

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ  
ФЕДЕРАЦИИ**

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

**Кафедра инфокоммуникаций**

**«Основы программирования Go»**

**Отчет по лабораторной работе №1  
по дисциплине «Программная инженерия»**

Выполнил студент группы ПИЖ-б-о-21-1  
Халимендик Я. Д. « » 2024г.

Подпись студента \_\_\_\_\_

Работа защищена « » \_\_\_\_\_ 2024г.

Проверила Воронкин Р.А. \_\_\_\_\_  
(подпись)

Ставрополь 2024

Цель: исследование назначения и способов установки Go, исследование типов данных, констант и арифметических операций языка программирования Go.

Ход работы:

Пример 1. Вывода строки «Hello, Go!»

Листинг:

```
package main
import "fmt"
func main() {
    fmt.Printf("Hello, Go!")
}
```

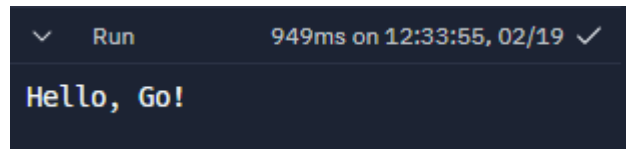


Рисунок 1 – Результат выполнения

Пример 2. Особенности вывода символов

Листинг:

```
package main
import "fmt"
func main() {
    fmt.Println("Hello Go"[0])
}

////////////////////////////////////

package main
import "fmt"
func main() {
    fmt.Println(string("Hello Go"[0]))
}
```

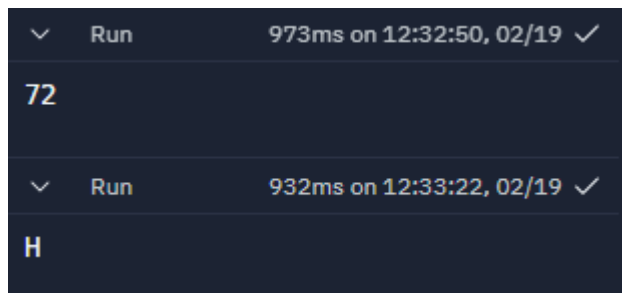


Рисунок 2 – Результат выполнения

### Пример 3. Переменные

Листинг:

```
package main
```

```
import "fmt"
```

```
func main() {  
    var hello string  
    hello = "Hello Go!"  
    var a int = 2024  
    fmt.Println(hello)  
    fmt.Println(a)  
}
```

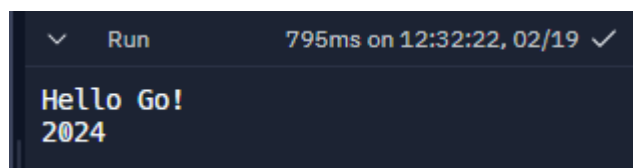


Рисунок 3 – Результат выполнения

### Пример 4. Переменные

Листинг:

```
package main
```

```

import "fmt"

func main() {
    var (
        name string = "Yana"
        age int = 20
    )
    fmt.Println(name)
    fmt.Println(age)
}

```

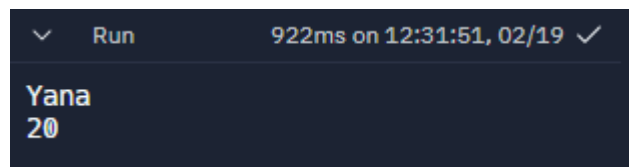


Рисунок 4 – Результат выполнения

## Пример 5. Арифметические операции

Листинг:

```

package main

import "fmt"

func main() {
    a := 100
    b := 10
    fmt.Println(a + b) // c = 110,
    fmt.Println(a * b) // c = 1000,
    fmt.Println(a - b) // c = 90,
    fmt.Println(a / b) // c = 10,
    var e int = 1
}

```

```

e++
fmt.Println(e)
var t int = 10
t--
fmt.Println(t)
}

```

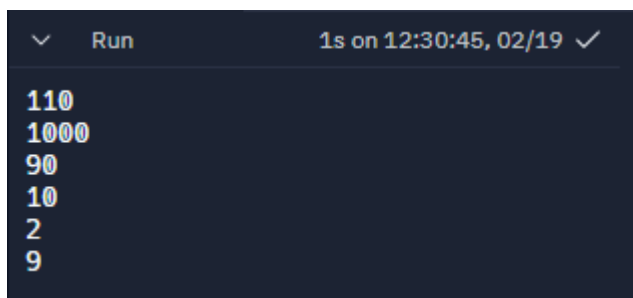


Рисунок 5 – Результат выполнения

#### Пример 6. Ввод данных

Листинг:

```

package main

import "fmt"

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

    fmt.Println(name, age)
}

