

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций
Отчет по лабораторной работе № 1
«Основы языка программирования Go»
по дисциплине «Программная инженерия»**

Выполнил студент группы

ПИЖ-б-о-21-1

Прокопов Дмитрий

« » _____ 20__ г.

Подпись студента _____

Работа защищена

« » _____ 20__ г.

Проверил: Воронкин Р.А. _____

(подпись)

Цель работы: Исследовать назначение и способы установки Go, исследовать типы данных, константы и арифметические операции языка программирования Go.

ВЫПОЛНЕНИЕ

```
-go main.go > ...
1  package main
2
3  import "fmt"
4
5  func main() {
6      fmt.Println("Привет мир")
7
8      var a string
9      fmt.Scan(&a)
10 }
11
```

Рисунок 1.1 – пример кода на GO

```
PS C:\Users\dmidt\go> go run main.go
Привет мир
^Z
```

Рисунок 1.2 – выполнение кода

```
PS C:\Users\dmidt\go> go build main.go
```

Рисунок 1.3 – создание программы

```
C:\Users\dmidt\go\main.exe
Привет мир
```

Рисунок 1.4 – выполнение программы

```

package main

import "fmt"

func main() {
    var hello string
    hello = "Hello Go!"
    var a int = 2019
    fmt.Println(hello)
    fmt.Println(a)
}

```

Рисунок 1.5 – код примера

```

PS C:\Users\dmidt\go> go run main.go
Hello Go!
2019

```

Рисунок 1.6 – выполнение примера

```

package main

import "fmt"

func main() {
    var symbol int32 = 'c'
    fmt.Println(string(symbol))
}

```

Рисунок 1.7 – код примера

```

PS C:\Users\dmidt\go> go run main.go
c

```

Рисунок 1.8 – выполнение примера

```

package main

import "fmt"

func main() {
    var a int32 = 5
    b := 4
    fmt.Println(b)
    fmt.Println(a)
}

```

Рисунок 1.9 – код примера

```

package main

import "fmt"

func main() {
    var (
        b string = "Ivan"
        a int     = 14
    )
    fmt.Println(b)
    fmt.Println(a)
}

```

Рисунок 1.10 – код примера

```

package main

import "fmt"

func main() {
    var (
        name string
        age  int
    )
    fmt.Print("Введите имя: ")
    fmt.Scan(&name)
    fmt.Print("Введите возраст: ")
    fmt.Scan(&age)

    fmt.Println(name, age)
}

```

Рисунок 1.11 – код примера

```

PS C:\Users\dmidt\go> go run main.go
Введите имя: Иван
Введите возраст: 65
Иван 65

```

Рисунок 1.12 – выполнение примера

```

package main

import "fmt"

func main() {
    name := "Ivan"
    age := 24
    fmt.Println("My name is", name, "and I am", age,
        "years old.")
}

```

Рисунок 1.13 – код примера

```
PS C:\Users\dmidt\go> go run main.go
My name is Ivan and I am 24 years old.
```

Рисунок 1.14 – выполнение примера

```
package main

import "fmt"

const (
    a int = 45
    b
    c float32 = 3.3
    d
)

func main() {
    fmt.Println(a, b, c, d)
}
```

Рисунок 1.15 – код примера

```
PS C:\Users\dmidt\go> go run main.go
45 45 3.3 3.3
```

Рисунок 1.16 – выполнение примера

```
import "fmt"

const (
    Sunday = iota
    Monday
    Tuesday
    Wednesday
    Thursday
    Friday
    Saturday
)

func main() {
    fmt.Println(Sunday)
    fmt.Println(Saturday)
}
```

Рисунок 1.17 – код примера

```
PS C:\Users\dmidt\go> go run main.go
0
6
```

Рисунок 1.18 – выполнение примера

```
package main

import "fmt"

func main() {
    ⚡ fmt.Print("I like Go!")
}
```

Рисунок 1.19 – код 1-го задания

```
PS C:\Users\dmidt\go> go run main.go
I like Go!
```

