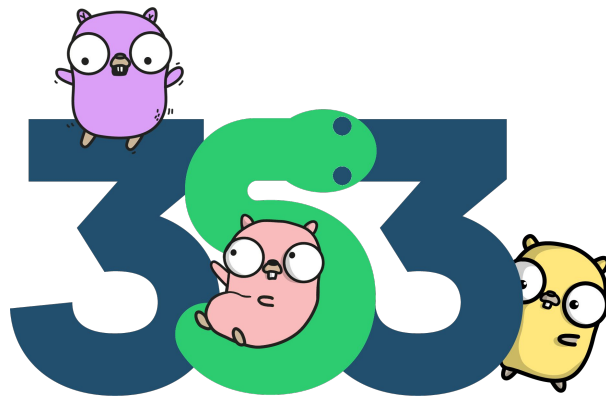


One Does Not  
Simply Iterate  
over Mordor

# miki @tebeka



CEO,  
CTO,  
UFO



Idiotic  
Programmers

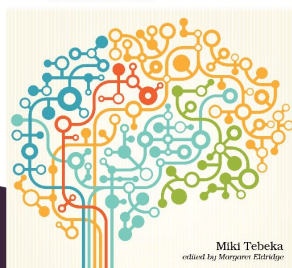
**Effective Go Recipes**  
Fast Solutions to Common Tasks



Miki Tebeka  
edited by Margaret Eide

Idiotic  
Programmers

**Go Brain Teasers**  
Exercise Your Mind



Miki Tebeka  
edited by Margaret Eide

Idiotic  
Programmers

**Python Brain Teasers**  
Exercise Your Mind



Miki Tebeka  
edited by Margaret Eide

Idiotic  
Programmers

**Pandas Brain Teasers**  
Exercise Your Mind



Miki Tebeka  
edited by Margaret Eide



# Range Over Function

Go 1.23, August 2024

# Generators

Python 2.3, July 2003



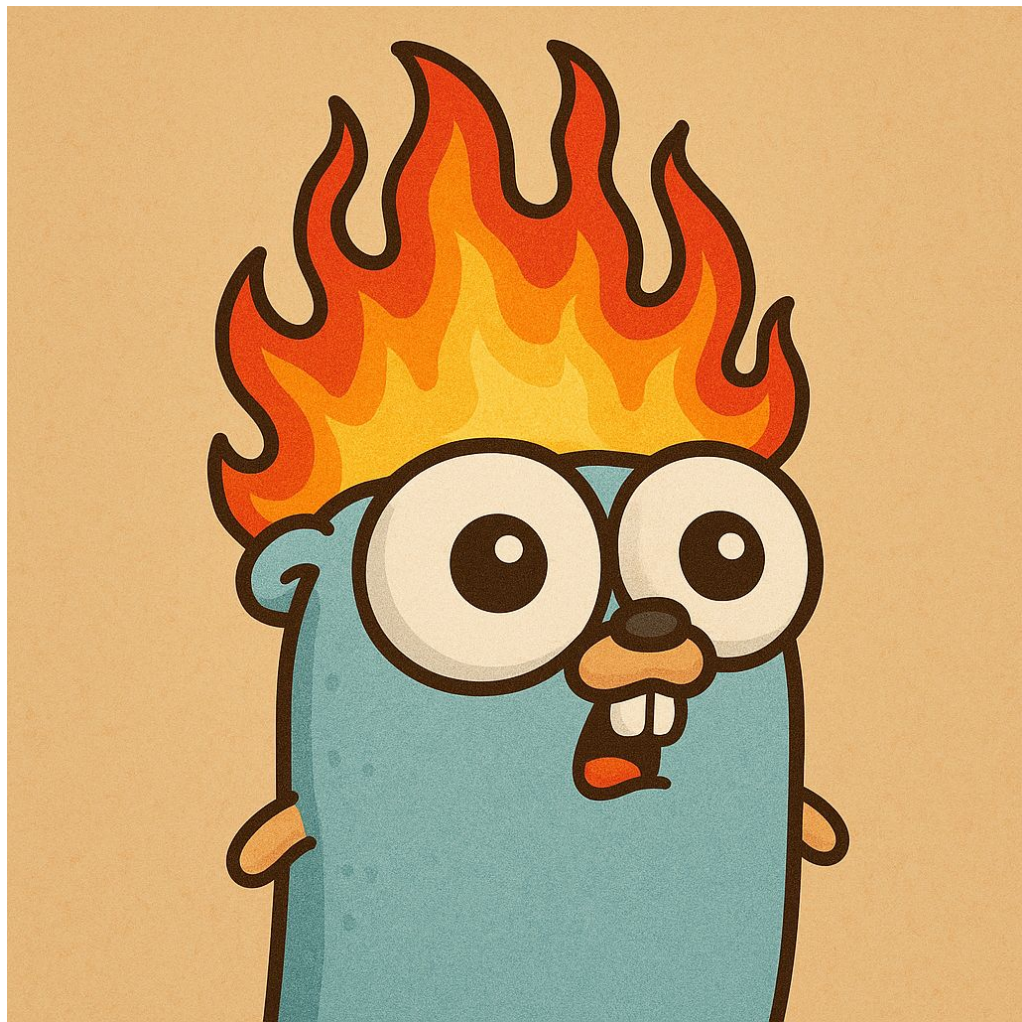
## type Seq

```
type Seq[V any] func(yield func(V) bool)
```