```

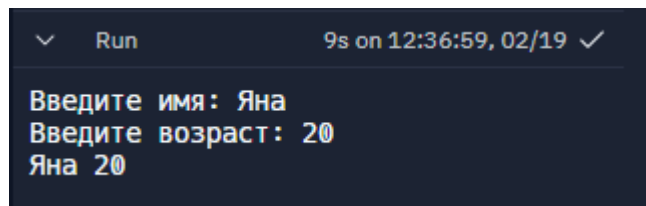
A terminal window with a dark background. At the top, it shows a dropdown arrow, the word 'Run', and a status bar indicating '9s on 12:36:59, 02/19' with a checkmark. The main area contains three lines of text: 'Введите имя: Яна', 'Введите возраст: 20', and 'Яна 20'.

Рисунок 6 – Результат выполнения

Пример 7. Вывод на консоль

Листинг:

```
package main
```

```
import "fmt"
```

```
func main() {
```

```
    fmt.Print("hello, world")
```

```
    fmt.Print("hello, world")
```

```
    fmt.Println("\n")
```

```
    fmt.Println("hello, world \n")
```

```
    fmt.Print("hello, world")
```

```
    fmt.Println("\n")
```

```
    fmt.Print("Ivan", 27)
```

```
    fmt.Println("Ivan", 27)
```

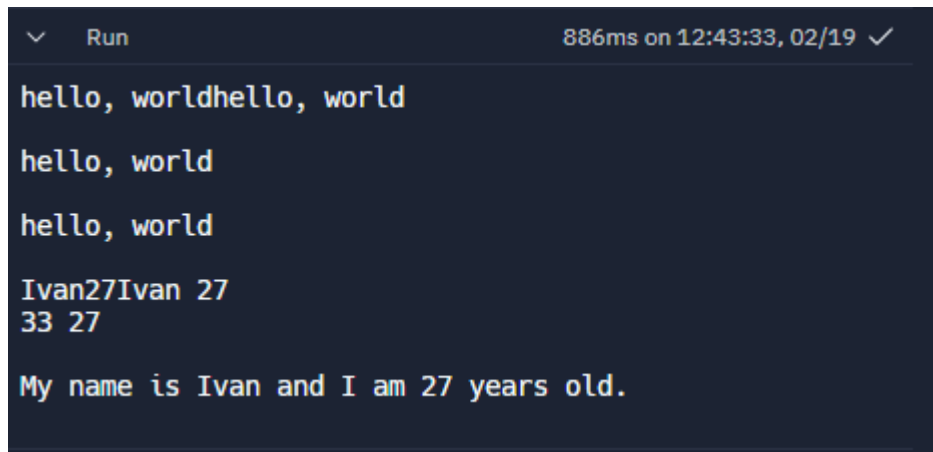
```
    fmt.Print(33, 27)
```

```
    fmt.Println("\n")
```

```

name := "Ivan"
age := 27
fmt.Println("My name is", name, "and I am", age, "years old.")
}

```



The screenshot shows a terminal window with a dark background. At the top, it says 'Run' and '886ms on 12:43:33, 02/19'. The output of the program is as follows:

```

hello, worldhello, world
hello, world
hello, world
Ivan27Ivan 27
33 27
My name is Ivan and I am 27 years old.

```

Рисунок 7 – Результат выполнения

## Пример 8. Комментарии

Листинг:

```
/*
```

Первая программа

на языке Go

```
*/
```

```
package main // определение пакета для текущего файла
```

```
import "fmt" // подключение пакета fmt
```

```
// определение функции main
```

```
func main() {
```

```
    fmt.Println("Hello Go!") // вывод строки на консоль
}
```

```
}
```

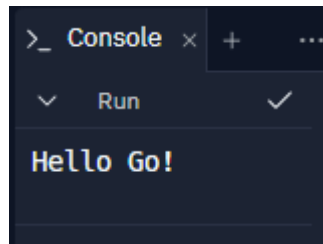


Рисунок 8 – Результат выполнения

### Пример 9. Константы

Листинг:

```
package main
```

```
import (  
    "fmt"  
)
```

```
const(  
    A int = 45  
    B  
    C float32 = 3.3  
    D  
)
```

```
func main() {  
    fmt.Println(A, B, C, D) }
```

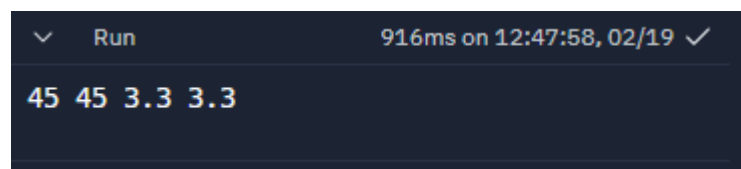


Рисунок 9 – Результат выполнения

Пример 10.



