실행 결과

```
User1Thread: 100
User2Thread: 50
```

```java
public class SynchronizedExample {
  public static void main(String[] args) {
    Calculator calculator = new Calculator();

    User1Thread user1Thread = new User1Thread();
    user1Thread.setCalculator(calculator);
    user1Thread.start();

    User2Thread user2Thread = new User2Thread();
    user2Thread.setCalculator(calculator);
    user2Thread.start();
  }
}
```

```java
public class Calculator {
  private int memory;

  public int getMemory() {
    return memory;
  }

  public synchronized void setMemory1(int memory) {
    this.memory = memory;
    try {
      Thread.sleep(2000);
    } catch(InterruptedException e) {}
    System.out.println(Thread.currentThread().getName() + ": " + this.memory);
  }

  public void setMemory2(int memory) {
    synchronized(this) {
      this.memory = memory;
      try {
        Thread.sleep(2000);
      } catch(InterruptedException e) {}
      System.out.println(Thread.currentThread().getName() + ": " + this.memory);
    }
  }
}
```

동기화 메소드

메모리 값 저장

2초간 일시 정지

메모리 값 읽기

동기화 블록

메모리 값 저장

2초간 일시 정지

메모리 값 읽기

```java
public class User1Thread extends Thread {
  private Calculator calculator;

  public User1Thread() {
    setName("User1Thread");
  }

  public void setCalculator(Calculator calculator) {
    this.calculator = calculator;
  }

  @Override
  public void run() {
    calculator.setMemory1(100);
  }
}
```

스레드 이름 변경

외부에서 공유 객체인 Calculator를 받아 필드에 저장

동기화 메소드 호출

```java
public class User2Thread extends Thread {
  private Calculator calculator;

  public User2Thread() {
    setName("User2Thread");
  }

  public void setCalculator(Calculator calculator) {
    this.calculator = calculator;
  }

  @Override
  public void run() {
    calculator.setMemory2(50);
  }
}
```

스레드 이름 변경

외부에서 공유 객체인 Calculator를 받아 필드에 저장

동기화 블록을 가진 메소드 호출