

ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

ассистент

должность, уч. степень, звание

подпись, дата

Н. И. Чулочникова

инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ № 2

МНОГОПОТОЧНОЕ ПРОГРАММИРОВАНИЕ

по курсу:

КРОССПЛАТФОРМЕННОЕ ПРОГРАММИРОВАНИЕ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. №

4326

подпись, дата

Г. С. Томчук

инициалы, фамилия

Санкт-Петербург 2025

1 Задание

Работа выполнялась по варианту № 17.

Напишите программу, которая каждую секунду отображает на экране данные о времени, прошедшем от начала сессии, а другой её поток выводит сообщение каждые 5 секунд. Предусмотрите возможность ежесекундного оповещения потока, воспроизводящего сообщение, потоком, отсчитывающим время. Не внося изменений в код потока-"хронометра", добавьте ещё один поток, который выводит на экран другое сообщение каждые 7 секунд.

2 Листинг программы

Ниже представлен листинг полной программы, реализованной на Java.

Листинг 1 — Листинг программы

```
public class MultiThreadLab {
    public static void main(String[] args) {
        Object lock = new Object(); // общий объект синхронизации

        TimerThread timer = new TimerThread(lock);
        MessageThread message5 = new MessageThread(lock, 5, "Сообщение каждые 5 секунд");
        MessageThread message7 = new MessageThread(lock, 7, "Сообщение каждые 7 секунд");

        timer.start();
        message5.start();
        message7.start();
    }
}

// Поток-хронометр
class TimerThread extends Thread {
    private final Object lock;
    private int secondsPassed = 0;

    public TimerThread(Object lock) {
        this.lock = lock;
    }

    @Override
    public void run() {
        try {
            while (true) {
                Thread.sleep(1000);
                secondsPassed++;
                System.out.println("Прошло секунд: " + secondsPassed);
                synchronized (lock) {
                    lock.notifyAll(); // оповещаем другие потоки
                }
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

```

    }
}

// Поток, который выводит сообщение message каждые interval секунд
class MessageThread extends Thread {
    private final Object lock;
    private final int interval;
    private final String message;
    private int currentSeconds = 0;

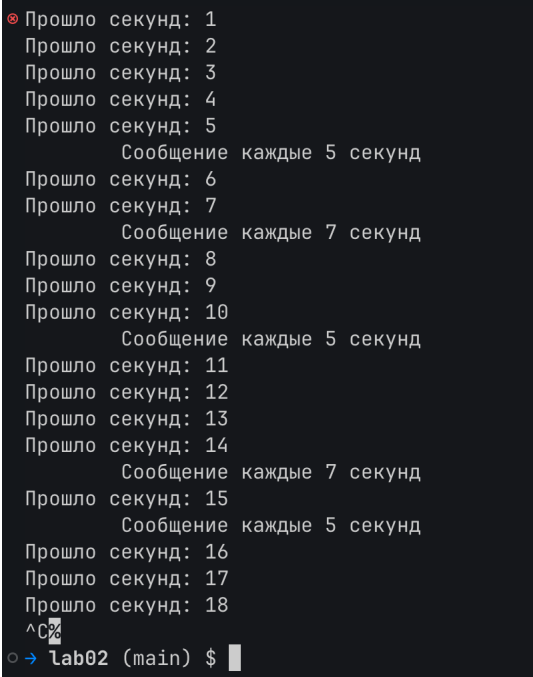
    public MessageThread(Object lock, int interval, String message) {
        this.lock = lock;
        this.interval = interval;
        this.message = message;
    }

    @Override
    public void run() {
        try {
            while (true) {
                synchronized (lock) {
                    lock.wait(); // ждём уведомления от хронометра
                    currentSeconds++;
                    if (currentSeconds % interval == 0) {
                        System.out.println("\t" + message);
                    }
                }
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}

```

3 Результаты работы программы

На рис. 1 изображен результат исполнения программы.



```

Прошло секунд: 1
Прошло секунд: 2
Прошло секунд: 3
Прошло секунд: 4
Прошло секунд: 5
    Сообщение каждые 5 секунд
Прошло секунд: 6
Прошло секунд: 7
    Сообщение каждые 7 секунд
Прошло секунд: 8
Прошло секунд: 9
Прошло секунд: 10
    Сообщение каждые 5 секунд
Прошло секунд: 11
Прошло секунд: 12
Прошло секунд: 13
Прошло секунд: 14
    Сообщение каждые 7 секунд
Прошло секунд: 15
    Сообщение каждые 5 секунд
Прошло секунд: 16
Прошло секунд: 17
Прошло секунд: 18
^C
lab02 (main) $

```

Рисунок 1 — Тестирование программы