## type Seq2

```
type Seq2[K, V any] func(yield func(K, V) bool)
```

[pkg.go.dev/iter](http://pkg.go.dev/iter)





# CODE

`tick/tick.go`



# Range-Over Functions in Go ArdanLab Blog



Load logs from  
compressed  
JSON file.

```
$ zcat logs.json.gz | wc
```

```
5350200 16045852 770589508
```

```
$ zcat logs.json.gz | head -1
```

```
{"host": "199.72.81.55", "time":  
"1995-07-01T00:00:01Z", "request": "GET /history/apollo/  
HTTP/1.0", "status": 200, "bytes": 6245}
```

# CODE

`log/log.go`

# CODE

`slice/slice.go`

# CODE

load

# Map

lower: ["Suite", "Up"] -> ["suite", "up"]

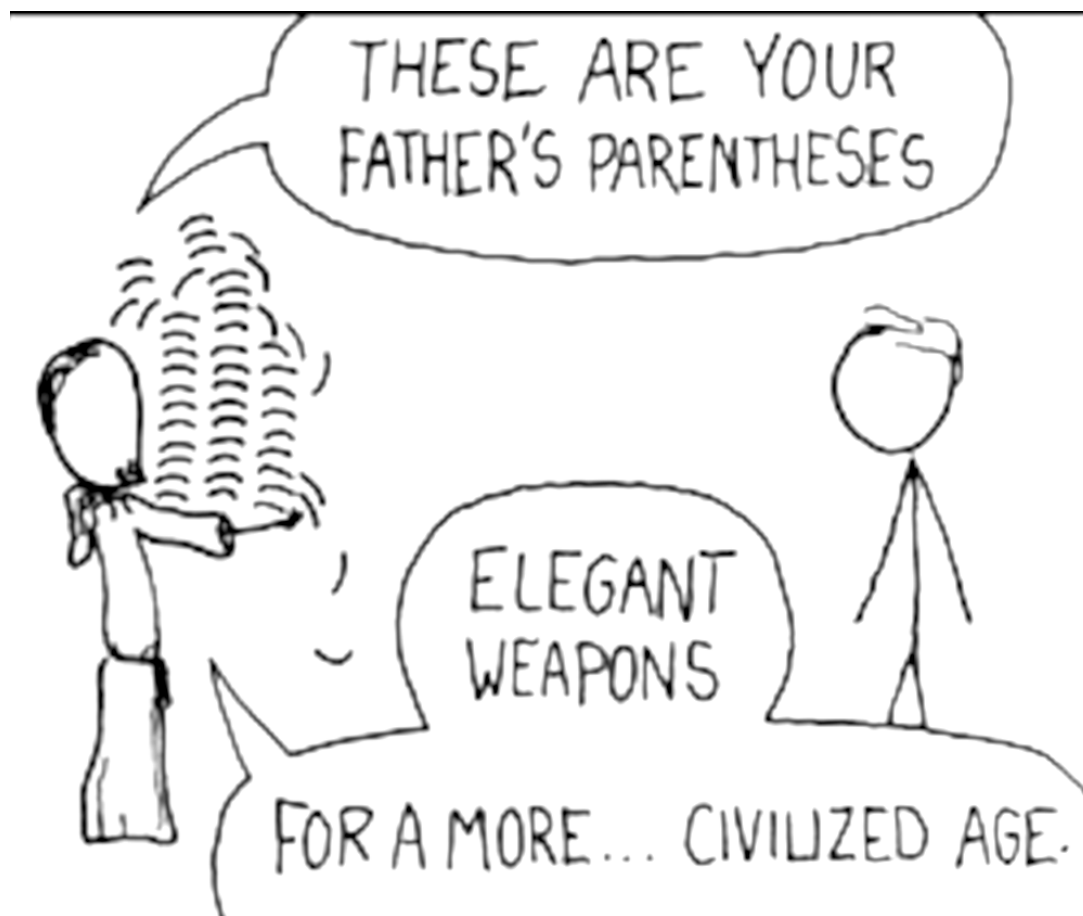
# Filter

$\geq 0$ : [2, -1, 3, 4] -> [2, 3, 4]

# Reduce

sum: [1, 2, 3, 4]  $\Rightarrow$  10





<https://xkcd.com/297/>

reduce



map



```