Листинг:

```
package main
```

```
import (
```

```
    "fmt"
```

```
)
```

```
const (
```

```
    Sunday = 0
```

```
    Monday = 1
```

```
    Tuesday = 2
```

```
    Wednesday = 3
```

```
    Thursday = 4
```

```
    Friday = 5
```

```
    Saturday = 6
```

```
)
```

```
func main() {
```

```
    fmt.Println(Sunday) // ВЫВОД 0
```

```
    fmt.Println(Saturday) // ВЫВОД 6
```

```
}
```

```
////////////////////////////////////
```

```
package main
```

```
import (
```

```
    "fmt"
```

```
)
```

```
const (
```

```

    Sunday = iota
    Monday
    Tuesday
    Wednesday
    Thursday
    Friday
    Saturday
    _
    Add
)

func main() {
    fmt.Println(Sunday) // ВЫВОД 0
    fmt.Println(Saturday) // ВЫВОД 6
}

```

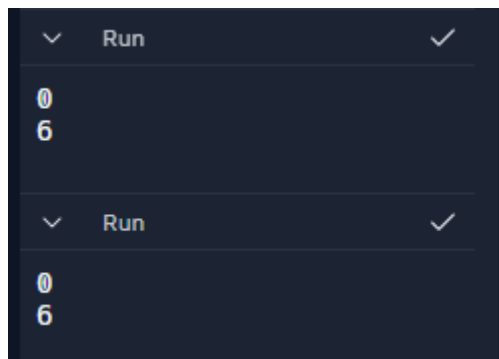


Рисунок 10 – Результат выполнения

## Пример 11. Константы

Листинг:

```
package main
```

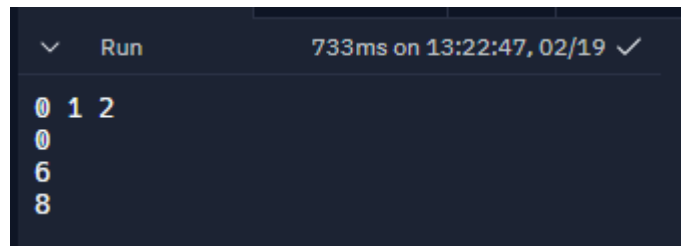
```
import (
    "fmt"

```

)

```
func main() {  
    const (  
        c0 = iota // c0 == 0  
        c1 = iota // c1 == 1  
        c2 = iota // c2 == 2  
    )  
    fmt.Println(c0, c1, c2) // ВЫВОД: 0 1 2  
    const (  
        Sunday = iota  
        Monday  
        Tuesday  
        Wednesday  
        Thursday  
        Friday  
        Saturday  
        _ // пропускаем 7  
        Add  
    )  
    fmt.Println(Sunday) // ВЫВОД: 0  
    fmt.Println(Saturday) // ВЫВОД: 6  
    fmt.Println(Add) // ВЫВОД: 8  
    const (  
        u = iota * 42 // u == 0 (индекс - 0, поэтому 0 * 42 = 0)  
        v float64 = iota * 42 // v == 42.0 (индекс - 1, поэтому 1.0 * 42 = 42.0)  
        w = iota * 42 // w == 84 (индекс - 2, поэтому 2 * 42 = 84)  
    )  
    // переменные ни в одном блоке const, поэтому индекс не увеличился
```

```
const x = iota // x == 0
const y = iota // y == 0
}
```

A screenshot of a Go program's execution output. At the top, it shows 'Run' and a duration of '733ms on 13:22:47, 02/19' with a checkmark. Below this, the output is displayed on a dark background with light blue text. The first line is '0 1 2', followed by '0', '6', and '8' on subsequent lines, representing the values of x and y for each iteration of iota.

```
0 1 2
0
6
8
```

Рисунок 11 – Результат выполнения

Пример 12.

