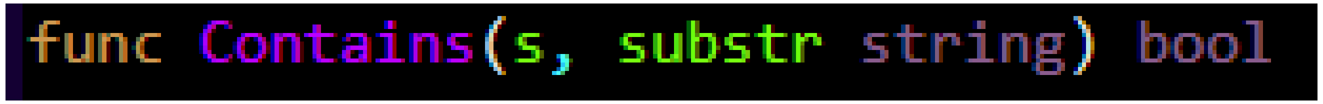
Go Training

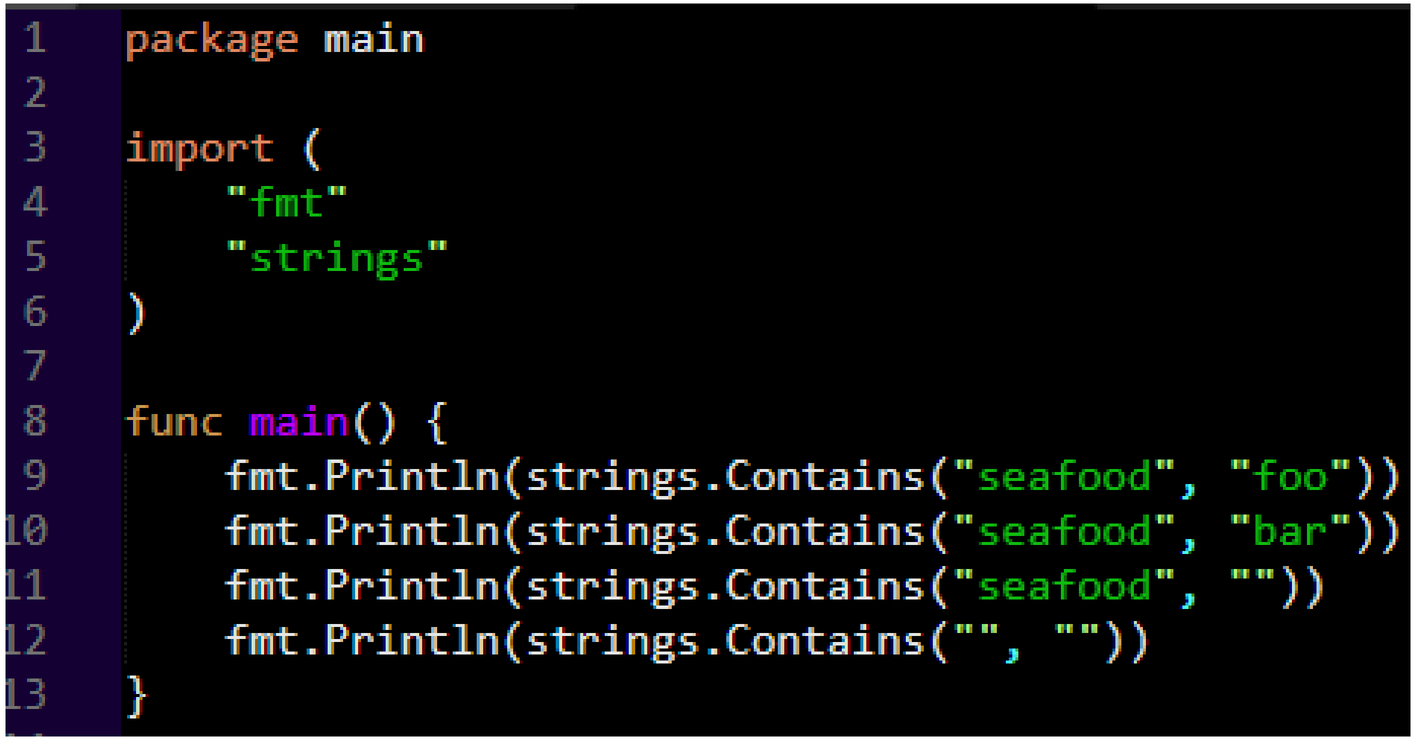
Session 11

strings package

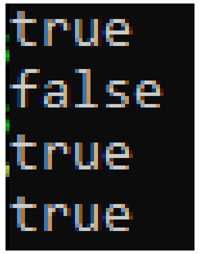
- https://golang.org/pkg/strings/
- Import “strings”



```
func Contains(s, substr string) bool
```

