

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ ВО «СЕВЕРО-КАВКАЗСКАЯ ФЕДЕРАЛЬНЫЙ
УНИВЕРСИТЕТ» ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ
КАФЕДРА ИНФОКОММУНИКАЦИЙ

Дисциплина: Программная инженерия

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

«Основы языка программирования Go»

Выполнила:

студентка 3 курса

группы ПИЖ-б-о-21-1

Кучеренко Светлана Юрьевна

(Подпись)

Проверил:

доцент кафедры инфокоммуникаций

Воронкин Роман Александрович

(Подпись)

Работа защищена с оценкой:

Ставрополь, 2024

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

Выполнение работы:

Примеры теоретической части:

Пример 1.

```
package main

import "fmt"

func main() {
    fmt.Println("Hello, Go!")
}
```

```
● svetik@MacBook-Air-Svetik LR1 % go run primer_1.go
Hello, Go!
```

Рисунок 1 – Результат работы программы

Пример 2.

```
package main

import "fmt"

func main() {
    fmt.Println("Hello Go"[0]) // вывод: 72
}
```

```
svetik@MacBook-Air-Svetik LR1 % go run primer_2.go
72
```

Рисунок 2 – Результат работы программы

Пример 3.

```
package main

import "fmt"

func main() {
    fmt.Println(string("Hello Go"[0])) // вывод: H
}
```

```
svetik@MacBook-Air-Svetik LR1 % go run primer_3.go
H
```

Рисунок 3 – Результат работы программы

Пример 4.

```
package main

import "fmt"

func main() {
    var hello string

    hello = "Hello Go!"

    var a int = 2019

    fmt.Println(hello)
    fmt.Println(a)
}
```

```
● svetik@MacBook-Air-Svetik LR1 % go run primer_4.go
Hello Go!
2019
```

Рисунок 4 – Результат работы программы

Пример 5.

```
package main

import "fmt"
```

```
func main() {
    var (
        name string = "Sveta"
        age  int    = 20
    )

    fmt.Println(name)
    fmt.Println(age)
}
```

```
svetik@MacBook-Air-Svetik LR1 % go run primer_5.go
Sveta
20
```

Рисунок 5 – Результат работы программы

Пример 6.

```
package main

import "fmt"

func main() {
    var a int = 10 / 6
    var m float32 = 10.0 / 6
    var c int = 10 % 3

    var x int = 1
    x++

    var y int = 10
    y--

    fmt.Println(a)
    fmt.Println(m)
    fmt.Println(c)
    fmt.Println(x)
    fmt.Println(y)
}
```

```
● svetik@MacBook-Air-Svetik LR1 % go run primer_6.go
1
1.6666666
1
2
9
```

Рисунок 6 – Результат работы программы

Пример 7.

```
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)
}
```

```
● svetik@MacBook-Air-Svetik LR1 % go run primer_7.go
Введите имя: Sveta
Введите возраст: 20
Sveta 20
```

Рисунок 7 – Результат работы программы

Пример 8.

```
package main

import "fmt"
```

```
func main() {
    name := "Sveta"
    age := 20
    fmt.Println("Me name is", name, "and I'm", age, "years old.")
}
```

```
● svetik@MacBook-Air-Svetik LR1 % go run primer_8.go
Me name is Sveta and I'm 20 years old.
```

Рисунок 8 – Результат работы программы

Пример 9.

```
package main

import (
    "fmt"
)

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

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

```
● svetik@MacBook-Air-Svetik LR1 % go run primer_9.go
45 45 3.3 3.3
```

Рисунок 9 – Результат работы программы

Пример 10.

```
package main
```

```

import (
    "fmt"
)

const (
    Sunday   = 0
    Monday   = 1
    Tuesday  = 2
    Wednesday = 3
    Thursday = 4
    Friday   = 5
    Saturday = 6
)

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

```

```

● svetik@MacBook-Air-Svetik LR1 % go run primer_10.go
0
6

```

Рисунок 10 – Результат работы программы

Пример 11.

