



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.01 ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ
ТЕХНИКА

О Т Ч Е Т

по лабораторной работе № 5

Название: Основы асинхронного программирования на Golang

Дисциплина: Языки интернет-программирования

Студент

ИУ6-32Б
(Группа)

(Подпись, дата)

В.А. Баринова
(И.О. Фамилия)

Преподаватель

(Подпись, дата)

В.Д. Шульман
(И.О. Фамилия)

Москва, 2024

Цель работы: изучение основ асинхронного программирования с использованием языка Golang.

Задание: в рамках данной лабораторной работы предлагается продолжить изучение Golang и познакомиться с продвинутыми конструкциями языка.

Ход работы

Задание 1.

Вам необходимо написать функцию calculator следующего вида:

```
func calculator(firstChan <-chan int, secondChan <-chan int, stopChan <-chan struct{}) <-chan int
```

Функция получает в качестве аргументов 3 канала, и возвращает канал типа <-chan int.

- в случае, если аргумент будет получен из канала firstChan, в выходной (возвращенный) канал вы должны отправить квадрат аргумента.
- в случае, если аргумент будет получен из канала secondChan, в выходной (возвращенный) канал вы должны отправить результат умножения аргумента на 3.
- в случае, если аргумент будет получен из канала stopChan, нужно просто завершить работу функции.

Функция calculator должна быть неблокирующей, сразу возвращая управление. Ваша функция получит всего одно значение в один из каналов - получили значение, обработали его, завершили работу.

После завершения работы необходимо освободить ресурсы, закрыв выходной канал, если вы этого не сделаете, то превысите предельное время работы.

Напишите программу. Тестируется через stdin → stdout

✓ Всё правильно.

Верно решили 1 684 учащихся
Из всех попыток 12% верных

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

```
1
2 func calculator(firstChan <-chan int, secondChan <-chan int, stopChan <-chan struct{}) <-chan int {
3     output := make(chan int)
4     go func(ch chan int) {
5         defer close(ch)
6
7         select {
8             case x := <-firstChan:
9                 ch <- x * x
10            case x := <-secondChan:
11                ch <- x * 3
12            case <-stopChan:
13            }
14        }(output)
15
16        return output
17    }
18
19 }
```

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рисунок 1—Прохождение тестов по заданию 1 на Stepik

Задание 2.

Напишите элемент конвейера (функцию), что запоминает предыдущее значение и отправляет значения на следующий этап конвейера только если оно отличается от того, что пришло ранее.

Ваша функция должна принимать два канала - `inputStream` и `outputStream`, в первый вы будете получать строки, во второй вы должны отправлять значения без повторов. В итоге в `outputStream` должны остаться значения, которые не повторяются подряд. Не забудьте закрыть канал ;)

Функция должна называться `removeDuplicates()`

Напишите программу. Тестируется через stdin → stdout

Верно. Так держать!

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

Верно решили 2 192 учащихся

Из всех попыток 20% верных

```
1 func removeDuplicates(inputStream, outputStream chan string) {
2     var str string
3     for value := range inputStream {
4         if value != str {
5             str = value
6             outputStream <- value
7         }
8     }
9     close(outputStream)
10 }
11
12
13
```

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рисунок 2—Прохождение тестов по заданию 2 на Stepik

Задание 3.

Внутри функции `main` (функцию объявлять не нужно), вам необходимо в отдельных горутинах вызвать функцию `work()` 10 раз и дождаться результатов выполнения вызванных функций.

Функция `work()` ничего не принимает и не возвращает. Пакет `"sync"` уже импортирован.

Напишите программу. Тестируется через stdin → stdout

Верно решили 2 050 учащихся
Из всех попыток 19% верных

✓ Здорово, всё верно.

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

```
1 var wg sync.WaitGroup
2
3 // Запускаем 10 горутин
4 for i := 0; i < 10; i++ {
5     wg.Add(1) // Увеличиваем счетчик горутин
6     go func() {
7         defer wg.Done() // Уменьшаем счетчик после завершения работы горутины
8         work()
9     }()
10 }
11
12 wg.Wait()
13
14
15
16
17
18
```

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рисунок 3—Прохождение тестов по заданию 3 на Stepik

Список использованных источников

1. <https://stepik.org/course/54403/info>