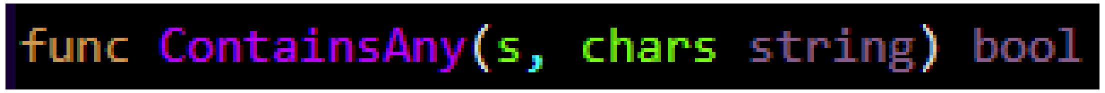


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	fmt.Println(strings.Contains("seafood", "foo"))
	fmt.Println(strings.Contains("seafood", "bar"))
	fmt.Println(strings.Contains("seafood", ""))
	fmt.Println(strings.Contains("", ""))
}
```

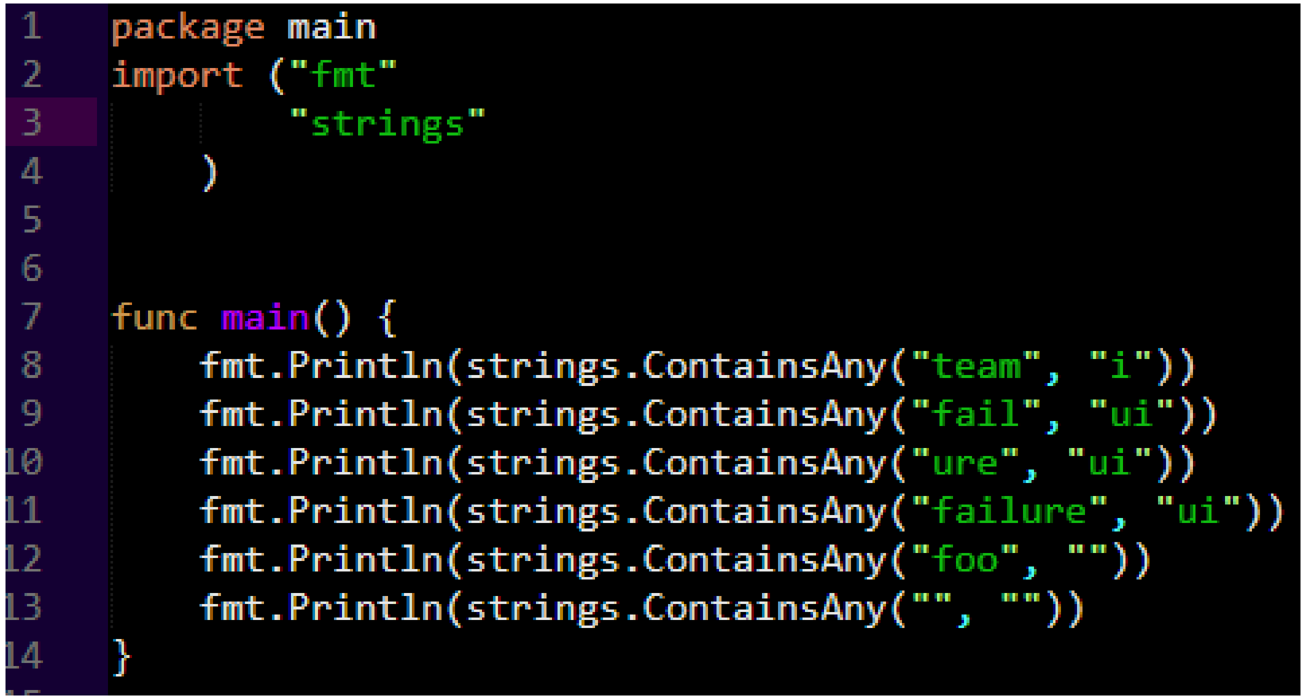


```
true
false
true
true
```

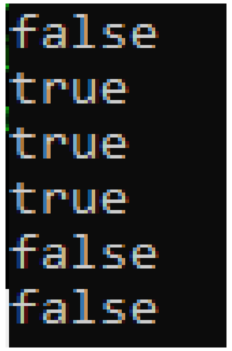
strings package



