

# Строки

```
package main
```

```
import "fmt"
```

```
func main() {
```

```
    fmt.Println()
```

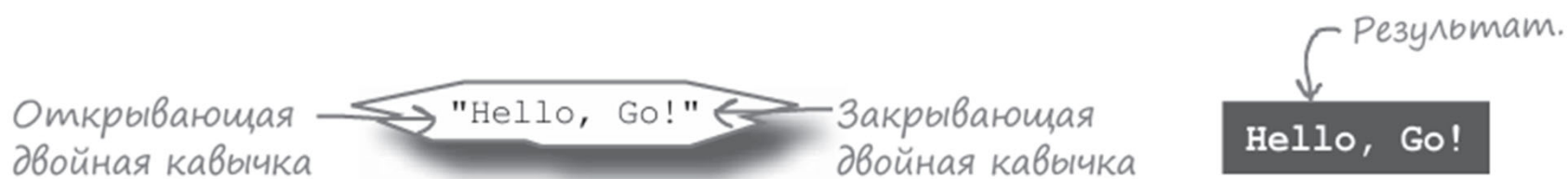
```
}
```

Вставьте сюда  
свой код!



# Записать Строки

В аргументах `Println` передавались **строки**. Строка представляет собой последовательность байтов, которые обычно представляют символы текста. Строки можно определять прямо в программе в виде **строковых литералов**: компилятор Go интерпретирует текст, заключенный в двойные кавычки, как строку.



## Записать Строки

Некоторые управляющие символы, которые неудобно вводить с клавиатуры (символы новой строки, табуляции и т. д.), внутри строк могут представляться в виде **служебных последовательностей**: символа «обратный слеш», за которым следует другой символ (или символы).

<i>Последовательность sequence</i>	<i>Значение</i>
<code>\n</code>	Символ новой строки
<code>\t</code>	Символ табуляции
<code>\"</code>	Двойная кавычка
<code>\\</code>	Обратный слеш

# Строки


<i>Новая строка внутри строки.</i>	<i>Результат.</i>
"Hello, \nGo!"	Hello, Go!
"Hello, \tGo!"	Hello,    Go!
"Quotes: \"\""	Quotes: ""
"Backslash: \\"	Backslash: \

## Записать Руны

Если строки обычно используются для представления последовательностей символов, то **руны** в языке Go представляют отдельные символы.

Строковые литералы заключаются в двойные кавычки ("), а **рунные литералы** записываются в одиночных кавычках (').

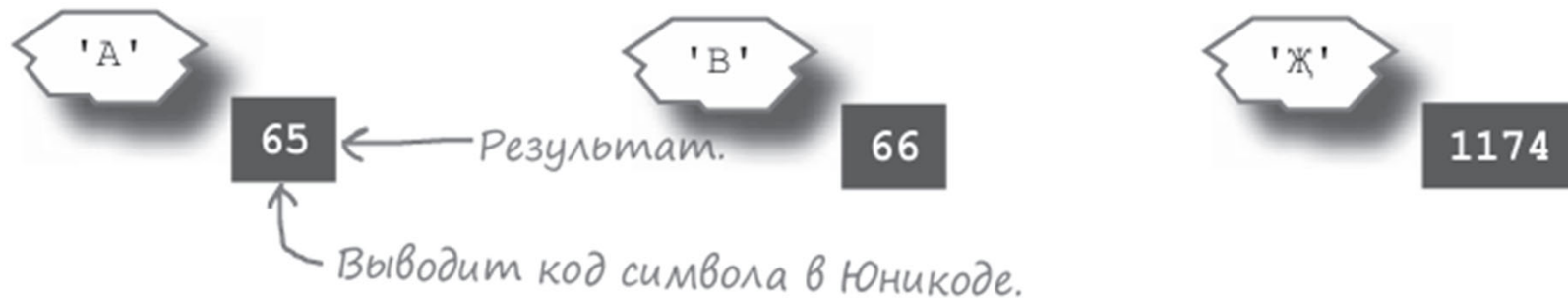
# Руны

```
package main    И снова наш шаблон...  
  
import "fmt"  
  
func main() {  
    fmt.Println()  
}
```

Вставьте  
сюда свой  
код!