Листинг:

```
package main
```

```
import (
    "fmt"
    "math" // Add this line to import the math package
)
```

```
func main() {
    var result = math.Abs(-5.67) //абсолютное значение
    fmt.Println(result)
```

```
    result = math.Ceil(5.67) // округление вверх до ближайшего целого
    fmt.Println(result)
```

```
    result = math.Floor(5.67) // округление вниз до ближайшего целого
    fmt.Println(result)
```

```
    result = math.Sqrt(16) // квадратный корень
    fmt.Println(result)
```

```
result = math.Pow(2, 3) // возведение в степень  
fmt.Println(result)
```

```
var sinValue = math.Sin(math.Pi / 2) // синус  
fmt.Println(sinValue)
```

```
result = math.Log(10) // логарифм (натуральный)  
fmt.Println(result)
```

```
var maxVal = math.Max(3, 7) // максимальное значение  
var minVal = math.Min(3, 7) // минимальное значение  
fmt.Println(maxVal, minVal)
```

```
result = math.Mod(10, 3) // остаток от деления  
fmt.Println(result)
```

```
result = math.Round(5.67) // округление до ближайшего целого  
fmt.Println(result)
```

```
var posInf = math.Inf(1) // Возвращает положительную бесконечность  
var negInf = math.Inf(-1) // Возвращает отрицательную бесконечность  
fmt.Println(posInf, negInf)
```

```
var nan = math.NaN() // Возвращает "Not a Number" (NaN)  
fmt.Println(nan)
```

```
result = math.Exp(2) // Возвращает экспоненту в степени  
fmt.Println(result)
```

```
result = math.Exp2(3) // Возвращает экспоненту в степени 2  
fmt.Println(result)
```

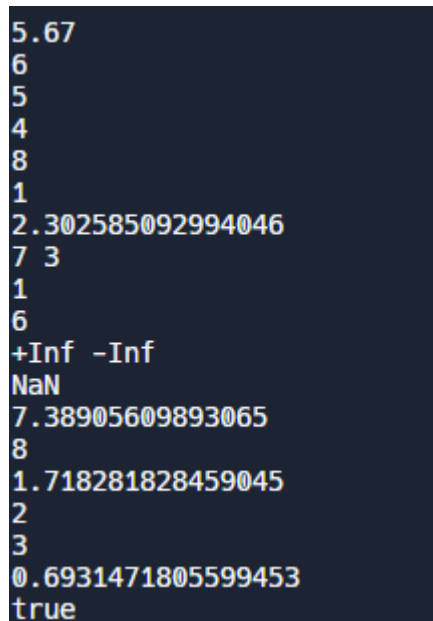
```
result = math.Expm1(1) // Возвращает экспоненту в степени -1  
fmt.Println(result)
```

```
result = math.Log10(100) // Возвращает десятичный логарифм числа  
fmt.Println(result)
```

```
result = math.Log2(8) // Возвращает двоичный логарифм числа  
fmt.Println(result)
```

```
result = math.Log1p(1) // Возвращает натуральный логарифм числа  
fmt.Println(result)
```

```
var isNegative = math.Signbit(-5) // Возвращает true, если число  
отрицательное  
fmt.Println(isNegative)  
}
```



```
5.67
6
5
4
8
1
2.302585092994046
7 3
1
6
+Inf -Inf
NaN
7.38905609893065
8
1.718281828459045
2
3
0.6931471805599453
true
```

Рисунок 12 – Результат выполнения

Практическая часть.

Задача 1: Напишите программу, которая выводит "I like Go!"

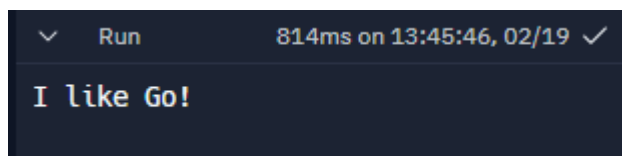
Листинг:

// Задача: Напишите программу, которая выводит "I like Go!"

```
package main
```

```
import "fmt"
```

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



```
Run 814ms on 13:45:46, 02/19 ✓
I like Go!
```

Рисунок 13 – Результат выполнения

Задача 2: Напишите программу, которая выведет "I like Go!" 3 раза.