SELECT SUM(amount * price)
FROM sales
WHERE date = '2025-05-25';
```



filter

# CODE

`lazy/lazy.go`

# CODE

push / push . go

# CODE

`pull/pull.go`



## func Collect

```
func Collect[E any](seq iter.Seq[E]) []E
```

Collect collects values from seq into a new slice and returns it.

## func Values

```
func Values[Slice ~[]E, E any](s Slice) iter.Seq[E]
```

Values returns an iterator that yields the slice elements in order.

[pkg.go.dev/slices](https://pkg.go.dev/slices)



## func Values

```
func Values[Map ~map[K]V, K comparable, V any](m Map) iter.Seq[V]
```

Values returns an iterator over values in m. The iteration order is not specific

## func All

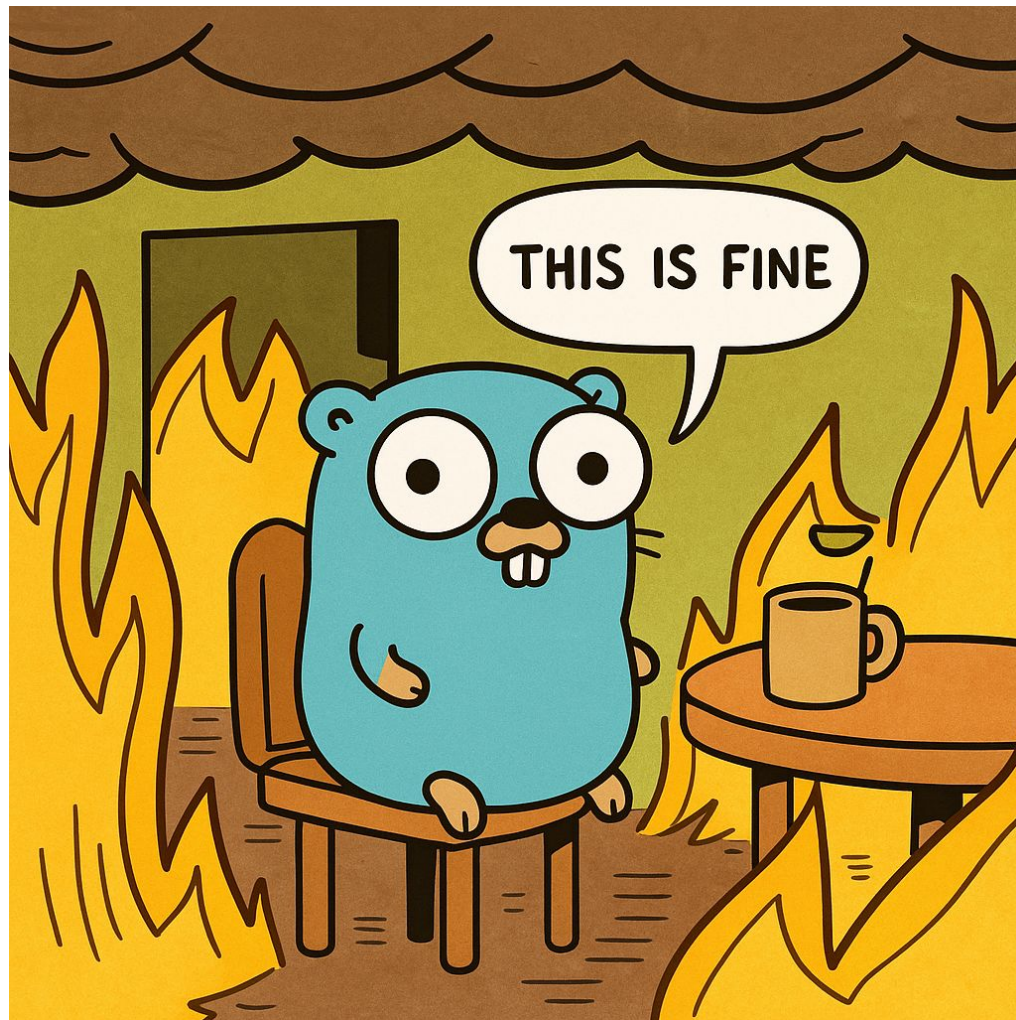
```
func All[Map ~map[K]V, K comparable, V any](m Map) iter.Seq2[K, V]
```

