МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт Искусственного Интеллекта
Базовая кафедра №252 «Информационная безопасность»

ОТЧЁТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

Тема практики: «Разработка высоконагруженных сервисов на Go»

приказ Университета о направлении на практику от «__» _____ 20__ г. № _____

Отчёт представлен к
рассмотрению:

Студент группы ККСО-02-20 «__» _____ 20__ г. _____ / Шинкарев М.С.
(подпись и расшифровка подписи)

Отчёт утверждён.
Допущен к защите:

Руководитель практики
от кафедры «__» _____ 20__ г. _____ / Васильев А.Ю.
(подпись и расшифровка подписи)

Москва 2022

МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт Искусственного Интеллекта
Базовая кафедра №252 «Информационная безопасность»

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ НА ПРОИЗВОДСТВЕННУЮ ПРАКТИКУ

Студенту 3 **курса учебной группы** ККСО-02-20

_____
(фамилия, имя и отчество)

Место и время проведения практики: базовая кафедра №252 «Информационная безопасность»

1. СОДЕРЖАНИЕ ПРАКТИКИ:

1.1. Изучить: _____

1.2. Практически выполнить: _____

1.3. Ознакомиться: _____

2. ДОПОЛНИТЕЛЬНОЕ ЗАДАНИЕ: _____

2. ОРГАНИЗАЦИОННО-МЕТОДИЧЕСКИЕ УКАЗАНИЯ: _____

_____

Руководитель практики от кафедры
«___» _____ 20___ г. _____ (Васильев А.Ю.)
(подпись) (ФИО)

Задание получил:
«___» _____ 20___ г. _____ (Шинкарев М. С.)
(подпись) (ФИО)

СОГЛАСОВАНО:
Заведующий БК № 252:
«___» _____ 20___ г. _____ (Корольков А.В.)
(подпись) (ФИО)

_____ (_____)
(Расшифровка, должность)

Москва 2022

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

РАБОЧИЙ ГРАФИК ПРОВЕДЕНИЯ ПРОИЗВОДСТВЕННОЙ ПРАКТИКИ

студента 3 курса группы ККСО-02-20 очной формы обучения,

обучающегося по направлению подготовки «Компьютерная безопасность»,
профиль «Анализ безопасности компьютерных систем»

Неделя	Сроки выполнения	Этап	Отметка о выполнении

Руководитель практики от кафедры _____ Васильев А.Ю., к.ф.-м.н.
(подпись) (ФИО, ученая степень, ученое звание)

Обучающийся _____ Шинкарев М. С.
(подпись) (ФИО, ученая степень, ученое звание)

Согласовано:
Заведующий БК №252 _____ Корольков А.В., с.н.с.
(подпись) (ФИО, ученая степень, ученое звание)

Москва 2022

Оглавление

История

В 2007 году Google начал разработку язык программирования Go в качестве внутреннего проекта. Язык создали Роб Пайк и Кен Томпсон. Оба являются выдающимися деятелями ИТ, а в прошлом – сотрудниками Bell Labs. Томпсон также один из создателей ОС UNIX и языка B (предшественника C). Два года спустя, 10 ноября 2009 года язык был анонсирован, а в марте 2012 года вышла версия 1.0. При этом язык продолжает развиваться. Текущей версией является версия 1.19, которая вышла в августе 2022 года.

Язык Go задумывали как универсальный язык, и с этой ролью он справляется, так как в его разработке пытались объединить скорость, типичную языков типа C и легкость написания, подобную Python. Однако Golang в основном используется в бэкенде.

Особенностью Go является его минимализм. В отличие от других языков, Golang находится на пути улучшения присутствующих инструментов, а не создания новых, поскольку Golang был создан с самого начала для идеального решения предоставленной ему задачи.

Назначение Go

Язык Go был разработан как язык программирования для создания высокоэффективных программ, работающих на современных распределённых системах и многоядерных процессорах. Это можно рассматривать как попытку создать альтернативу языкам C и C++, учитывающую, эволюцию компьютерных технологий и опыт, накопленный при разработке крупных систем. Вот что говорит Роб Пайк о назначении Go: «Целью Go не является проведение исследований в области дизайна языков программирования; она заключается в улучшении рабочей среды его разработчиков и их коллег. Go больше нацелен на разработку программ, нежели на исследования в области языков программирования. Или, другими словами, Go был спроектирован так, чтобы служить в первую очередь целям разработки».