```

package main

import (
    "fmt"
)

const (
    Sunday = iota
    Monday
    Tuesday
    Wednesday
    Thursday
    Friday
    Saturday

```

```
)

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

```
● svetik@MacBook-Air-Svetik LR1 % go run primer_11.go
0
6
```

Рисунок 11 – Результат работы программы

Пример 12.

```
package main

import "fmt"

const (
    c0 = iota // c0 == 0
    c1 = iota // c1 == 1
    c2 = iota // c2 == 2
)

const (
    Sunday = iota
    Monday
    Tuesday
    Wednesday
    Thursday
    Friday
    Saturday
    _ // пропускаем 7
    Add
)

const (
    u = iota * 42 // u == 0 (индекс - 0, поэтому 0 * 42 = 0)
    v float64 = iota * 42 // v == 42 (индекс - 1, поэтому 1 * 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
    y = iota // y == 1
)

func main() {
    fmt.Println(c0, c1, c2) // вывод: 0 1 2
    fmt.Println(Sunday)    // вывод: 0
    fmt.Println(Saturday)  // вывод: 6
    fmt.Println(Add)       // вывод: 8
    fmt.Println(u, v, w)   // вывод: 0 42 84
    fmt.Println(x, y)      // вывод: 0 1
}

```

```

● svetik@MacBook-Air-Svetik LR1 % go run primer_12.go
0 1 2
0
6
8
0 42 84
0 1

```

Рисунок 12 – Результат работы программы

Пример 13.

```

package main

import (
    "fmt"
    "math"
)

func main() {
    a := math.Abs(-5.67) // Возвращает абсолютное значение числа.
    b := math.Ceil(5.67) // Округляет число вверх до ближайшего целого.
    c := math.Floor(5.67) // Округляет число вниз до ближайшего целого.
    d := math.Sqrt(16)    // Возвращает квадратный корень числа.
    f := math.Pow(2, 3)   // Возводит число x в степень y .
}

```

```
sinValue := math.Sin(math.Pi / 2) // Возвращают синус, косинус и тангенс угла в радианах соответственно.
g := math.Log(10)                // Возвращает натуральный логарифм числа.
maxVal := math.Max(3, 7)         // Макс значение
minVal := math.Min(3, 7)        // Мин значение
h := math.Mod(10, 3)             // Возвращает остаток от деления x на y.
i := math.Round(5.67)           // Округляет число к ближайшему целому.
j := math.Trunc(5.67)            // Отбрасывает дробную часть числа.
posInf := math.Inf(1)           // Возвращает положительную бесконечность
negInf := math.Inf(-1)          // Возвращает отрицательную бесконечность
nan := math.NaN()               // Возвращает "Not a Number" (NaN).
k := math.Exp(2)                 // Возвращает экспоненту (e) в степени x.
l := math.Exp2(3)                // Возвращает 2 в степени x.
m := math.Expm1(1)               // Возвращает e в степени x минус 1
n := math.Log10(100)             // Возвращает десятичный логарифм числа.
o := math.Log2(8)                // Возвращает двоичный логарифм числа.
p := math.Log1p(1)               // Возвращает двоичный логарифм числа x плюс 1
isNegative := math.Signbit(-5)   // Возвращает true, если x отрицательное или отрицательный ноль.
```

```
fmt.Println(a)
fmt.Println(b)
fmt.Println(c)
fmt.Println(d)
fmt.Println(f)
fmt.Println(sinValue)
fmt.Println(g)
fmt.Println(maxVal)
fmt.Println(minVal)
fmt.Println(h)
fmt.Println(i)
fmt.Println(j)
fmt.Println(posInf)
fmt.Println(negInf)
fmt.Println(nan)
fmt.Println(k)
fmt.Println(l)
fmt.Println(m)
fmt.Println(n)
fmt.Println(o)
fmt.Println(p)
fmt.Println(isNegative)
```

```
}
```

```
● svetik@MacBook-Air-Svetik LR1 % go run primer_13.go
5.67
6
5
4
8
1
2.302585092994046
7
3
1
6
5
+Inf
-Inf
NaN
7.38905609893065
8
1.718281828459045
2
3
0.6931471805599453
true
```

Рисунок 13 – Результат работы программы

Практическая часть:

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

```
package main

import "fmt"

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

```
● svetik@MacBook-Air-Svetik Task1 % go run task1.go
I like Go!
```

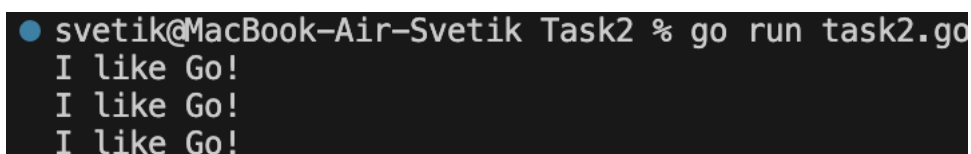
Рисунок 14 – Результат работы программы

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

```
package main

import "fmt"

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



```
svetik@MacBook-Air-Svetik Task2 % go run task2.go
I like Go!
I like Go!
I like Go!
```

Рисунок 15 – Результат работы программы

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

- 1) Число умножается на 2;
- 2) Затем к числу прибавляется 100

После этого должен быть вывод получившегося числа на экран.

```
package main

import "fmt"

func main() {
    var number int
    var res int

    fmt.Print("Введите число: ")
    fmt.Scan(&number)

    res = number * 2
    res = res + 100
```

```
fmt.Println(res)
}
```

```
● svetik@MacBook-Air-Svetik Task3 % go run task3.go
Введите число: 1
102
```

Рисунок 16 – Результат работы программы

Задача 4: 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, b, c 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)
}
```

```
● svetik@MacBook-Air-Svetik Task4 % go run task4.go
2
2
8
```

Рисунок 17 – Результат работы программы

Задача 5: По данному целому числу, найдите его квадрат.

```

package main

import "fmt"

func main() {
    var a int

    fmt.Print("Введите целое число:")
    fmt.Scan(&a)

    result := a * a
    fmt.Println(result)
}

```

```

● svetik@MacBook-Air-Svetik Task5 % go run task5.go
Введите целое число:3
9

```

Рисунок 18 – Результат работы программы

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.

```

package main

import (
    "fmt"
    "math"
)

func main() {
    var a float64

    fmt.Print("Введите целое число:")
    fmt.Scan(&a)

    result := math.Mod(a, 10)
}

```

```
fmt.Println(result)
}
```

```
● svetik@MacBook-Air-Svetik Task6 % go run task6.go
Введите целое число:123
3
```

Рисунок 19 – Результат работы программы

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.

```
package main

import "fmt"

func main() {
    var a int

    fmt.Print("Введите целое положительное число:")
    fmt.Scan(&a)

    if a < 0 || a > 10000 {
        fmt.Println("Введенное число не соответствует условиям")
        return
    }

    result := (a / 10) % 10
    fmt.Println(result)
}
```

```

• svetik@MacBook-Air-Svetik Task7 % go run task7.go
Введите целое положительное число:2010
1
• svetik@MacBook-Air-Svetik Task7 % go run task7.go
Введите целое положительное число:-12
Введенное число не соответствует условиям
• svetik@MacBook-Air-Svetik Task7 % go run task7.go
Введите целое положительное число:10001
Введенное число не соответствует условиям

```

Рисунок 20 – Результат работы программы

Задача 8: 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("Введите целое положительное число:")
    fmt.Scan(&d)

    if d < 0 || d > 360 {
        fmt.Println("Введенное число не соответствует условиям")
        return
    }

    h := d / 30
    m := (d % 30) * 2
    fmt.Println("It is ", h, "hours ", m, "minutes.")
}

```



```
● svetik@MacBook-Air-Svetik Task8 % go run task8.go
Введите целое положительное число:90
It is 3 hours 0 minutes.
```

Рисунок 21 – Результат работы программы

Задача 9: 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() {
    var a2 int = 10
    a2 = a2 * 10
    fmt.Println(a2)
}
```

```
● svetik@MacBook-Air-Svetik Task9 % go run task9.go
100
```

Рисунок 22 – Результат работы программы

Задача 10: 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
    const b int = 10
    a = a + b
    fmt.Println(a)
}
```

```
● svetik@MacBook-Air-Svetik Task10 % go run task10.go
18
```

Рисунок 23 – Результат работы программы

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 .

```
package main

import (
    "fmt"
```

```

"math"
)

func main() {
    var a float64 = 8 // полуось эллипса по оси x
    var b float64 = 10 // полуось эллипса по оси y

    // Функция для вычисления y в зависимости от x
    y := func(x float64) float64 {
        return b * math.Sqrt(1 - math.Pow(x/a, 2))
    }

    // Шаг интегрирования
    step := 0.01

    // Вычисление площади поверхности
    S := 0.0
    for x := -a; x < a; x += step {
        rectWidth := step
        rectHeight := y(x)
        S += rectWidth * rectHeight
    }

    // Вычисление объема
    V := 0.0
    for x := -a; x < a; x += step {
        rectWidth := step
        rectHeight := math.Pow(y(x), 2)
        V += rectWidth * rectHeight
    }

    fmt.Println("Площадь поверхности: ", S)
    fmt.Println("Объем тела: ", V)
}

```

```

● svetik@MacBook-Air-Svetik Task11 % go run task11.go
Площадь поверхности: 125.66162738609269
Объем тела: 1066.666250000013

```

Рисунок 24 – Результат работы программы

Индивидуальное задание 1.

12. **Объем и площадь поверхности шара:** Задайте переменную для радиуса шара. Рассчитайте и выведите объем и площадь его поверхности.

```
package main

import (
    "fmt"
    "math"
)

func main() {
    // Задаем радиус шара
    var radius float64 = 5

    // Вычисляем объем шара
    V := (4.0 / 3.0) * math.Pi * math.Pow(radius, 3)

    // Вычисляем площадь поверхности шара
    S := 4 * math.Pi * math.Pow(radius, 2)

    fmt.Println("Объем шара: ", V)
    fmt.Println("Площадь поверхности шара: ", S)
}
```

```
● svetik@MacBook-Air-Svetik Ind % go run ind.go
Объем шара: 523.598775598299
Площадь поверхности шара: 314.1592653589793
```

Рисунок 25 – Результат работы программы

Индивидуальное задание 2.

12. Известна стоимость монитора, системного блока, клавиатуры и мыши.
Сколько будут стоить 3 компьютера из этих элементов? N компьютеров?

```
package main

import "fmt"

func main() {
```

```

var monitorCost float64 = 8000
var systemBlockCost float64 = 60000
var keyboardCost float64 = 1500
var mouseCost float64 = 800
var numberOfComputers int

fmt.Print("Введите количество компьютеров: ")
fmt.Scanln(&numberOfComputers)

// Вычисление общей стоимости компьютеров
totalCost := float64(numberOfComputers) * (monitorCost + systemBlockCost + keyboardCost + mouseCost)

fmt.Println("Общая стоимость", numberOfComputers, "компьютеров: ", totalCost)
}

```

```

● svetik@MacBook-Air-Svetik Ind2 % go run ind2.go
Введите количество компьютеров: 3
Общая стоимость 3 компьютеров: 210900

```

Рисунок 26 – Результат работы программы

Ответы на контрольные вопросы:

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

```
var a int
```

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

Значение по умолчанию для переменной типа `int` в Go равно 0.

3. Как изменить значение существующей переменной в Go?

Для изменения значения существующей переменной в Go, просто присвойте ей новое значение, используя оператор присваивания `"="`.

4. Что такое множественное объявление переменных в Go?

Множественное объявление переменных в Go позволяет объявить несколько переменных одновременно.

5. Как объявить константу в Go?

Для объявления константы используется ключевое слово «const».

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

Нельзя

7. Какие арифметические операторы поддерживаются в Go?

Сложение, вычитание, умножение, деление, остаток от деления

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

Оператор «%»

9. Какой результат выражения 5 / 2 в Go?

Значение «2», так как дробная часть отбрасывается.

10. Как считать строку с консоли в Go?

Нужно использовать функцию «fmt.Scan()».

11. Как считать целое число с консоли в Go?

Нужно использовать функцию «fmt.Scan()» в сочетании с оператором взятия адреса «&».

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

```
var num int
_, err := fmt.Scanln(&num)
if err != nil {
    fmt.Println("Ошибка:", err)
}
```

13. Как вывести строку в консоль в Go?

Нужно использовать функцию «fmt.Println(“Hello Go!”)».

14. Как вывести значение переменной типа int в консоль?

Нужно использовать функцию «fmt.Println(x)».

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

Для форматирования вывода числа с плавающей точкой в Go используйте функцию Printf() с форматом «%.nf», где n - количество знаков после запятой.

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

```
var b byte = 65
```

17. Чем отличается оператор := от оператора = в Go?

Оператор := используется для объявления и инициализации новых переменных, в то время как оператор = используется только для присваивания значения существующей переменной.

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

float32, float64.

19. Как объявить и использовать несколько переменных в Go?

```
var (  
x int = 10  
y int = 20  
)
```