```go
func ContainsAny(s, chars string) bool
```

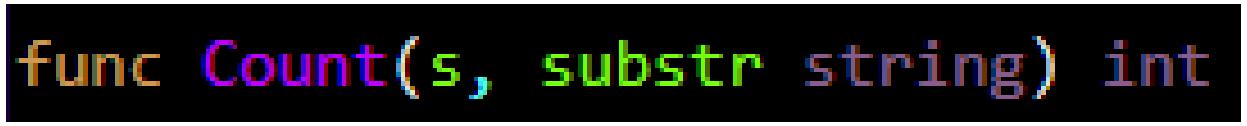


```go
package main
import ("fmt"
        "strings"
    )


func main() {
    fmt.Println(strings.ContainsAny("team", "i"))
    fmt.Println(strings.ContainsAny("fail", "ui"))
    fmt.Println(strings.ContainsAny("ure", "ui"))
    fmt.Println(strings.ContainsAny("failure", "ui"))
    fmt.Println(strings.ContainsAny("foo", ""))
    fmt.Println(strings.ContainsAny("", ""))
}
```

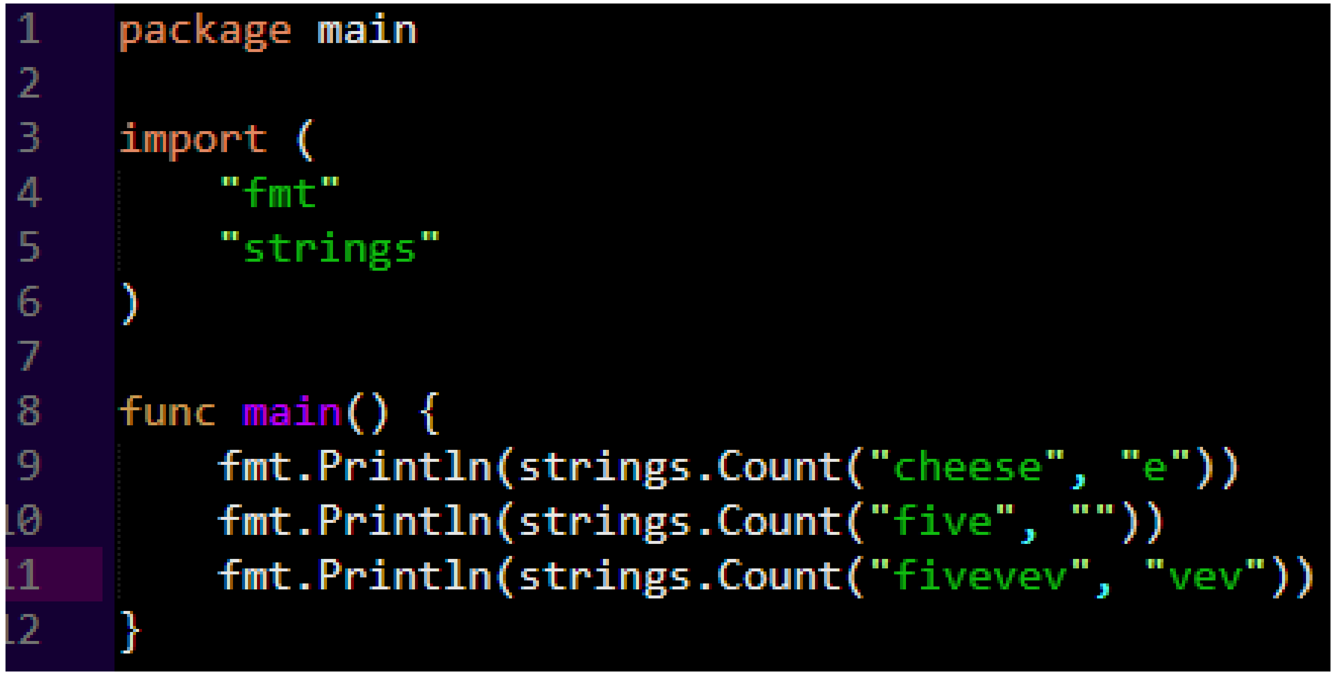


```
false
true
true
true
false
false
```

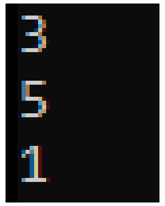
strings package