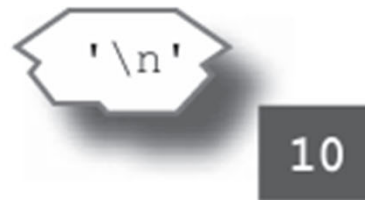
# Руны

В программах Go могут использоваться практически любые символы любых мировых языков, потому что в Go для хранения рун используется стандарт Юникод. Руны хранятся в виде числовых кодов, а не в виде символов; если передать руну функции `fmt.Println`, то выведется числовой код, а не исходный символ.



# Руны

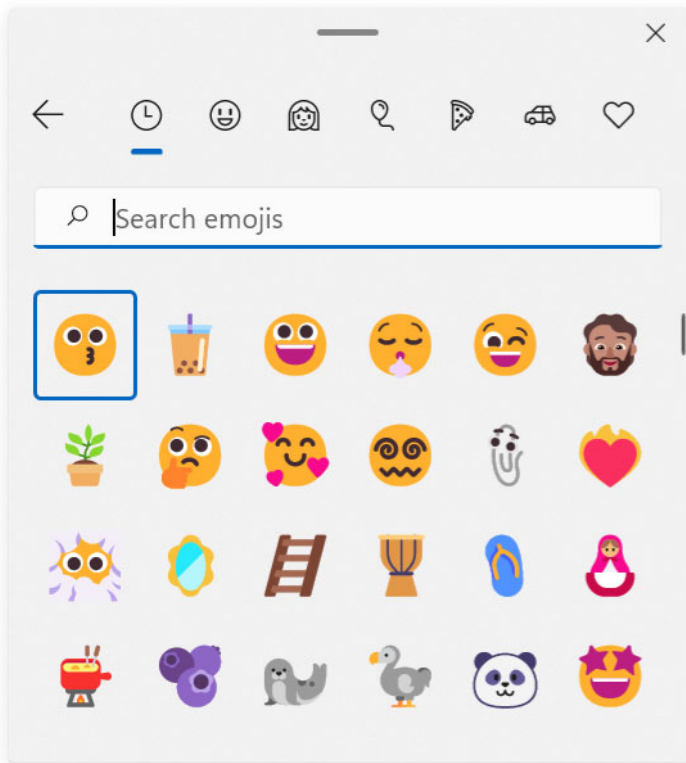
В рунных литералах (как и в строковых) можно использовать служебные последовательности для представления символов, которые неудобно вводить с клавиатуры для включения в программу:





# Руны

Чтобы открыть панель эмодзи в Windows 11, можно использовать сочетание клавиш «Win + Ю»



```
package main
```


```
import "fmt"
```

```
func main() {
```

```
    fmt.Println()
```

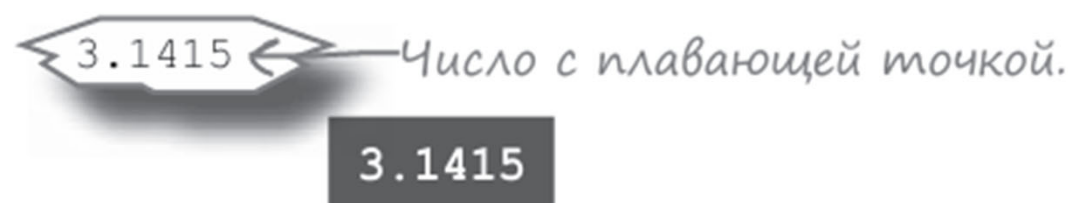
```
}
```

Вставьте сюда  
свой код!



# Числа

Числа тоже можно определять прямо в программном коде. Это еще проще, чем определять строковые литералы: просто введите нужное число.



## Записать Математические операции и сравнения

Основные математические операторы Go работают так же, как и в большинстве других языков.

Оператор `+` выполняет сложение,  
оператор `-` выполняет вычитание,  
оператор `*` — умножение,  
оператор `/` — деление.



