page] pages" where [page] is the corresponding page that is dequeued.
2. If the queue is empty without any elements, print "Queue is empty."

If the choice is 3:

1. The output prints "Print jobs in the queue: " followed by the space-separated pages present in the queue.
2. If there are no elements in the queue, print "Queue is empty."

If the choice is 4:

1. Exit the program and print "Exiting program"

If any other choice is entered, the output prints "Invalid option."

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1
10
1
20
1
30
1
40
1
50
1
60
3
2
3
4

Output: Print job with 10 pages is enqueued.
Print job with 20 pages is enqueued.
Print job with 30 pages is enqueued.
Print job with 40 pages is enqueued.
Print job with 50 pages is enqueued.
Queue is full. Cannot enqueue.
Print jobs in the queue: 10 20 30 40 50
Processing print job: 10 pages
Print jobs in the queue: 20 30 40 50
Exiting program

Answer

```
    //Type your code here
    //Type your code here
int isEmpty() {
    return front == -1;  // Changed condition to handle initialization
}
```

```
int isFull() {
    return (front == 0 && rear == MAX_SIZE-1) || (rear == (front-1)%(MAX_SIZE-1));
}

void enqueue(int pages) {
    if (isFull()) {
        printf("Queue is full. Cannot enqueue.\n");
        return;
    }
    else if (front == -1) {
        front = rear = 0;
    }
    else if (rear == MAX_SIZE-1 && front != 0) {
        rear = 0;
    }
    else {
        rear++;
    }
    queue[rear] = pages;
    printf("Print job with %d pages is enqueued.\n", pages);
}

void dequeue() {
    if (isEmpty()) {
        printf("Queue is empty.\n");
        return;
    }
    int data = queue[front];
    if (front == rear) {
        front = -1;
        rear = -1;
    }
    else if (front == MAX_SIZE-1) {
        front = 0;
    }
    else {
        front++;
    }
    printf("Processing print job: %d pages\n", data);
}

void display() {
```

```
    if (isEmpty()) {
        printf("Queue is empty.\n");
        return;
    }
    printf("Print jobs in the queue:");
    if (rear >= front) {
        for (int i = front; i <= rear; i++) {
            printf(" %d", queue[i]);
        }
    }
    else {
        for (int i = front; i < MAX_SIZE; i++) {
            printf(" %d", queue[i]);
        }
        for (int i = 0; i <= rear; i++) {
            printf(" %d", queue[i]);
        }
    }
    printf("\n");
}
```

Status : Correct ***Marks : 10/10***