```go
func Count(s, substr string) int
```

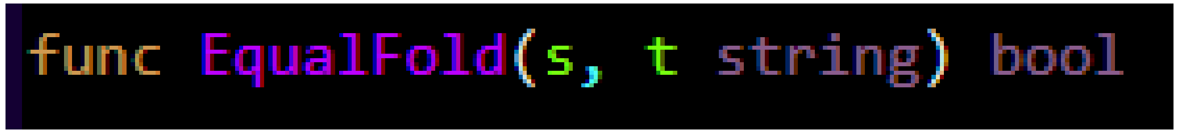


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	fmt.Println(strings.Count("cheese", "e"))
	fmt.Println(strings.Count("five", ""))
	fmt.Println(strings.Count("fivevev", "vev"))
}
```

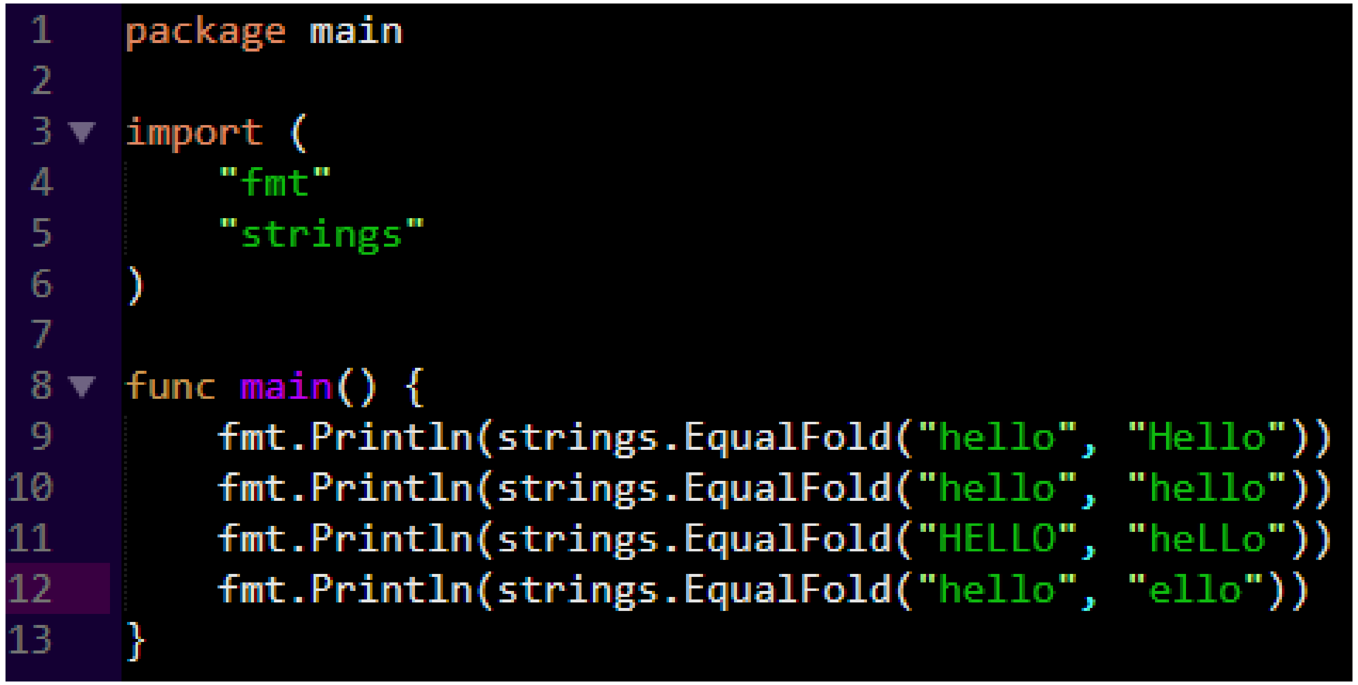


```
3
5
1
```

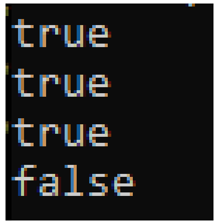
strings package



```go
func EqualFold(s, t string) bool
```

