Необхідно виконати обидва завдання

Task 1. Parallel algorithm and shared state

Переказ коштів

У Вас є клас ***Bank*** з методом *transfer ()* для переказу грошей з одного рахунку на інший в межах банку.

```
public class Bank {
    public void transfer(Account from, Account to,
int amount){
        atomic {
            from.withdraw(amount);
            to.deposit(amount);
        }
    }
}
```

На жаль, в Java відсутня конструкція типу *atomic* для виконання транзакцій.

Напишіть свою реалізація тіла методу *transfer()*, яка могла б працювати в багатопотоковому середовищі.

Під час переказу грошей з рахунку на рахунок, дані рахунки повинні блокуватися. Подумайте, як при цьому уникнути deadlock-ів.

Виконайте тестування методу *transfer()*. Для цього:

1. Створіть кілька десятків (сотень) рахунків і покладіть на них випадкову кількість грошей
2. Підрахуйте скільки всього є грошей (сума грошей на всіх рахунках)
3. Запустіть в декількох тисячах потоків одночасний переклад випадкових сум грошей з випадкового рахунку на випадковий рахунок (сума на рахунку не може бути негативною)

Дочекайтеся закінчення переказів і підрахуйте скільки грошей є всього в банку (сума грошей на всіх рахунках). Сума грошей в банку до перекладів і після, повинні збігатися.

Task 2. Producer-consumer task

Завдання. Реалізувати потокобезпечний кільцевий буфер та виконання наступних вимог:

1) П’ять потоків генерують рядки в кільцевий буфер. Рядок має формат: «Потік № … згенерував повідомлення …»
2) Інші два потоки перекладають рядки з першого кільцевого у другий кільцевий буфер. У другий кільцевий буфер записується рядок наступного формату: «Потік № … переклав повідомлення …».
3) Основний потік програми вичитує та роздруковує 100 повідомлень із другого буфера.

Потоки, описані у вимогах 1)-2) повинні бути потоками-демонами, інакше програма не завершить своє виконання.

Кільцевий буфер – це зв’язний список, який замкнений у кільце. Кільцевий буфер зберігає значення за допомогою двох індексів: головного та кінцевого. При занесенні даних в буфер збільшується кінцевий індекс. При видобуванні даних із буфера збільшується головний індекс. При виході за границю списку індекси автоматично корегуються, переходячи до першого елементу списку. Ситуація, коли головний елемент співпадає із кінцевим, інтерпретується як порожній буфер.

При роботі з кільцевим буфером врахувати наступне:

1) якщо кільцевий буфер порожній, то потік, що видобуває дані, повинен зачекати, поки у буфері з’являться дані;
2) якщо кільцевий буфер повний, то потік, що додає дані, повинен зачекати, поки у буфері не з’явиться місце.