Peter the Great
Saint-Petersburg Polytechnic University

Циклы. Условные конструкции. Исключения.

Исполнитель: Квентин Тарантино

06.07.1995

POLYTECH

Циклы.

- можно так

```
func main() {
    numbers := []int{7, 9, 1, 2, 4, 5}

    for i := 0; i < len(numbers); i++ {
        fmt.Println(numbers[i])

    }
}
```

Циклы.

- можно так

```
func main() {
    numbers := []int{7, 9, 1, 2, 4, 5}

    for index, a := range numbers {
        fmt.Println(index, a)
    }

}
```

Циклы.

- Но есть нюанс

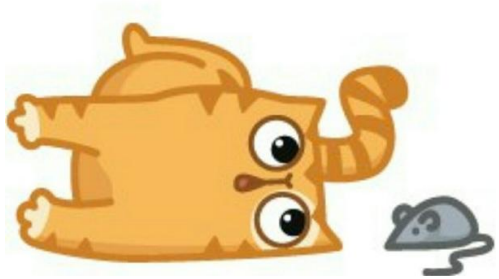
```
func main() {
    word := "+вайб"

    for index, a := range word {
        fmt.Println(index, a)
    }
}
```

Циклы.

- Но есть нюанс

```
0  43
1  1074
3  1072
5  1081
7  1073
```



```
0  +
1  в
3  а
5  й
7  б
```

Циклы.

- Спокойно.

```
func main() {
    word := "+вайб"

    for index, a := range word {
        fmt.Println(index, string(a))
    }
}
```

Циклы.

- Ничего нового.

```go
func main() {
    word := "-vibe"

    for i := 0; i < len(word); i++ {
        fmt.Println(i, string(word[i]))
    }
}
```

Циклы.

- Словарь

```go
func main() {
    books := map[string]int{
        "maths":     5,
        "biology":   9,
        "chemistry": 6,
        "physics":   3,
    }
    for key, val := range books {
        fmt.Println(key, val)
    }
}
```

```
maths 5
biology 9
chemistry 6
physics 3
```

Циклы.

- А что-то своё можно ?

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

type Person struct {
    Name   string
    Age    int
    Gender string
    Single bool
}
```

Можно!

Структуры

Можно!

```go
func main() {
    ubay := Person{
        Name:   "Nastya",
        Gender: "2",
        Age:    18,
        Single: True,
    }
    values := reflect.ValueOf(ubay)
    types := values.Type()
    for i := 0; i < values.NumField(); i++ {
        fmt.Println(types.Field(i).Index[0], types.Field(i).Name, values.Field(i))
    }
}
```

```
0 Name Nastya
1 Age 18
2 Gender 2
3 Single true
```

Структуры

Можно!

```
func main() {
    ubay := Person{
        Name:   "John",
        Gender: "Female",
        Age:    17,
        Single: false,
    }
    values := reflect.ValueOf(ubay)
    fmt.Println(values)
    types := values.Type()
    fmt.Println(types)
```

```
{John 17 Female false}
main.Person
```

Условные конструкции. Первый способ. If / Else

```
fmt.Println("If / Else:")
a := 8
b := 8
if a < b {
    fmt.Println("a меньше b")
} else if a > b {
    fmt.Println("a больше b")
} else {
    fmt.Println("a равно b")
}
```

Условные конструкции. Первый способ. If / Else

Что можно сравнивать?

- Числовые типы
- Строки
- Булевые значения
- Указатели (на один / не один элемент)
- Интерфейсы
- Срезы (slice), массивы (array) и карты (map)
- Значения nil

Условные конструкции. Первый способ. If / Else

```
if true {
    // Этот блок кода будет выполнен, так как условие истинно.
}

if x > 10 {
    // Этот блок кода выполнится, если x больше 10.
}

if isValid() {
    // Этот блок кода выполнится, если функция isValid() вернет true.
}
```

Условные конструкции. Второй способ. Switch

```
fmt.Println("Switch:")
a = 87
switch(a) {
    case 9: fmt.Println("a = 9")
    case 8: fmt.Println("a = 8")
    case 7: fmt.Println("a = 7")
    case 6, 5, 4:
        fmt.Println("a = 6 или 5 или 4")
    default:
        fmt.Println("значение переменной a другое")
}
```

Исключения. Без try / catch.

Прежде чем мы сможем обработать ошибку, нам нужно ее создать.

- errors.New
- fmt.Errorf

```
func main() {
        err := errors.New("barnacles")
        fmt.Println("Sammy says:", err)
}
```

```
Output

Sammy says: barnacles
```

Исключения. Errorf.

```
func main() {
        err := fmt.Errorf("error occurred at: %v", time.Now())
        fmt.Println("An error happened:", err)
}
```

```
Output
An error happened: Error occurred at: 2019-07-11 16:52:42.532621 -0400 EDT m=+0.000137103
```

Исключения. Делай как надо, и будет как надо.

```
func capitalize(name string) (string, error) {
        if name == "" {
                return "", errors.New("no name provided")
        }
        return strings.ToTitle(name), nil
}
```

```
name, err := capitalize("sammy")
if err != nil {
        fmt.Println("Could not capitalize:", err)
        return
}

fmt.Println("Capitalized name:", name)
```