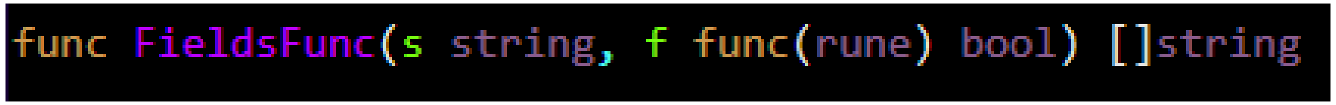


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	fmt.Println(strings.EqualFold("hello", "Hello"))
	fmt.Println(strings.EqualFold("hello", "hello"))
	fmt.Println(strings.EqualFold("HELLO", "heLLo"))
	fmt.Println(strings.EqualFold("hello", "ello"))
}
```

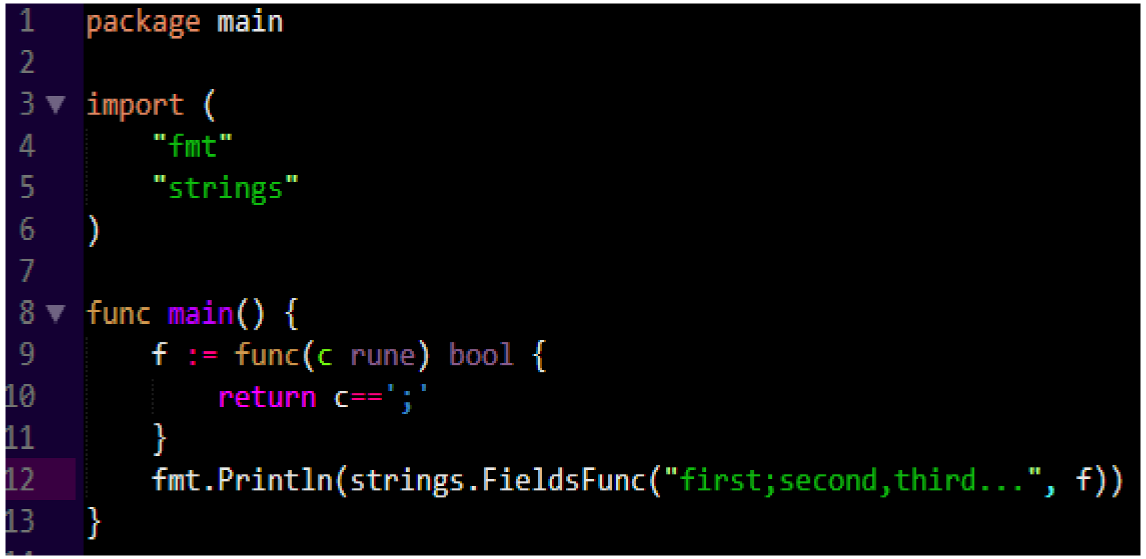


```
true
true
true
false
```

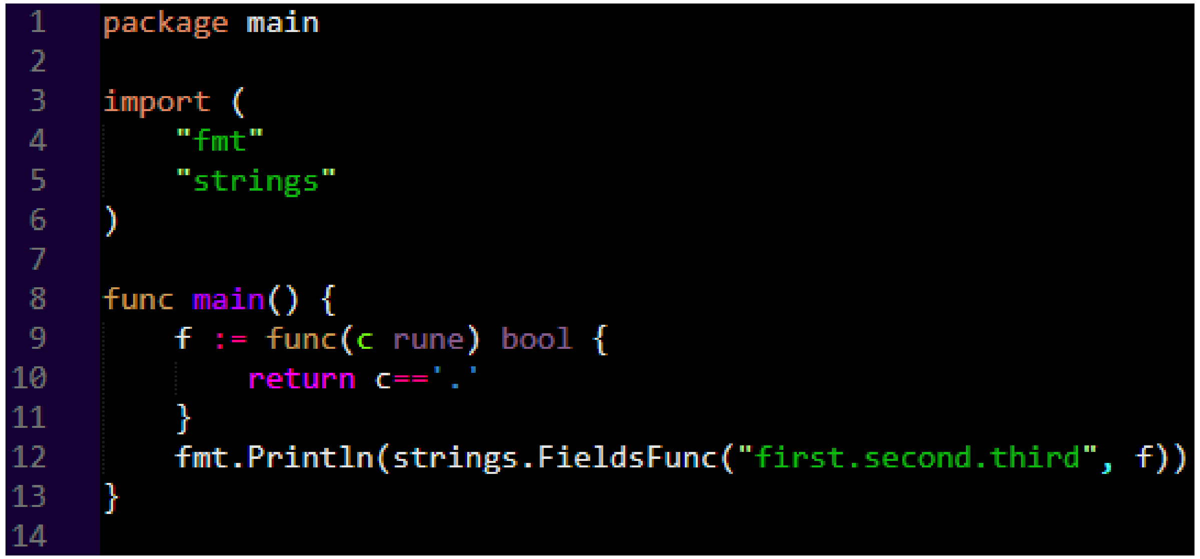
strings package



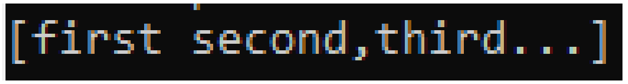
```go
func FieldsFunc(s string, f func(rune) bool) []string
```



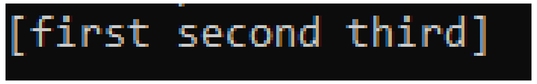
```go
package main

