


Вызов функции

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

Вызов
функции
Println.



В нашем примере вызывается функция Println из пакета fmt.

Записать Вызов функции

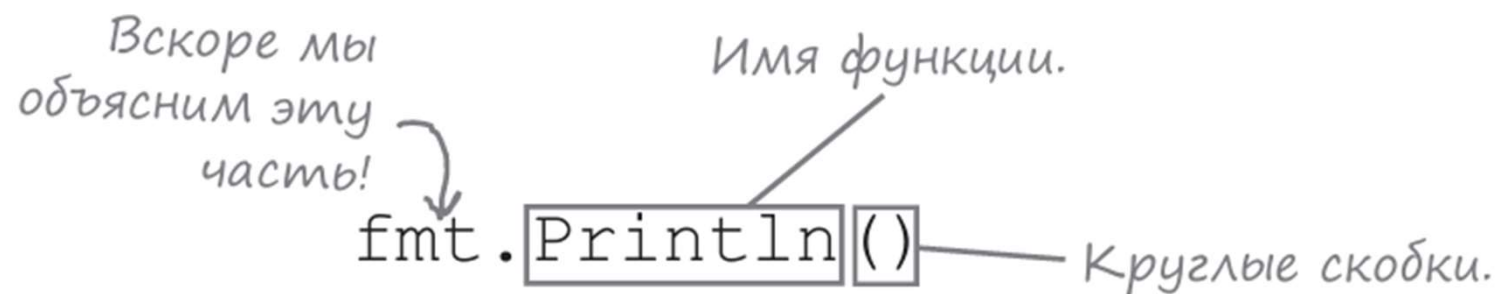
Чтобы вызвать функцию, введите имя функции (в данном случае Println) и пару круглых скобок.

Вскоре мы
объясним эту
часть!

Имя функции.

fmt.Println()

Круглые скобки.

The diagram shows the code snippet 'fmt.Println()' with several annotations. A curved arrow points from the text 'Вскоре мы объясним эту часть!' to the dot between 'fmt' and 'Println'. A straight line points from 'Имя функции.' to the 'Println' text. Another straight line points from 'Круглые скобки.' to the empty parentheses '()'.

Записать Вызов функции

Как и многие функции, `Println` может получать один или несколько аргументов — значений, с которыми должна работать функция. Аргументы перечисляются в круглых скобках после имени функции.

*В круглых скобках перечисляются
аргументы, разделенные запятыми.*

`fmt.Println("First argument", "Second argument")`

Результат. —→

First argument Second argument

Записать Функция Println

Все аргументы, переданные этой функции, выводятся в терминале (а их значения разделяются пробелами).


После вывода всех аргументов Println переходит на следующую строку в терминале. (Отсюда суффикс «ln» — сокращение от «line» — в конце имени.)

```
fmt.Println("First argument", "Second argument")  
fmt.Println("Another line")
```


Результат. —→

```
First argument Second argument  
Another line
```

Использование функций из других пакетов

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

Пакет «fmt» необходимо импортировать, чтобы вызвать его функцию Println.

 Сообщает, что вызываемая функция является частью пакета «fmt».

Записать Использование функций из других пакетов

После того как пакет будет импортирован, вы сможете вызывать функции из этого пакета. Для этого укажите имя пакета, поставьте точку (.) и введите имя нужной функции.

Имя пакета. Имя функции.

```
fmt.Println()
```

Записать Использование функций из других пакетов

После того как пакет будет импортирован, вы сможете вызывать функции из этого пакета. Для этого укажите имя пакета, поставьте точку (.) и введите имя нужной функции.

Имя пакета.

Имя функции.

```
fmt.Println()
```

The diagram illustrates the components of the function call `fmt.Println()`. The text `fmt` is enclosed in a box, with a line pointing to it from the label "Имя пакета." (Package name). The text `.Println()` is enclosed in a box, with a line pointing to it from the label "Имя функции." (Function name).

Использование функций из других пакетов

```
package main
import (
    "math"
    "strings"
)
```

Альтернативный формат инструкции `<import>` позволяет импортировать сразу несколько пакетов.

Импортируем пакет `<math>`, чтобы использовать функцию `math.Floor`.

Импортируем пакет `<strings>`, чтобы использовать функцию `strings.Title`.

```
func main() {
    math.Floor(2.75)
    strings.Title("head first go")
}
```

Вызываем функцию `Floor` из пакета `<math>`.

Вызываем функцию `Title` из пакета `<strings>`.

Программа ничего не выводит.
(Сейчас мы объясним почему!)

Записать **Использование функций из других пакетов**

Так как мы собираемся импортировать несколько пакетов, то переходим на альтернативный формат инструкции, который позволяет перечислять в круглых скобках сразу несколько пакетов, по одному имени пакета в строке.

Возвращаемые значения функций

```
package main

import (
    "math"
    "strings"
)

func main() {
    math.Floor(2.75)
    strings.Title("head first go")
}
```

*Программа ничего
не выводит!*

Записать Возвращаемые значения функций

При вызове функции `fmt.Println` вам не нужно обмениваться с ней дополнительной информацией. Вы передаете `Println` одно или несколько выводимых значений и ожидаете вывод. Но иногда программа должна вызвать функцию и получить от нее дополнительные данные. Из-за этого в большинстве языков программирования функции имеют **возвращаемые значения**, которые вычисляются функциями и возвращаются на сторону вызова.

Возвращаемые значения функций

```
package main
```

```
import (
```

```
    "fmt"
```

```
    "math"
```

```
    "strings"
```

```
)
```

```
func main() {
```

```
    fmt.Println(math.Floor(2.75))
```

```
    fmt.Println(strings.Title("lyc lyc golang lyc"))
```

```
}
```

Функция `fmt.Println` вызывается для возвращаемого значения функции `math.Floor`.

Функция `fmt.Println` вызывается для возвращаемого значения функции `strings.Title`.

← Также импортируется пакет «fmt».

Получает число, округляет его в меньшую сторону и возвращает полученное значение.

Результат.

2

Lyc Lyc Golang Lyc

Получает строку и возвращает новую строку, в которой все слова начинаются с буквы верхнего регистра.

У бассейна



Выловите из бассейна фрагменты кода и разместите их в пустых строках кода. Каждый фрагмент может использоваться **только один** раз; использовать все фрагменты необязательно. Ваша **задача**: построить код, который будет успешно выполняться и выведет показанный результат.

```
package main ← Мы вставили первый  
                фрагмент за вас!  
import (  
    _____  
)  
  
_____ main() {  
    fmt.Println(_____)  
}
```

Результат.

лицей 79

Примечание: каждый предмет из бассейна может использоваться только один раз!

