

Code:

class Account {

    static int amount = 50000;

    // synchronized method

    synchronized void withdraw(int amount) {

        if (this.amount < amount) {

            try {

                wait();

            } catch (Exception e) {

            }

        }

        this.amount = this.amount - amount;

        System.out.println("amount withdraw = " + amount);

        System.out.println("amount remaining = " + this.amount);

    }

    // synchronized method

    void deposit(int amount) {

        synchronized (this) {

            this.amount = this.amount + amount;

            System.out.println("amount deposit = " + amount);

            System.out.println("total amount after deposit = " + this.amount);

        }

    }

}

class UserA extends Thread {

    Account a = new Account();

    public void run() {

        a.withdraw(45000);

    }

}

class UserB extends Thread {

    Account a = new Account();

    public void run() {

        a.withdraw(20000);

    }

}

public class Main {

    public static void main(String[] args) {

        UserA ua = new UserA();

        UserB ub = new UserB();

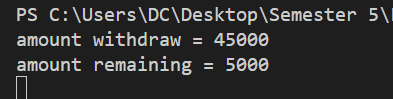
        ua.start();

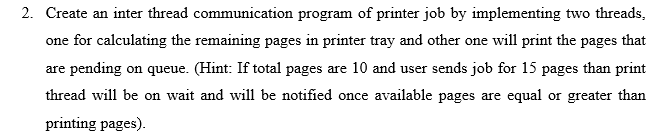
        ub.start();

    }

}

Output:





Code:

class Printer {

    private int pagesInTray = 10; // Initial pages in the printer tray

    // Method to simulate the process of printing pages

    public synchronized void printPages(int pages) {

        while (pagesInTray < pages) {

            try {

                // Wait if there are not enough pages to print

                System.out.println("Not enough pages. Waiting for more pages...");

                wait();

            } catch (InterruptedException e) {

                System.out.println(e);

            }

        }

        pagesInTray -= pages; // Reduce pages from the tray

        System.out.println("Printing " + pages + " pages. Remaining pages: " + pagesInTray);

    }

    // Method to add pages to the printer tray

    public synchronized void addPages(int pages) {

        pagesInTray += pages; // Add pages to the tray

        System.out.println(pages + " pages added. Total pages in tray: " + pagesInTray);

        notify(); // Notify the printing thread that pages are available

    }

}

class PageCalculator extends Thread {

    private Printer printer;

    // Constructor to initialize the printer instance

    public PageCalculator(Printer printer) {

        this.printer = printer;

    }

    @Override

    public void run() {

        // Simulate the process of adding pages to the printer tray

        try {

            for (int i = 0; i < 3; i++) {

                // Simulate a delay in adding pages

                Thread.sleep(1000);

                printer.addPages(5); // Add 5 pages every second

            }

        } catch (InterruptedException e) {

            System.out.println(e);

        }

    }

}

class PrintJob extends Thread {

    private Printer printer;

    private int pagesToPrint;

    // Constructor to initialize the printer and pages to print

    public PrintJob(Printer printer, int pagesToPrint) {

        this.printer = printer;

        this.pagesToPrint = pagesToPrint;

    }

    @Override

    public void run() {

        // Simulate the process of printing pages

        printer.printPages(pagesToPrint);

    }

}

public class Lab6Task2 {

    public static void main(String[] args) {

        Printer printer = new Printer();

        // Creating threads for printing jobs and adding pages to the tray

        PrintJob printJob1 = new PrintJob(printer, 15); // Job for 15 pages

        PrintJob printJob2 = new PrintJob(printer, 5); // Job for 5 pages

        PageCalculator pageCalculator = new PageCalculator(printer);

        // Start all threads

        pageCalculator.start();

        printJob1.start();

        printJob2.start();

    }

}

Output:

