

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

Дисциплина: Программная инженерия

ОТЧЕТ ПО ПРАКТИЧЕСКОМУ ЗАНЯТИЮ №1

Основы языка программирования Go

Выполнил:

студент 3 курса направления
подготовки «Программная
инженерия»
группы ПИЖ–б–о–21–1

Образцова Мария Дмитриевна

(Подпись)

Проверил:

канд. тех. наук., доцент кафедры
инфокоммуникаций института
цифрового развития СКФУ

Воронкин Роман Александрович

(Подпись)

Работа защищена с оценкой:

Ставрополь, 2024

Тема: основы языка программирования Go

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

Примеры:

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\primers\pr_1> go run pr_1.go  
Hello, Go!
```

Рисунок 1 – Пример 1

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\primers\pr_2> go run pr_2.go  
72
```

Рисунок 2 – Пример 2

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\primers\pr_3> go run pr_3.go  
H
```

Рисунок 3 – Пример 3

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\primers\pr_4> go run pr_4.go  
Hello Go!  
2019
```

Рисунок 4 – Пример 4

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\primers\pr_5> go run pr_5.go  
Maria  
20
```

Рисунок 5 – Пример 5

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\primers\pr_6> go run pr_6.go  
1  
1.6666666  
1  
2  
9
```

Рисунок 6 – Пример 6

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\primers\pr_7> go run pr_7.go  
Введите имя: Maria  
Введите возраст: 20  
Maria 20
```

Рисунок 7 – Пример 7

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\primers\pr_8> go run pr_8.go  
Me name is Maria and I'm 20 years old.
```

Рисунок 8 – Пример 8

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\primers\pr_9> go run pr_9.go  
45 45 3.3 3.3
```

Рисунок 9 – Пример 9

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\primers\pr_10> go run pr_10.go  
0  
6
```

Рисунок 10 – Пример 10

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\primers\pr_11> go run pr_11.go  
0  
6
```

Рисунок 11 – Пример 11

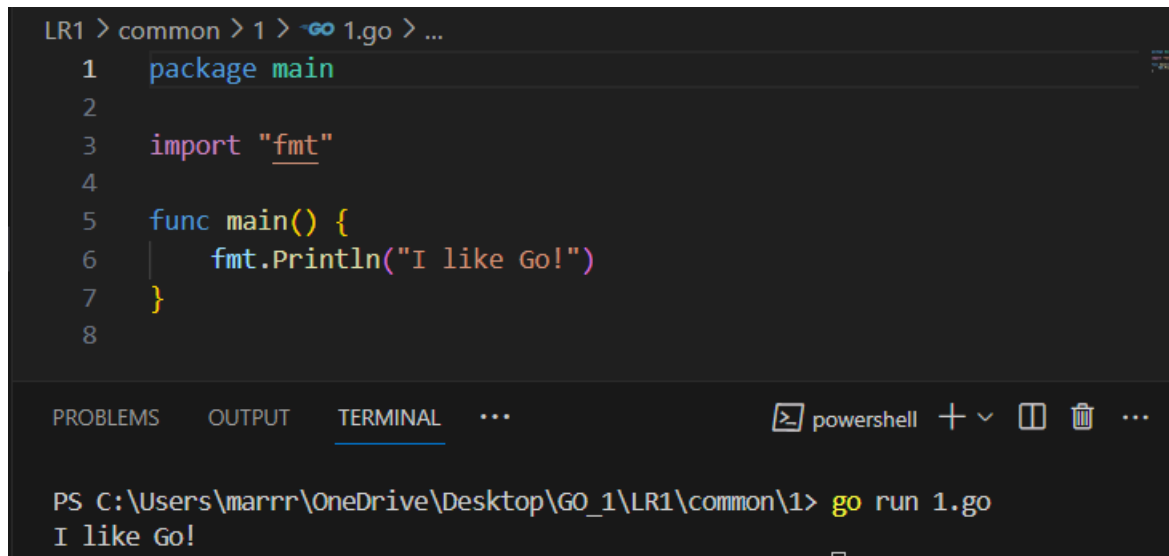
```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\primers\pr_12> go run pr_12.go  
0 1 2  
0  
6  
8  
0 42 84  
0 1
```

Рисунок 12 – Пример 12

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\primers\pr_13> go run pr_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 – Пример 13

Общие



```
LR1 > common > 1 > -go 1.go > ...  
1 package main  
2  
3 import "fmt"  
4  
5 func main() {  
6     fmt.Println("I like Go!")  
7 }  
8
```

PROBLEMS OUTPUT TERMINAL ... powershell + ▾ □ ☒ ...

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\1> go run 1.go  
I like Go!
```

Рисунок 14 – Общее задание 1



```
LR1 > common > 2 > -go 2.go > ...  
1 package main  
2  
3 import "fmt"  
4  
5 func main() {  
6     fmt.Println("I like Go!")  
7     fmt.Println("I like Go!")  
8     fmt.Println("I like Go!")  
9 }  
10
```

PROBLEMS OUTPUT TERMINAL ... powershell + ▾ □ ☒ ...

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\2> go run 2.go  
I like Go!  
I like Go!  
I like Go!
```

Рисунок 15 – Общее задание 2

```
LR1 / common / 3 / 3.go / main
1  package main
2
3  import "fmt"
4
5  func main() {
6      var number int
7      var res int
8
9      fmt.Print("Enter num: ")
10     fmt.Scan(&number)
11
12     res = number * 2
13     res = res + 100
14
15     fmt.Println(res)
16 }
17
```

PROBLEMS OUTPUT TERMINAL ... powershell + - [] [X] ...

PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\3> go run 3.go
Enter num: 5
110

Рисунок 16 – Общее задание 3

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var a, b, c int
7
8     fmt.Scan(&a)
9     fmt.Scan(&b)
10    a = a * a
11    b = b * 2
12    c = a + b
13
14    fmt.Println(c)
15 }
16
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + -

PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\4> go run 4.go

5

2

29

Рисунок 17 – Общее задание 4

```
LR1 > common > 5 > 5.go > main
1  package main
2
3  import "fmt"
4
5  func main() {
6      var a int
7
8      fmt.Print("Enter num:")
9      fmt.Scan(&a)
10
11      result := a * a
12      fmt.Println(result)
13  }
14
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + v

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\5> go run 5.go
Enter num:5
25
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\5>
```

Рисунок 18 – Общее задание 5

```
LR1 > common > 6 > 6.go > main
1  package main
2
3  import (
4      "fmt"
5      "math"
6  )
7
8  func main() {
9      var a float64
10
11      fmt.Print("Enter num:")
12      fmt.Scan(&a)
13
14      result := math.Mod(a, 10)
15      fmt.Println(result)
16  }
17
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + v

PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\6> go run 6.go
Enter num:7
7
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\6> go run 6.go
Enter num:158
8
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\6>

Рисунок 19 – Общее задание 6


```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var a int
7
8     fmt.Print("Введите неотрицательное целое число:")
9     fmt.Scan(&a)
10
11     if a < 0 || a > 10000 {
12         fmt.Println("число не соответствует условиям")
13         return
14     }
15
16     result := (a / 10) % 10
17     fmt.Println(result)
18 }
19
```

PROBLEMS OUTPUT TERMINAL ... powershell + - [] [X]

PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\7> go run 7.go
Введите целое положительное число:157
5
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\7> go run 7.go
Введите целое положительное число:1497
9
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\7> |

Рисунок 20 – Общее задание 7

```
LR1 > common > 8 > go 8.go > main
1  package main
2
3  import "fmt"
4
5  func main() {
6      var d int
7
8      fmt.Print("Введите неотрицательное целое число:")
9      fmt.Scan(&d)
10
11     if d < 0 || d > 360 {
12         fmt.Println("число не соответствует условиям")
13         return
14     }
15
16     h := d / 30
17     m := (d % 30) * 2
18     fmt.Println("It is ", h, "hours ", m, "minutes.")
19 }
20
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + -

PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\8> go run 8.go
Введите неотрицательное целое число:180
It is 6 hours 0 minutes.
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\8> |

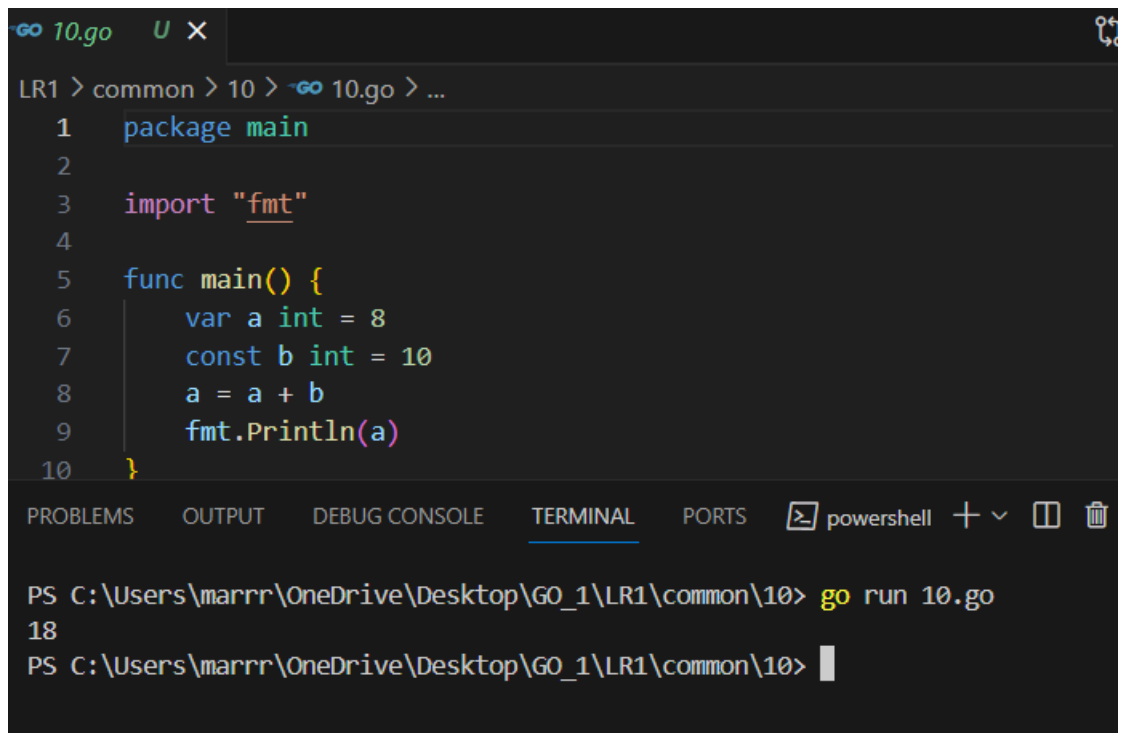
Рисунок 21 – Общее задание 8

```
LR1 > common > 9 > go 9.go > ...
1  package main
2
3  import "fmt"
4
5  func main() {
6      var a2 int = 10
7      a2 = a2 * 10
8      fmt.Println(a2)
9  }
10
```

PROBLEMS OUTPUT TERMINAL ... powershell + -

PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\9> go run 9.go
100
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\9> |

Рисунок 22 – Общее задание 9



The image shows a screenshot of a Go IDE. The top part displays a Go program in a file named `10.go`. The program defines a `main` package and a `main` function. Inside the function, it declares a variable `a` of type `int` with the value 8, a constant `b` of type `int` with the value 10, adds `a` and `b`, and prints the result using `fmt.Println(a)`.

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var a int = 8
7     const b int = 10
8     a = a + b
9     fmt.Println(a)
10 }
```

The bottom part of the IDE shows the `TERMINAL` tab. It displays the command `go run 10.go` being executed in a PowerShell window. The output of the program is `18`.

```
PS C:\Users\marrr\OneDrive\Desktop\GO_1\LR1\common\10> go run 10.go
18
PS C:\Users\marrr\OneDrive\Desktop\GO_1\LR1\common\10>
```

Рисунок 23 – Общее задание 9

```
LR1 > common > 11 > -go 11.go > main
1  package main
2
3  import (
4      "fmt"
5      "math"
6  )
7
8  func main() {
9      // Заданные значения a и b
10     var a, b float64 = 2.0, 3.0 // Замените на ваши значения
11
12     // Вычисление площади поверхности
13     surfaceArea := math.Pi*math.Pow((a+b)/2, 2) + math.Pi*(a*b)
14     fmt.Printf("Площадь поверхности: %.2f\n", surfaceArea)
15
16     // Вычисление объема
17     volume := (4.0 / 3.0) * math.Pi * a * b * b
18     fmt.Printf("Объем тела: %.2f\n", volume)
19 }
20
```

PROBLEMS OUTPUT TERMINAL ... powershell + v [] [X] ...

