МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНСТИТУТ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ СИСТЕМ

ФАКУЛЬТЕТ КОМПЬЮТЕРНЫХ И ФИЗИКО-МАТЕМАТИЧЕСКИХ НАУК

КАФЕДРА ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Отчет по лабораторной №4

по дисциплине «Системное программирование»

**Синхронизация потоков**

|  |  |  |  |
| --- | --- | --- | --- |
| Выполнил: | студент гр. ФИБ-3301-51-01 |  | / В. Р. Кочкин / |
| Проверил: | к.ф.-м.н. доцент каф. ПМиИ |  | / В. А. Бызов / |

Киров 2022

Цель работы

Изучить классические задачи синхронизации. Обеспечить правильную

работу программы с использованием различных примитивов синхронизации.

Задания

Все задачи были реализованы на языке программирования *go*, так как в нём есть встроенные механизмы синхронизации, и мне было интересно научиться использовать новые для себя инструменты в *go*.

**Задание 1**

«Производители – потребители». Два потока обрабатывают заявки. Первый поток читает заявку и помещает её в очередь размера N, если очередь переполнена, то заявке отказывается. Второй поток берет заявки из очереди и обрабатывает их, если в очереди заявок нет, то поток засыпает. Написать параллельную программу, реализующую правильную работу этих потоков.

**Полученные результаты**

Изображение выглядит как текст

Автоматически созданное описание

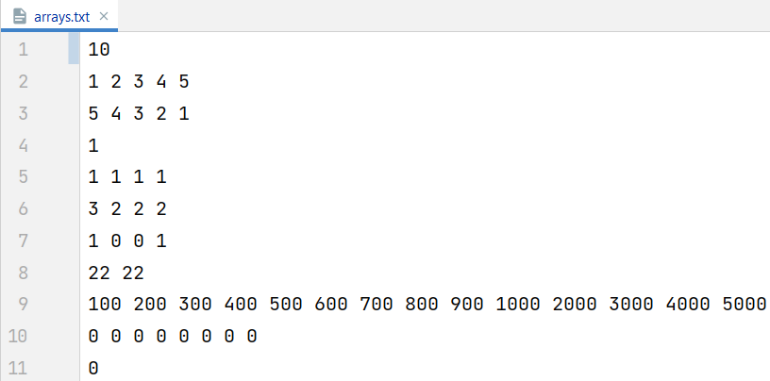
Листинг программы приведен в репозитории GitHub [task\_1](https://github.com/vyatsu-parallel-programming/lab_4/blob/master/internal/task_1/run.go).

**Задание 2**

«Фоновая обработка». Два потока обрабатывают файл, содержащий набор целочисленных массивов. В файле находится M массивов, каждый в отдельной строке. В первой строке указано число M – количество массивов, а в следующих M строках записаны элементы массивов, разделенные пробелами. Первый поток по очереди читает массивы из файла, а второй по очереди считает суммы элементов этих массивов. Результаты первый поток должен записать в новый файл. Каждую сумму – в отдельной строке. Организовать правильную работу потоков.

**Полученные результаты**

Входной файл:



Трассировка работы программы:

Изображение выглядит как стол

Автоматически созданное описание

Выходной файл:

Изображение выглядит как стол

Автоматически созданное описание

Листинг программы приведен в репозитории GitHub [task\_2](https://github.com/vyatsu-parallel-programming/lab_4/blob/master/internal/task_2/run.go).

**Задание 3**

«Читатели – писатели». Имеется хранилище данных, с которым работают одновременно несколько потоков. Первые N потоков случайным образом изменяют 2 данные в хранилище (при этом изменять данные в некоторый момент времени может ровно один поток), остальные M потоков периодически читают данные из хранилища. Несколько читателей могут находиться в хранилище одновременно, но читатель и писатель не могут одновременно использовать хранилище. Эмулировать работу хранилища.

**Полученные результаты**

Изображение выглядит как текст

Автоматически созданное описание

Листинг программы приведен в репозитории GitHub [task\_3](https://github.com/vyatsu-parallel-programming/lab_4/blob/master/internal/task_3/run.go).

Вывод

В ходе лабораторной работы я изучил классические задачи синхронизации, реализовал 3 программы на языке *go* и обеспечил их правильную работу с использованием примитивов синхронизации.