Основные проблемы Google, подтолкнувшие к созданию Go:

- медленная сборка программ;
- неконтролируемые зависимости;
- использование разными программистами разных подмножеств языка;
- трудности с пониманием программ, вызванные нечитаемостью кода, плохой документацией и т. д;
- дублирование усилий;
- высокая стоимость обновлений;
- перекос версий (version skew);
- сложность разработки инструментария;
- кросс-языковые сборки.

Основными требованиями к языку стали:

- Go должен работать на больших масштабах, для крупных команд разработчиков, для программ с большим количеством зависимостей.
- Go должен быть знакомым, условно говоря, похожим на C. Google нужно, чтобы программисты могли быстро начать продуктивно

работать с новым языком, поэтому он не должен быть слишком радикальным.

- Go должен быть современным. Ему нужны такие возможности, как конкурентность, чтобы программисты могли эффективно использовать многоядерные машины. В нем должны быть встроенные библиотеки для работы с сетью и в качестве веб-сервера.

В результате выполнения этих требований и исправления ошибок получился язык, «который не стал прорывом, но тем не менее явился отличным инструментом для разработки крупных программных проектов».

Преимущества Go

Простота. Фактически, простота была основной целью создания языка, и это было реализовано. Go имеет довольно простой синтаксис (с некоторыми допущениями), поэтому приложения можно разрабатывать быстрее, чем на некоторых других языках.

Большое количество библиотек. Если вам недостаточно встроенных функций в Go, вы можете воспользоваться одной из множества библиотек и выполнить требуемую задачу.

Чистота кода. Чистота кода. Компилятор Go позволяет вам сохранять код читаемым и неперегруженным. К примеру, неиспользуемые переменные считаются ошибками компиляции. Большинство проблем с форматированием решаются в Go. Это делается, например, с помощью программы gofmt во время записи или компиляции.

Сборка мусора (Garbage collector). В отличие от C, в Go нет необходимости освобождать динамически выделяемую память и беспокоиться о висящих указателях (dangling pointers). Сборщик мусора сделает всё за вас.

Статическая типизация. Go – язык со статической типизацией. Это означает, что вы уже на этапе компиляции должны объявить типы всех переменных, аргументов функций и возвращаемых значений. Такой подход может показаться неприятным, но на самом деле благодаря этому обнаруживается множество ошибок уже во время компиляции.

Конкурентность. Go предлагает первоклассную поддержку конкурентности, что является одним из важнейших преимуществ этого

языка программирования. Механизмы реализации конкурентности Go основаны на работе Тони Хоара (Tony Hoare) «Communicating Sequential Processes».

GoDoc. Утилита, которая упрощает документирование кода. Большим плюсом GoDoc является то, что здесь не используются дополнительные языки, такие как JavaDoc, PHPDoc или JSDoc. Утилита использует максимальный объем информации, который она извлекает из документируемого кода.

Переносимость. Поскольку программа компилируется напрямую в машинный код, следовательно, бинарные файлы являются переносимыми среди одной операционной системы и архитектуры.

Типы данных в Go

Все данные, хранящиеся в памяти, по сути, представляют собой просто набор битов. И именно тип данных определяет, как эти данные интерпретируются и какие операции с ними можно выполнять.

Рассмотрим базовые встроенные типы данных, которые мы можем использовать.

Целочисленные типы

int8: представляет целое число от -128 до 127 и занимает в памяти 1 байт (8 бит).

int16: представляет целое число от -32768 до 32767 и занимает в памяти 2 байта (16 бит).

int32: представляет целое число от -2147483648 до 2147483647 и занимает 4 байта (32 бита).

int64: представляет целое число от –9 223 372 036 854 775 808 до 9 223 372 036 854 775 807 и занимает 8 байт (64 бита).

uint8: представляет целое число от 0 до 255 и занимает 1 байт.

uint16: представляет целое число от 0 до 65535 и занимает 2 байта.

uint32: представляет целое число от 0 до 4294967295 и занимает 4 байта.

uint64: представляет целое число от 0 до 18 446 744 073 709 551 615 и занимает 8 байт.

byte: синоним типа uint8, представляет целое число от 0 до 255 и занимает 1 байт.

rune: синоним типа int32, представляет целое число от -2147483648 до 2147483647 и занимает 4 байта.

int: представляет целое число со знаком, которое в зависимости о платформы может занимать либо 4 байта, либо 8 байт. То есть соответствовать либо int32, либо int64.

uint: представляет целое беззнаковое число только без знака, которое, аналогично типу int, в зависимости о платформы может занимать либо 4 байта, либо 8 байт. То есть соответствовать либо uint32, либо uint64.