Рисунок 1.20 – выполнение кода

```
package main

import "fmt"

func main() {
    for i := 0; i < 3; i++ {
        fmt.Println("I like Go!")
    }
}
```

Рисунок 1.21 – код 2-го задания

```
PS C:\Users\dmidt\go> go run main.go
I like Go!
I like Go!
I like Go!
```

Рисунок 1.22 – выполнение кода

```
package main

import "fmt"

func main() {
    var n uint
    fmt.Print("Введите число: ")
    fmt.Scan(&n)
    fmt.Println(n*2 + 100)
}
```

Рисунок 1.23 – код 3-го задания

```
PS C:\Users\dmidt\go> go run main.go
Введите число: 9
118
```

Рисунок 1.24 – выполнение кода

```
package main

import (
    "fmt"
)

func main() {
    var a, b int
    fmt.Print("Введите первое число: ")
    fmt.Scan(&a)
    fmt.Print("Введите второе число: ")
    fmt.Scan(&b)
    fmt.Println(a*a + b*b)
}
```

Рисунок 1.25 – код 4-го задания

```
PS C:\Users\dmidt\go> go run main.go
Введите первое число: 7
Введите второе число: 3
58
```

Рисунок 1.26 – выполнение кода

```
package main

import "fmt"

func main() {
    var n uint
    fmt.Print("Введите число: ")
    fmt.Scan(&n)
    fmt.Println(n * n)
}
```

Рисунок 1.27 – код 5-го задания

```
PS C:\Users\dmidt\go> go run main.go
Введите число: 9
81
```

Рисунок 1.28 – выполнение кода

```

package main

import (
    "fmt"
)

func main() {
    var a int
    fmt.Print("Введите число: ")
    fmt.Scan(&a)
    fmt.Println(a % 10)
}

```

Рисунок 1.29 – код 6-го задания

```

PS C:\Users\dmidt\go> go run main.go
Введите число: 7352
2


```

Рисунок 1.30 – выполнение кода

```

package main

import (
    "fmt"
)

func main() {
    var a int
    fmt.Print("Введите число: ")
    fmt.Scan(&a)
     fmt.Println(a % 100 / 10)
}

```

Рисунок 1.31 – код 7-го задания

```

PS C:\Users\dmidt\go> go run main.go
Введите число: 7352
5

```

Рисунок 1.32 – выполнение кода


```

package main

import (
    "fmt"
)

func main() {
    var a uint8
    fmt.Print("Введите число от 1 до 359: ")
    fmt.Scan(&a)
    fmt.Println("It is ", a*2/60, "hours", a*2%60, "minutes")
}

```

Рисунок 1.33 – код 8-го задания

```

PS C:\Users\dmidt\go> go run main.go
Введите число от 1 до 359: 90
It is  3 hours 0 minutes

```

Рисунок 1.34 – выполнение кода

```

package main

import "fmt"

func main() {

    var a2 int = 10

    a2 = a2 * 10
    fmt.Println(a2)
}

```

Рисунок 1.35 – код 9-го задания

```

PS C:\Users\dmidt\go> go run main.go
100

```

Рисунок 1.36 – выполнение кода

```

package main

import "fmt"

func main() {
    var a int = 8
    var b int = 10
    a = a + b
    b = b + a
    fmt.Println(a)
}

```

Рисунок 1.37 – код 10-го задания

```

PS C:\Users\dmidt\go> go run main.go
18

```

Рисунок 1.38 – выполнение кода

```

package main

import (
    "fmt"
    "math"
)

func main() {
    var a, b float64
    fmt.Print("Введите значение a: ")
    fmt.Scan(&a)
    fmt.Print("Введите значение b: ")
    fmt.Scan(&b)
    s := 2 * math.Pi * a * b
    v := (4 / 3) * math.Pi * a * b * b
    fmt.Printf("Площадь = %1.2f \nОбъем = %1.2f", s, v)
}