All returns an iterator over key-value pairs from m. The iteration order is not

[pkg.go.dev/maps](https://pkg.go.dev/maps)

| Iterator                                     | Arguments                   | Results                                         | Example                                                        |
|----------------------------------------------|-----------------------------|-------------------------------------------------|----------------------------------------------------------------|
| <a href="#"><u>accumulate()</u></a>          | p [,func]                   | p0, p0+p1, p0+p1+p2, ...                        | <code>accumulate([1,2,3,4,5])</code> → 1 3 6 10 15             |
| <a href="#"><u>batched()</u></a>             | p, n                        | (p0, p1, ..., p_n-1), ...                       | <code>batched('ABCDEFGG', n=3)</code> → ABC DEF G              |
| <a href="#"><u>chain()</u></a>               | p, q, ...                   | p0, p1, ... plast, q0, q1, ...                  | <code>chain('ABC', 'DEF')</code> → A B C D E F                 |
| <a href="#"><u>chain.from_iterable()</u></a> | iterable                    | p0, p1, ... plast, q0, q1, ...                  | <code>chain.from_iterable(['ABC', 'DEF'])</code> → A B C D E F |
| <a href="#"><u>compress()</u></a>            | data, selectors             | (d[0] if s[0]), (d[1] if s[1]), ...             | <code>compress('ABCDEF', [1,0,1,0,1,1])</code> → A C E F       |
| <a href="#"><u>dropwhile()</u></a>           | predicate, seq              | seq[n], seq[n+1], starting when predicate fails | <code>dropwhile(lambda x: x&lt;5, [1,4,6,3,8])</code> → 6 3 8  |
| <a href="#"><u>filterfalse()</u></a>         | predicate, seq              | elements of seq where predicate(elem) fails     | <code>filterfalse(lambda x: x&lt;5, [1,4,6,3,8])</code> → 6 8  |
| <a href="#"><u>groupby()</u></a>             | iterable[, key]             | sub-iterators grouped by value of key(v)        | <code>groupby(['A','B','DEF'], len)</code> → (1, A B) (3, DEF) |
| <a href="#"><u>islice()</u></a>              | seq, [start,] stop [, step] | elements from seq[start:stop:step]              | <code>islice('ABCDEFGG', 2, None)</code> → C D E F G           |

[docs.python.org/3/library/itertools.html](https://docs.python.org/3/library/itertools.html)



# CODE

`fibs/fibs.go`

# CODE

`order/order.go`

# CODE

`leak/leak.go`

# CODE

`range/range.go`





# Questions?



[github.com/tebeka/talks/tree/main/gcil-2025](https://github.com/tebeka/talks/tree/main/gcil-2025)