Числа с плавающей точкой

float32: представляет число с плавающей точкой от $1.4*10^{-45}$ до $3.4*10^{38}$(для положительных). Занимает в памяти 4 байта (32 бита)

float64: представляет число с плавающей точкой от $4.9*10^{-324}$ до $1.8*10^{308}$ (для положительных) и занимает 8 байт.

Комплексные числа

complex64: комплексное число, где вещественная и мнимая части представляют числа float32

complex128: комплексное число, где вещественная и мнимая части представляют числа float64

Тип bool

Логический тип или тип bool может иметь одно из двух значений: **true** (истина) или **false** (ложь).

Строки

Строки представлены типом string. В Go строке соответствует строковый литерал - последовательность символов, заключенная в двойные кавычки.

Кроме обычных символов строка может содержать специальные последовательности (управляющие последовательности), которые начинаются с обратного слеша \. Наиболее распространенные последовательности:

\n: переход на новую строку

\r: возврат каретки

\t: табуляция

\": двойная кавычка внутри строк

\\: обратный слеш

Значение по умолчанию

Если переменной не присвоено значение, то она имеет значение по умолчанию, которое определено для ее типа. Для числовых типов это

число 0, для логического типа - false, для строк - "" (пустая строка).

Примеры программ на Go

1) Программа calculator.go представляет собой реализацию простейшего калькулятора на языке программирования Go. Выражение, которое необходимо посчитать, указывается в аргументах командной строки в виде: -first=<первое число> -oper=<операция> -second=<второе число>. Код программы приведен ниже:

```
package main

import (
	"flag"
	"fmt"
)

func main() {

	var firstNumber, secondNumber float64
	var operation string

	flag.Float64Var(&firstNumber, "first", 0, "Ваше первое число")
	flag.Float64Var(&secondNumber, "second", 1, "Ваше второе число")
	flag.StringVar(&operation, "oper", "", "Ваша операция (+, -, *, /)")
	flag.Parse()


	switch operation {
	case "+":
		fmt.Print(firstNumber, operation, secondNumber, "=", firstNumber+secondNumber)

	case "-":
		fmt.Print(firstNumber, operation, secondNumber, "=", firstNumber-secondNumber)

	case "*":
		fmt.Print(firstNumber, operation, secondNumber, "=", firstNumber*secondNumber)

	case "/":
		if secondNumber == 0 {
			fmt.Println("Division by zero!")
		} else {
			fmt.Print(firstNumber, operation, secondNumber, "=", firstNumber/secondNumber)
		}

	default:
		fmt.Println("No such operation")
	}

}
```

Листинг 1 – calculator.go

2) Программа BMI.go представляет собой реализацию калькулятора индекса массы тела на языке программирования Go. В качестве аргументов командной строки указывается масса в килограммах и рост в метрах в виде: -weight=<масса> -height=<рост>. Код программы приведен ниже:

```
package main

import (
	"flag"
	"fmt"
)

func main() {
	var weight, height float64
	flag.Float64Var(&weight, "weight", 1.0, "Введите ваш вес в килограммах")
	flag.Float64Var(&height, "height", 1.0, "Введите ваш рост в метрах")
	flag.Parse()

	var BMI = weight / (height * height)

	if BMI < 16 {
		fmt.Println("Выраженный дефицит массы тела")
	} else if BMI >= 16 && BMI < 18.5 {
		fmt.Println("Недостаточная масса тела")
	} else if BMI >= 18.5 && BMI < 24.99 {
		fmt.Println("Норма")
	} else if BMI >= 25 && BMI < 30 {
		fmt.Println("Избыточная масса тела")
	} else if BMI >= 30 && BMI < 35 {
		fmt.Println("Ожирение первой степени")
	} else if BMI >= 35 && BMI < 40 {
		fmt.Println("Ожирение второй степени")
	} else {
		fmt.Println("Ожирение третьей степени")
	}
}
```

Листинг 2 – BMI.go

Вывод

В ходе данной работы были получены базовые знания о языке программирования Go и были написаны простейшие программы с использованием различных условных операторов, аргументов командной строки. Так же были начаты несколько курсов по изучению языка, такие как A Tour Of Go на сайте https://go.dev/tour/welcome/1, и Go for Beginners на сайте https://hyperskill.org/tracks/25.

Список литературы

1. https://go.dev/tour/welcome/1
2. https://hyperskill.org/tracks/25
3. https://talks.golang.org/2012/splash.article
4. https://habr.com/ru/company/vk/blog/446914/
5. https://metanit.com/go/tutorial/2.3.php
6. https://ru.wikipedia.org/wiki/Go
7. https://golangify.com/go-beginning