Листинг:

// Задача: Напишите программу, которая выведет "I like Go!" 3 раза

```
package main
```

```
import "fmt"
```

```
func main() {
```

```
    fmt.Printf("I like Go! \nI like Go! \nI like Go!")
```

```
}
```

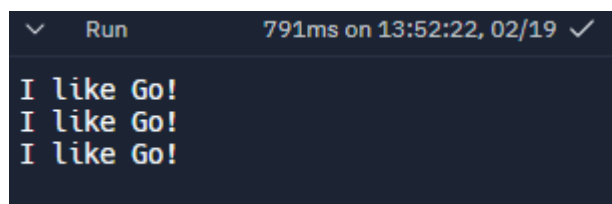


Рисунок 14 – Результат выполнения

Задача 3: Напишите программу, которая последовательно делает следующие операции с введённым числом:

1. Умножает его на 2;
2. Затем к числу прибавляется 100

Листинг:

/\* Задача: Напишите программу, которая последовательно делает следующие операции с введённым числом:

1. Умножает его на 2;
2. Затем к числу прибавляется 100\*/

```
package main
```

```
import "fmt"
```

```
func main() {
```

```
    var a float32
```

```
    fmt.Print("Введите число: ")
```



```

    fmt.Scan(&a)

    a = a * 2 + 100

    fmt.Println("a * 2 + 100 =", a)

}

```

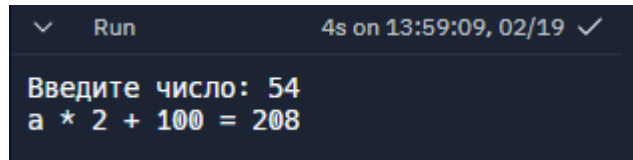


Рисунок 15 – Результат выполнения

Задача 4: Петя торопился в школу и неправильно написал программу, которая сначала находит квадраты двух чисел, а затем их суммирует. Исправьте его программу.

Листинг:

/\* Задача: Петя торопился в школу и неправильно написал программу, которая сначала находит квадраты двух чисел, а затем их суммирует. Исправьте его программу.

```

package main

import "fmt"

func main(){

    var a int

    fmt.Scan(&a) // считаем переменную 'a' с консоли
    fmt.Scan(&b) // считаем переменную 'b' с консоли

    a = a * a
    b = b * 2
    c = a + b

    fmt.Println(c)

}*/

```

```

package main

import "fmt"

func main(){

    var a, b, c int

    fmt.Scan(&a) // считаем переменную 'a' с консоли
    fmt.Scan(&b) // считаем переменную 'b' с консоли

    a = a * a
    b = b * b
    c = a + b

    fmt.Println(c)

}

```

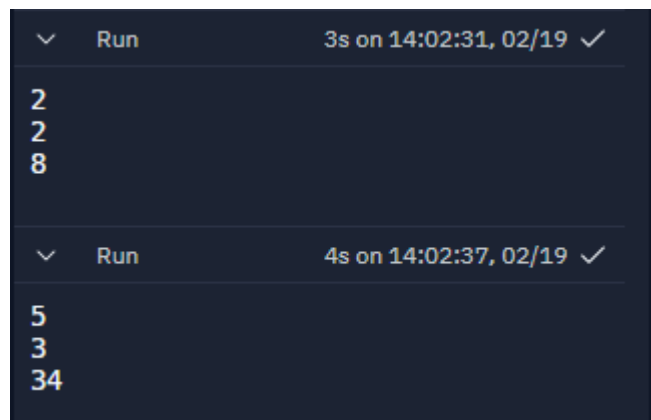


Рисунок 16 – Результат выполнения

Задача 5. По данному целому числу, найдите его квадрат

Листинг:

/\* Задача: По данному целому числу, найдите его квадрат\*/

```

package main

```

```

import (
    "fmt"
)

```

```
func main(){

    var a, result int

    fmt.Print("Введите число: ")

    fmt.Scan(&a) // считаем переменную 'a' с консоли

    result = a * a

    fmt.Println("Возведение в квадрат", result)

}
```

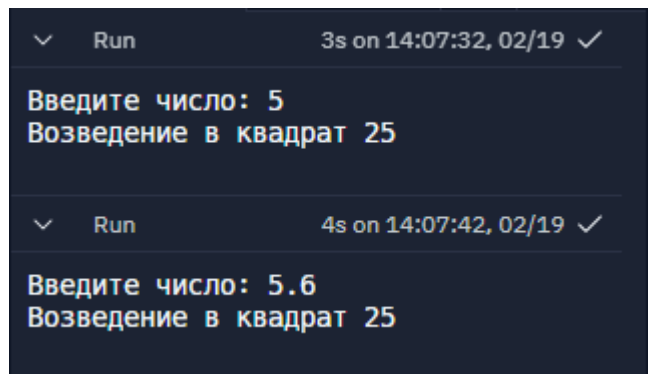


Рисунок 17 – Результат выполнения

Задача 6: Дано натуральное число, выведите его последнюю цифру. На вход дается натуральное число  $N$ , не превосходящее 10000. Выведите одно целое число – ответ на задачу.