```

Рисунок 1.39 – код 11-го задания

```

PS C:\Users\dmidt\go> go run main.go
Введите значение a: 30
Введите значение b: 55
Площадь = 10367.26
Объем = 285099.53

```

Рисунок 1.40 – выполнение кода

```

package main

import "fmt"

func main() {
    var f, n, a int
    var t int = 10
    fmt.Print("Введите силу: ")
    fmt.Scan(&f)
    fmt.Print("Введите расстояние: ")
    fmt.Scan(&n)
    a = f * n
    fmt.Println("Работа: ", a)
    fmt.Println("Мощность: ", a/t)
}

```

Рисунок 1.41 – индивидуальное задание №1 (7.Работа и мощность)

```

PS C:\Users\dmidt\go> go run main.go
Введите силу: 9
Введите расстояние: 40
Работа: 360
Мощность: 36

```

Рисунок 1.42 – выполнение задания

```

package main

import (
    "fmt"
    "math"
)

func main() {
    var a, b, c, d, h float64
    fmt.Print("1-ое основание: ")
    fmt.Scan(&a)
    fmt.Print("2-ое основание: ")
    fmt.Scan(&b)
    fmt.Print("Высота: ")
    fmt.Scan(&h)
    d = (b - a) / 2
    c = float64(math.Sqrt(h*h + d*d))
    fmt.Printf("%.2f", a+b+c*2)
}

```

Рисунок 1.43 – индивидуальное задание №2 (7-е задание)

```

PS C:\Users\dmidt\go> go run main.go
1-ое основание: 45
2-ое основание: 80
Высота: 14
169.82

```

Рисунок 1.44 – выполнение задания

Вывод: Изучены основы работы с языком программирования Go, а именно: установка и настройка среды для программирования на этом языке, типы данных, работа с переменными и арифметическими операциями, встроенные функции, а также на основе полученных знания написаны программы.

Ответы на вопросы:

1. Как объявить переменную типа int в Go?

```
var name int;
```

```
name := 10 или
```

```
var name int = 10
```

2. Какое значение по умолчанию присваивается переменной типа int в Go?

По умолчанию значение переменной int в Go: 0.

3. Как изменить значение существующей переменной в Go?

```
var x int = 10
```

x = 20

4. Что такое множественное объявление переменных в Go?

Язык Go позволяет объявление нескольких переменных сразу, например, инициализация трех переменных одного типа может выглядеть следующим образом:

1) в несколько строк:

```
var a, b, c int // Объявление трех переменных типа int
```

```
a = 1
```

```
b = 2
```

```
c = 3
```

2) в одну строку:

```
var x, y, z = 4, 5, "text"
```

Здесь же идет и объявление и инициализация трех переменных разных типов сразу: x типа int, y типа int и z типа string.

5. Как объявить константу в Go?

Для определения констант применяется ключевое слово const:

```
const pi float64 = 3.1415
```

Константы, как и обычные переменные, можно объявлять в блоке:

```
const (
```

```
    a int = 45
```

```
    b float32 = 3.3
```

```
)
```

или

```
const(
```

```
    A int = 45
```

```
    B
```

```
    C float32 = 3.3
```

```
    D
```

```
)
```

6. Можно ли изменять значение константы после её объявления в Go?
Мы не можем менять значение константы после объявления.

7. Какие арифметические операторы поддерживаются в Go?

В языке Go поддерживаются операторы:

- Сложение (+)
- Вычитание (-)
- Умножение (*)
- Деление (/)
- Остаток от деления (%)
- Инкремент (++)
- Декремент (--)

8. Какой оператор используется для выполнения операции остатка в Go?

В Go для выполнения операции остатка от деления используется оператор %.

9. Какой результат выражения $5 / 2$ в Go?

Результатом выражения будет 2, так как мы получим только целую часть ответа (2.5).

10. Как считать строку с консоли в Go?

Считать строку с консоли можно методом `fmt.Scan(&a)`, где `&a` – ссылка (более точно - адрес) на переменную `a`. Если проще, то введённое число запишется из консоли напрямую в эту переменную и там будет храниться, пока не понадобится её куда-нибудь пристроить или изменить.