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\11> go run 11.go
Площадь поверхности: 96.59
Объем тела: 75.40
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\common\11>
```

Рисунок 24 – Общее задание 8

ИНДИВИДУАЛЬНОЕ 1

17. Сферический треугольник: Задайте переменные для длин трех сторон сферического треугольника. Рассчитайте его углы и выведите результат.

```
LR1 > ind > 1 > 1.go > main
2
3 import (
4     "fmt"
5     "math"
6 )
7
8 func main() {
9     var sideA, sideB, sideC float64
10
11     // Ввод длин сторон с клавиатуры
12     fmt.Println("Введите длины трех сторон сферического треугольника:")
13     fmt.Print("Длина стороны A: ")
14     fmt.Scan(&sideA)
15     fmt.Print("Длина стороны B: ")
16     fmt.Scan(&sideB)
17     fmt.Print("Длина стороны C: ")
18     fmt.Scan(&sideC)
19
20     // Вычисление углов сферического треугольника
21     angleA := math.Acos((math.Cos(sideA) - math.Cos(sideB) * math.Cos(sideC)) /
22     angleB := math.Acos((math.Cos(sideB) - math.Cos(sideA) * math.Cos(sideC)) /
23     angleC := math.Acos((math.Cos(sideC) - math.Cos(sideA) * math.Cos(sideB)) /
```

PROBLEMS OUTPUT TERMINAL ... powershell + -

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\ind\1> go run 1.go
Введите длины трех сторон сферического треугольника:
Длина стороны A: 1.46
Длина стороны B: 0.785
Длина стороны C: 1.45
Угол A: 87.93 градусов
Угол B: 45.29 градусов
Угол C: 86.55 градусов
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\ind\1>
```

Рисунок 25 – Индивидуальное задание 1

ИНДИВИДУАЛЬНОЕ 2

17. Напишите программу, в которой вычисляется сумма, разность, произведение, частное и среднее арифметическое двух целых чисел, введенных с клавиатуры. Например, при вводе чисел 2 и 7 должен быть получен ответ вида:

$2+7=9$ $2-7=-5$ $2*7=14$ $2/7=0.2857142857142857$ $(2+7)/2=4.5$

```
LR1 > ind > 2 > go 2.go > main
1  package main
2
3  import (
4      "fmt"
5  )
6
7  func main() {
8      var num1, num2 int
9
10     fmt.Print("Введите первое число: ")
11     fmt.Scan(&num1)
12     fmt.Print("Введите второе число: ")
13     fmt.Scan(&num2)
14
15     // Вычисление суммы, разности, произведения и част
16     sum := num1 + num2
17     diff := num1 - num2
18     prod := num1 * num2
19     quotient := float64(num1) / float64(num2)
20
21     // Вычисление среднего арифметического
22     average := float64(sum) / 2.0
23
24     // Вывод результатов
25     fmt.Printf("%d+%d=%d\n", num1, num2, sum)
26     fmt.Printf("%d-%d=%d\n", num1, num2, diff)
27     fmt.Printf("%d*%d=%d\n", num1, num2, prod)
28     fmt.Printf("%d/%d=%.16f\n", num1, num2, quotient)
29     fmt.Printf("(%d+%d)/2=%.1f\n", num1, num2, average)
30 }
31
```

PROBLEMS OUTPUT TERMINAL ... powershell + - [] [X] ...

```
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\ind\2> go run 2.go
Введите первое число: 5
Введите второе число: 9
5+9=14
5-9=-4
5*9=45
5/9=0.5555555555555556
(5+9)/2=7.0
PS C:\Users\marr\OneDrive\Desktop\GO_1\LR1\ind\2>
```

Рисунок 26 – Индивидуальное задание 2

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

Контрольные вопросы:

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

Для объявления переменной типа `int` в Go используется следующий синтаксис: `var x int`

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

Переменной типа `int` в Go по умолчанию присваивается нулевое значение, которое для типа `int` равно 0.

