ASSIGNMENT-6 - PRINTER SPOOLER

/*Name: Sahil Badve PRN: B24CE1114 Div: S.Y.B-Tech 2 Batch C*/

/*Printer Spooler (Circular Queue):
In a multi-user environment, printers often use a circular queue to manage print jobs. Each print job is added to the queue, and the printer processes them in the order they arrive. Once a print job is completed, it moves out of the queue, and the next job is processed, efficiently managing the flow of print tasks. Implement the Printer Spooler system using a circular queue without using built-in queues.
*/

```
#include <iostream>
using namespace std;

#define SIZE 10   // maximum number of print jobs in the spooler

class PrinterSpooler {
private:
    int jobs[SIZE];   // array to hold job IDs
    int front, rear, count;

public:
    PrinterSpooler() {
        front = 0;
        rear = -1;
        count = 0;
    }

    // Check if queue is empty
    bool isEmpty() {
        return (count == 0);
    }

    // Check if queue is full
    bool isFull() {
        return (count == SIZE);
    }

    // Add a new print job
    void addJob(int jobID) {
        if (isFull()) {
            cout << "Spooler is FULL. Cannot add job " << jobID << endl;
            return;
```

```
        }
        rear = (rear + 1) % SIZE;
        jobs[rear] = jobID;
        count++;
        cout << "Added print job: " << jobID << endl;
    }

    // Process and remove a print job
    void processJob() {
        if (isEmpty()) {
            cout << "No jobs in the spooler to process." << endl;
            return;
        }
        cout << "Processing print job: " << jobs[front] << endl;
        front = (front + 1) % SIZE;
        count--;
    }

    // Display all jobs in the queue
    void displayQueue() {
        if (isEmpty()) {
            cout << "No jobs in the spooler." << endl;
            return;
        }
        cout << "Jobs in spooler: ";
        int i = front;
        for (int c = 0; c < count; c++) {
            cout << jobs[i] << " ";
            i = (i + 1) % SIZE;
        }
        cout << endl;
    }
};

// Driver program
int main() {
    PrinterSpooler spooler;
    int choice, jobID;

    cout << "Printer Spooler" << endl;

    do {
        cout << "\n1. Add Print Job";
        cout << "\n2. Process Print Job";
```

```
        cout << "\n3. Display Spooler Queue";
        cout << "\n4. Exit";
        cout << "\nEnter your choice: ";
        cin >> choice;

        switch (choice) {
        case 1:
            cout << "Enter Job ID: ";
            cin >> jobID;
            spooler.addJob(jobID);
            break;

        case 2:
            spooler.processJob();
            break;

        case 3:
            spooler.displayQueue();
            break;

        case 4:
            cout << "Exiting Printer Spooler..." << endl;
            break;

        default:
            cout << "Invalid choice. Try again." << endl;
        }
    } while (choice != 4);

    return 0;
}
```

OUTPUT:-

Printer Spooler

1. Add Print Job
2. Process Print Job
3. Display Spooler Queue

4. Exit
Enter your choice: 1
Enter Job ID: 123
Added print job: 123

1. Add Print Job
2. Process Print Job
3. Display Spooler Queue
4. Exit
Enter your choice: 2
Processing print job: 123

1. Add Print Job
2. Process Print Job
3. Display Spooler Queue
4. Exit
Enter your choice: 3
No jobs in the spooler.

1. Add Print Job
2. Process Print Job
3. Display Spooler Queue
4. Exit
Enter your choice: 1
Enter Job ID: 456
Added print job: 456

1. Add Print Job
2. Process Print Job
3. Display Spooler Queue
4. Exit
Enter your choice: 3
Jobs in spooler: 456

1. Add Print Job
2. Process Print Job
3. Display Spooler Queue
4. Exit
Enter your choice: 7
Invalid choice. Try again.

1. Add Print Job
2. Process Print Job
3. Display Spooler Queue

4. Exit
Enter your choice: 4
Exiting Printer Spooler...