Листинг:

/\* Задача: Дано натуральное число, выведите его последнюю цифру. На вход дается натуральное число  $N$ , не превосходящее 10000. Выведите одно целое число - ответ на задачу.\*/

```
package main
```

```
import "fmt"
```

```
func main(){
```

```
    var a, result int
```

```
    fmt.Print("Введите число: ")
```

```

fmt.Scan(&a) // считаем переменную 'a' с консоли
result = a % 10
fmt.Println("Последняя цифра числа: ", result)
}

```

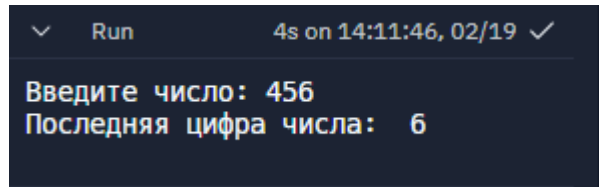


Рисунок 18 – Результат выполнения

Задача 7: Дано неотрицательное целое число. Найдите число десятков (то есть вторую цифру справа). На вход дается натуральное число  $N$ , не превосходящее 10000. Выведите одно целое число - число десятков.\*/

Листинг:

/\* Задача: Дано неотрицательное целое число. Найдите число десятков (то есть вторую цифру справа). На вход дается натуральное число  $N$ , не превосходящее 10000. Выведите одно целое число - число десятков.\*/

```

package main
import "fmt"
func main(){

    var a, result int
    fmt.Print("Введите число: ")
    fmt.Scan(&a) // считаем переменную 'a' с консоли
    result = (a % 100 - a % 10) / 10
    fmt.Println("Количество десятков: ", result)
}

```

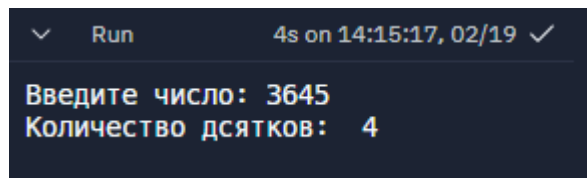


Рисунок 19 – Результат выполнения

Задача 8. Задача: Часовая стрелка повернулась с начала суток на  $d$  градусов. Определите, сколько сейчас целых часов  $h$  и целых минут  $m$ . На вход программе подается целое число  $d$  ( $0 < d < 360$ ). Выведите на экран фразу: It is ... hours ... minutes. Вместо многоточий программа должна выводить значения  $h$  и  $m$ , отделяя их от слов ровно одним пробелом.

Листинг:

/\* Задача: Часовая стрелка повернулась с начала суток на  $d$  градусов. Определите, сколько сейчас целых часов  $h$  и целых минут  $m$ . На вход программе подается целое число  $d$  ( $0 < d < 360$ ). Выведите на экран фразу: It is ... hours ... minutes.

Вместо многоточий программа должна выводить значения  $h$  и  $m$ , отделяя их от слов ровно одним пробелом.\*/

```
package main

import "fmt"

func main(){

    var d int

    fmt.Print("Enter d: ")

    fmt.Scan(&d) // считаем переменную 'a' с консоли

    fmt.Println("It is", d / 60, "hours", d % 60, "minutes.")

}
```

```
Run 5s on 14:22:10, 02/19 ✓
Введите число: 54
It is 0 hours 54 minutes.

Run 3s on 14:22:18, 02/19 ✓
Введите число: 124
It is 2 hours 4 minutes.
```

Рисунок 20 – Результат выполнения

Задача 9: Уберите лишние комментарии так, чтобы программа вывела число 100.