Также можно читать с консоли сразу несколько переменных: `fmt.Scan(&a, &b, &c)`.

11. Как считать целое число с консоли в Go?

Необходимо сначала объявить переменную для целого числа до считывания, например, `var num int`. Затем, воспользоваться либо функцией

`fmt.Scan(&num)`, либо `fmt.Scanf("%d", &num)`, где `%d` целые числа в десятичном формате.

Основное различие между `Scan` и `Scanf` заключается в том, что `Scan` используется для сканирования строки без явного форматирования, в то время как `Scanf` используется для сканирования ввода с определенным форматом.

12. Как обработать ошибку при считывании данных с консоли в Go?

Обработать ошибку можно несколькими способами.

Самым простым способом является:

```
var input string
// Считываем строку с консоли
if _, err := fmt.Scan(&input); err != nil {
    fmt.Println("Произошла ошибка:", err)
    return
}
// Выводим данные, если считывание прошло успешно
fmt.Println("Вы ввели:", input)
```

`_` – результат сканирования, `err` – переменная, в которой будет храниться ошибка, если она произойдет во время операции сканирования.

`err != nil` – проверка на наличие ошибки. Если переменная `err` не равна `nil`, значит, произошла ошибка (`nil` используется для представления нулевого значения – `null`).

Более сложным способом обработки ошибки является использование стандартного пакета «errors».

13. Как вывести строку в консоль в Go?

Для вывода данных на консоль мы на данном этапе можно использовать методы, которые присутствуют в пакете `fmt` – `Print()` и `Println()`, например, `fmt.Print("hello, world")`.

Первый метод при выводе нескольких объектов вставляет между ними пробелы, если среди них нет строк.

Второй всегда ставит пробелы между выводимыми объектами, плюс добавляет новую строку. То есть он пригодится, если нам необходимо будет сделать вывод на нескольких строках.

Можно выводить и несколько объектов:

```
fmt.Print("Ivan", 27) // Ivan27
fmt.Println("Ivan", 27) // Ivan27
fmt.Println("My name is", name, "and I am", age, "years old.")
```

14. Как вывести значение переменной типа `int` в консоль?

```
var number int = 42
fmt.Println(number)
или
fmt.Printf("%d", number)
```

15. Как форматировать вывод числа с плавающей точкой в Go?

В Go для форматирования вывода числа с плавающей точкой вы можете использовать спецификатор формата «%f».

```
var value float64 = 3.14159
fmt.Printf("%.2f", value)
```

Выводом будет 3.14.

16. Как объявить переменную типа `byte` и присвоить ей значение 65?

Чем отличается оператор `:=` от оператора `=` в Go?

Переменную типа `byte` можно объявить так: `var byteValue byte = 65`

Оператор «`:=`» позволяет одновременно объявить переменную и

присвоить ей значение. Он используется внутри функций для создания новых переменных.

Оператор «=» просто присваивает значение переменной.

Он используется для изменения значения существующей переменной.

17. Какие типы данных можно использовать для представления чисел с плавающей точкой в Go?

float32 – 32-битное представление числа с плавающей точкой одинарной точности.

float64 (или просто float) – 64-битное представление числа с плавающей точкой двойной точности.

Обычно для большинства ситуаций используется тип float64, так как он обеспечивает большую точность по сравнению с float32, хотя может занимать больше памяти.

18. Как объявить и использовать несколько переменных в Go?

1) Используя оператор var: var x, y, z int

2) Используя оператор := для краткого объявления переменных: x, y, z := 10, 20, 0

3) Объявление с использованием разных типов

```
данных: var (  
    name string  
    age int  
    height float64  
)
```

4) Множественное

присваивание x, y := 10, 20

x, y = y, x // Обмен значениями переменных без использования временной переменной

Переменным можно присваивать значения, использовать их в выражениях и операциях, а также выводить их значения или использовать их в других частях программы.