3. Как изменить значение существующей переменной в Go?

Чтобы изменить значение существующей переменной в Go, можно просто присвоить ей новое значение с помощью оператора присваивания `=`. Например:

`x := 5` // объявление переменной `x` и присвоение ей значения 5
`x = 10` // изменение значения переменной `x` на 10

4. Что такое множественное объявление переменных в Go?

Множественное объявление переменных в Go позволяет объявить несколько переменных одновременно в одном операторе. Например:

`var x, y, z int` // объявление трех переменных типа `int`

5. Как объявить константу в Go?

Для объявления константы в Go используется ключевое слово `const`. Например:

`const pi = 3.14159`

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

Нет, нельзя изменить значение константы после ее объявления в Go.

7. Какие арифметические операторы поддерживаются в Go?

В Go поддерживаются следующие арифметические операторы:

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

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

Оператор % используется для выполнения операции остатка от деления в Go.

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

В Go результат выражения $5 / 2$ будет равен 2, так как оба операнда являются целыми числами, поэтому результат тоже будет целым числом.

10. Как считать строку с консоли в Go?

Для считывания строки с консоли в Go используется функция `fmt.Scanln()` или `fmt.Scan()`.

11. Как считать целое число с консоли в Go?

Для считывания целого числа с консоли в Go используется функция `fmt.Scanln()` или `fmt.Scan()` в сочетании с оператором ввода `&`.

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

Для обработки ошибок при считывании данных с консоли в Go можно использовать функцию `fmt.Scanln()` или `fmt.Scan()` в сочетании с проверкой возвращаемого значения функции на ошибку.

13. Как вывести строку в консоль в Go?

Для вывода строки в консоль в Go используется функция `fmt.Println()` или `fmt.Printf()`.

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

Для вывода значения переменной типа `int` в консоль в Go используется функция `fmt.Println()` или `fmt.Printf()` с форматированием.

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

Для форматирования вывода числа с плавающей точкой в Go используются строковые форматы, такие как `%f`, `%e`, `%g` и другие, в сочетании с функцией `fmt.Printf()`.

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

Чтобы объявить переменную типа `byte` и присвоить ей значение 65 в Go, можно использовать следующий код:

```
var b byte = 65
```

Оператор `:=` используется для объявления и инициализации переменной одновременно. Он используется только внутри функций.

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

В Go типы данных для представления чисел с плавающей точкой включают в себя `float32` и `float64`.

18. Как объявить и использовать несколько переменных в Go?

Для объявления и использования нескольких переменных в Go можно использовать множественное объявление переменных, разделяя их запятыми,

или использовать операторы присваивания в сочетании с множественным возвращением значений из функций.

19. Управление памятью Гоу (как распределяется)

20. Алгоритм управления памятью

Go использует управление памятью с помощью сборщика мусора. Вот краткое описание того, как это происходит:

Распределение памяти: когда программа создает новые переменные или структуры данных, Go автоматически выделяет память для них. Это происходит при помощи встроенных функций и операторов, таких как ``new``, ``make`` или синтаксические элементы, например, объявление переменных.

Сборка мусора: во время выполнения программы сборщик мусора в Go периодически сканирует все выделенные объекты в памяти и определяет, какие из них больше не используются. Это происходит автоматически, без явного участия программиста. Когда сборщик мусора обнаруживает объекты, на которые нет ссылок, он освобождает память, занимаемую этими объектами, чтобы она могла быть повторно использована для новых объектов.

Алгоритм управления памятью в Go основан на маркировке и освобождении (mark-and-sweep). В процессе маркировки сборщик мусора проходит по всем объектам в памяти и маркирует те, на которые есть ссылки. Затем в процессе освобождения он освобождает память, занимаемую объектами, на которые не указывает ни одна ссылка. Это позволяет Go автоматически управлять памятью и избежать утечек памяти.

Такой подход упрощает работу программиста, так как ему не нужно явно управлять памятью, но также вносит некоторые накладные расходы на выполнение программы из-за работы сборщика мусора. Однако в большинстве случаев это компенсируется удобством и надежностью работы с памятью.