Листинг: /\* Задача: Уберите лишние комментарии так, чтобы программа вывела число 100.

```
package main
import "fmt"
func main(){
    // a:=44
    /*
    var a2 int = 10
    *
    a2 = a2 * 10
    fmt.Println(a2)
    */
package main
import "fmt"
func main(){
    // a:=44
    var a2 int = 10
    a2 = a2 * 10
    fmt.Println(a2)
```

```
}
```

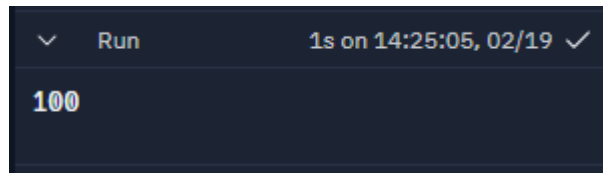


Рисунок 21 – Результат выполнения

Задача 10. Исправьте ошибку в программе ниже

Листинг:

/\* Задача: Исправьте ошибку в программе ниже:

```
package main
```

```
import "fmt"
```

```
func main(){
```

```
    var a int = 8
```

```
    const b int = 10
```

```
    a = a + b
```

```
    b = b + a
```

```
    fmt.Println(a)
```

```
}*/
```

```
package main
```

```
import "fmt"
```

```
func main(){
```

```
    var a int = 8
```

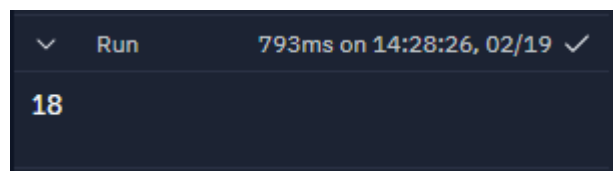
```
    var b int = 10
```

```
    a = a + b
```

```
    b = b + a
```

```
    fmt.Println(a)
```

```
}
```



## Рисунок 22 – Результат выполнения

Задача 11. : Напишите программу, которая для заданных значений  $a$  и  $b$  вычисляет площадь поверхности и объем тела, образованного вращением эллипса, заданного уравнением:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

вокруг оси  $Ox$ .

Листинг:

/\*

Задача: Напишите программу, которая для заданных значений  $a$  и  $b$  вычисляет площадь поверхности и объем тела, образованного вращением эллипса, заданного уравнением:

$$x^2/a^2 + y^2/b^2 = 1$$

вокруг оси  $Ox$ .

\*/

package main

import (

"fmt"

"math"

"os"

)

func main() {

var a float64

var b float64

fmt.Print("Enter a numb: ")

fmt.Fscan(os.Stdin, &a)



```

fmt.Print("Enter b numb: ")
fmt.Fscan(os.Stdin, &b)
// Шаг для интегрирования
step := 0.0001

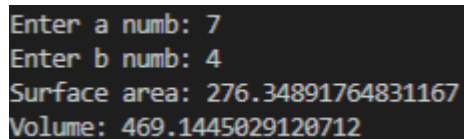
// Инициализация переменных для хранения площади поверхности и
объема
surfaceArea := 0.0
volume := 0.0

// Интегрирование для вычисления площади поверхности и объема
for x := -a; x <= a; x += step {
    y := b * math.Sqrt(1-math.Pow(x/a, 2)) // Функция описывающая
ЭЛЛИПС

    surfaceArea += 2 * math.Pi * y * step
    volume += math.Pi * math.Pow(y, 2) * step
}

fmt.Println("Surface area:", surfaceArea)
fmt.Println("Volume:", volume)
}

```



```

Enter a numb: 7
Enter b numb: 4
Surface area: 276.34891764831167
Volume: 469.1445029120712

```

Рисунок 23 – Результат выполнения

## Индивидуальное задание

Задача 1. Вариант 28 (8). Объем и площадь цилиндрической банки:  
Задайте переменные для радиуса и высоты цилиндрической банки.  
Рассчитайте и выведите объем и полную поверхность цилиндра.

Листинг:

```
// Вариант 28 (8). Объем и площадь цилиндрической банки:
// Задайте переменные для радиуса и высоты цилиндрической банки.
// Рассчитайте и выведите объем и полную поверхность цилиндра

package main

import (
    "fmt"
    "math"
    "os"
)

func main() {
    var radius float64
    var height float64
    fmt.Print("Enter first numb:")
    fmt.Fscan(os.Stdin, &radius)
    fmt.Print("Enter second numb:")
    fmt.Fscan(os.Stdin, &height)

    fmt.Println("Volume: ", math.Pi*math.Pow(radius, 2)*height)
    fmt.Println("Surface area: ", 2*math.Pi*math.Pow(radius,
2)+2*math.Pi*radius*height)
}
```

```
Enter first numb:5
Enter second numb:4
Volume: 314.1592653589793
Surface area: 282.743388230814
PS C:\Users\ynakh\OneDrive\Рабочий стол\Git\PI 1> █
```

Рисунок 24 – Результат выполнения

Задача 2. Вариант 28 (4). Даны два числа. Найти их сумму, разность, произведение, а также частное от деления первого числа на второе.

