Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное 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 «Системы обработки информации и управления»

Домашнее задание
по дисциплине «Парадигмы и конструкции языков программирования»

Выполнил:
студент группы ИУ5-33Б Павлинич А. А.
подпись_____
дата: 03.11.2025

Проверил:
Гапанюк Ю. Е.
подпись_____
дата:
2025 г.

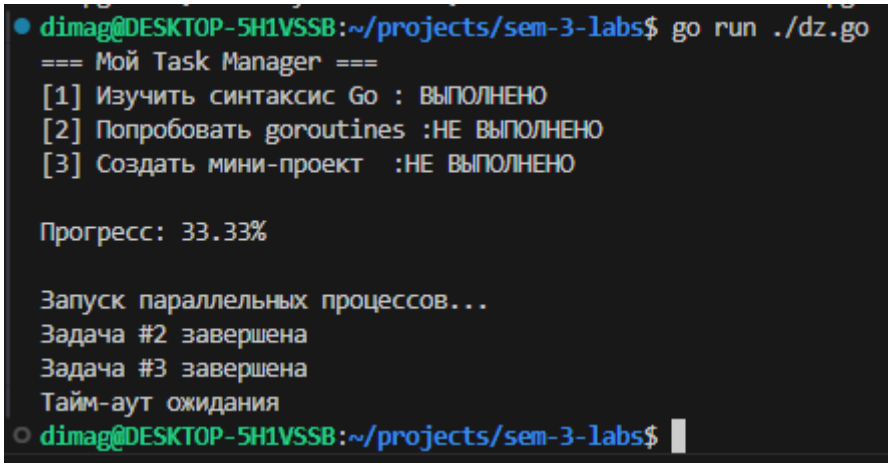
2025 г.

dz.go

```go
package main

import (
    "fmt"
    "time"
)

// Структура Task описывает задачу
type Task struct {
    ID       int
    Title    string
    Done     bool
}

// Метод для вывода задачи
func (t Task) String() string {
    status := ":НЕ ВЫПОЛНЕНО"
    if t.Done {
        status = ": ВЫПОЛНЕНО"
    }
    return fmt.Sprintf("[%d] %-20s %s", t.ID, t.Title, status)
}

// Функция добавления задач в список
func initTasks() []Task {
    return []Task{
        {ID: 1, Title: "Изучить синтаксис Go", Done: true},
        {ID: 2, Title: "Попробовать goroutines", Done: false},
        {ID: 3, Title: "Создать мини-проект", Done: false},
    }
}

// Рассчитать процент готовности
func calcProgress(tasks []Task) float64 {
    total := len(tasks)
    done := 0
    for _, t := range tasks {
        if t.Done {
            done++
        }
    }
    return float64(done) / float64(total) * 100
}

// Параллельная функция для отметки выполнения
func markAsDoneAsync(task *Task, ch chan<- string) {
    time.Sleep(time.Second) // имитация работы
    task.Done = true
```

```
    ch <- fmt.Sprintf("Задача #%d завершена", task.ID)
}

func main() {
    tasks := initTasks()

    fmt.Println("=== Мой Task Manager ===")
    for _, t := range tasks {
        fmt.Println(t)
    }

    fmt.Printf("\nПрогресс: %.2f%%\n\n", calcProgress(tasks))

    // Используем goroutines
    fmt.Println("Запуск параллельных процессов...")
    ch := make(chan string)

    for i := range tasks {
        if !tasks[i].Done {
            go markAsDoneAsync(&tasks[i], ch)
        }
    }

    // Получаем сообщения о завершении
    for range tasks {
        select {
        case msg := <-ch:
            fmt.Println(msg)
        case <-time.After(time.Second * 5):
            fmt.Println("Тайм-аут ожидания")
            return
        }
    }

    fmt.Println("\nОбновлённый список задач:")
    for _, t := range tasks {
        fmt.Println(t)
    }
    fmt.Printf("\nПрогресс: %.2f%%\n", calcProgress(tasks))
}
```

Вывод:


dimag@DESKTOP-5H1VSSB:~/projects/sem-3-labs$ go run ./dz.go
=== Мой Task Manager ===
[1] Изучить синтаксис Go : ВЫПОЛНЕНО
[2] Попробовать goroutines :НЕ ВЫПОЛНЕНО
[3] Создать мини-проект :НЕ ВЫПОЛНЕНО
Прогресс: 33.33%
Запуск параллельных процессов...
Задача #2 завершена
Задача #3 завершена
Тайм-аут ожидания
dimag@DESKTOP-5H1VSSB:~/projects/sem-3-labs$