Листинг:

// Вариант 28 (4). Даны два числа. Найти их сумму, разность, произведение, а также частное от деления первого числа на второе

```
package main
```

```
import (
```

```
    "fmt"
```

```
    "os"
```

```
)
```

```
func main() {
```

```
    var first float32
```

```
    var second float32
```

```
    fmt.Print("Enter first numb:")
```

```
    fmt.Fscan(os.Stdin, &first)
```

```
    fmt.Print("Enter second numb:")
```

```
    fmt.Fscan(os.Stdin, &second)
```

```
    fmt.Println("Sum:", first+second)
```

```
    fmt.Println("Subt:", first-second)
```

```
    fmt.Println("Mult:", first*second)
```

```
    fmt.Println("Div:", first/second)
```

```
}
```

```
Enter first numb:5.3
Enter second numb:2.7
Sum: 8
Subt: 2.6000001
Mult: 14.31
Div: 1.962963
```

Рисунок 25 – Результат выполнения

Вывод: В ходе выполнения лабораторной работы было проведено исследование назначения и способов установки Go, исследование типов данных, констант и арифметических операции языка программирования Go.

Вопросы для защиты работы

1. Как объявить переменную типа `int` в Go?

Для определения переменной применяется ключевое слово `var` , после которого идет имя переменной, а затем указывается ее тип:

```
var имя_переменной int
```

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

Когда объявляется переменная, она автоматически содержит значение по умолчанию для своего типа, 0 для `int`.

3. Как изменить значение существующей переменной в Go?

Необходимо использовать оператор присвоения.

4. Что такое множественное объявление переменных в Go?

Множественное объявление позволяет объявить сразу несколько переменных в одном блоке `var`.

```
var (
    name string = "Dima"
    age int = 23
)
```

5. Как объявить константу в Go?

Для определения констант применяется ключевое слово `const` : `const pi float64 = 3.1415`

6. Можно ли изменить значение константы после ее объявления в Go?

Константы, как и переменные, хранят некоторые данные, но, в отличие от переменных, значения констант нельзя изменить, они устанавливаются один раз.

7. Какие арифметические операторы поддерживаются в Go?

У переменных есть разные операции, как в алгебре: сложение, вычитание, умножение, деление, остаток от деления, постфиксный инкремент и декремент.

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

`%` – возвращает остаток от деления (в этой операции могут принимать участие только целые числа).

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

2

10. Как считать строку с консоли в Go?

Для того чтобы это сделать, нам нужно воспользоваться методом `fmt.Scan(&a)`, где `&a` - ссылка (более точно - адрес) на переменную `a`.

11. Как считать целое число с консоли в Go?

Можно использовать `fmt.Scanf` со спецификатором формата. Спецификатором формата для целого числа является `%d`.

```
var someVar int
fmt.Scanf("%d", &someVar)
```

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

Значения, возвращаемые функцией `Scan()`, мы присваиваем переменным. Чтобы обработать ошибку достаточно получить только второе значение, поэтому первое значение сохраняется в специальной переменной, название которой содержит один символ подчеркивания. Тем самым мы игнорируем первое возвращаемое значение.

## 13. Как вывести строку в консоль в Go?

Для вывода данных на консоль мы на данном этапе будем пользоваться двумя методами, которые присутствуют в пакете `fmt`. Это `Print()` и `Println()`.

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.

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

```
fmt.Println(27)
```

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

Форматирование в Go заимствовано из функции C — `printf`. Так называемые `verbs` используются для определения формируемого числа. Это может быть специальный символ `%X`, что является плейсхолдером для значения. Для всех опций форматирования используется пакет `fmt`. Пакет `strconv` также может быть полезен в случае, если вам нужно форматировать числа с другой основой.

## 16. Как объявить переменную типа `byte` и присвоить ей значение 65? Чем отличается оператор `:=` от оператора `=` в Go?

Тонкая разница между `=` и `:=` заключается в том, когда `=` используется в объявлениях переменных.

Общая форма объявления переменной в Go:

`var name type = expression`

приведенное выше объявление создает переменную определенного типа, присваивает ей имя и устанавливает ее начальное значение.

`:=` это объявление, тогда как `=` это присвоение

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

`float32` и `float64`, `complex64` и `complex128`.

18. Как объявить и использовать несколько переменных в Go?

Можно одновременно объявить сразу несколько переменных через запятую: `var a, b, c string`.

Также можно объявить сразу несколько переменных в одном блоке `var`:

```
var (  
    name string = "Dima"  
    age int = 23  
)
```