import (
    "fmt"
    "strings"
)

func main() {
    f := func(c rune) bool {
        return c==';'
    }
    fmt.Println(strings.FieldsFunc("first;second,third...", f))
}
```



```go
package main

import (
    "fmt"
    "strings"
)

func main() {
    f := func(c rune) bool {
        return c=='.'
    }
    fmt.Println(strings.FieldsFunc("first.second.third", f))
}
```

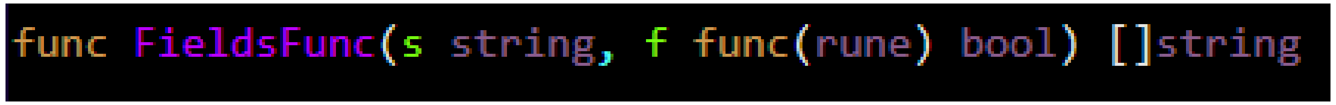


```
[first second,third...]
```

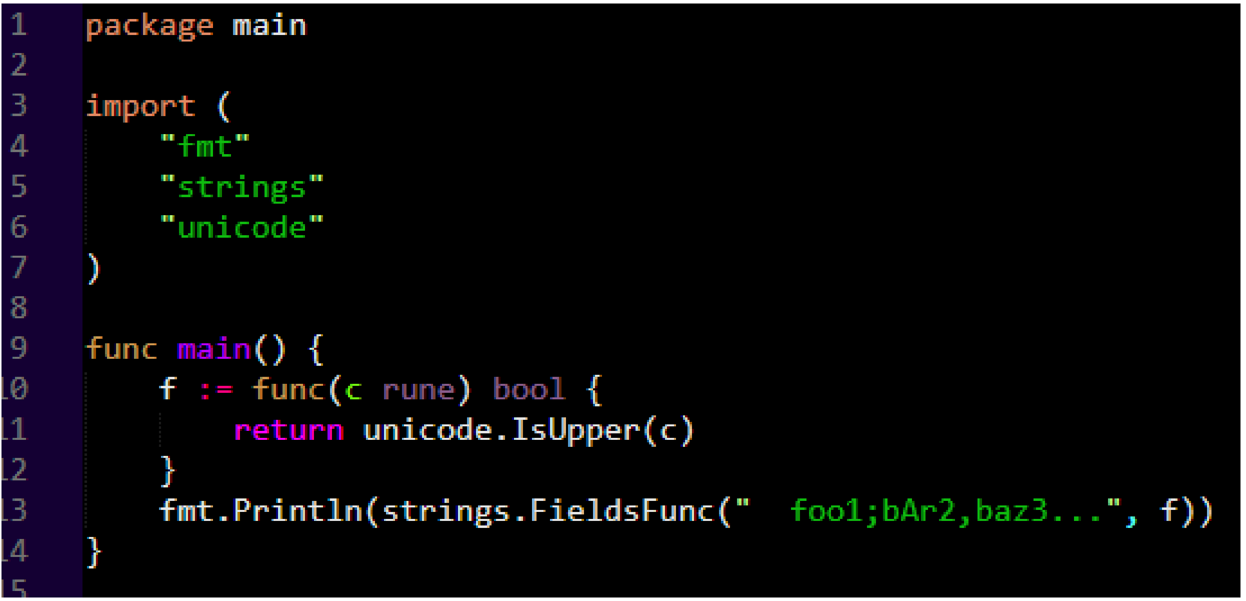


```
[first second third]
```

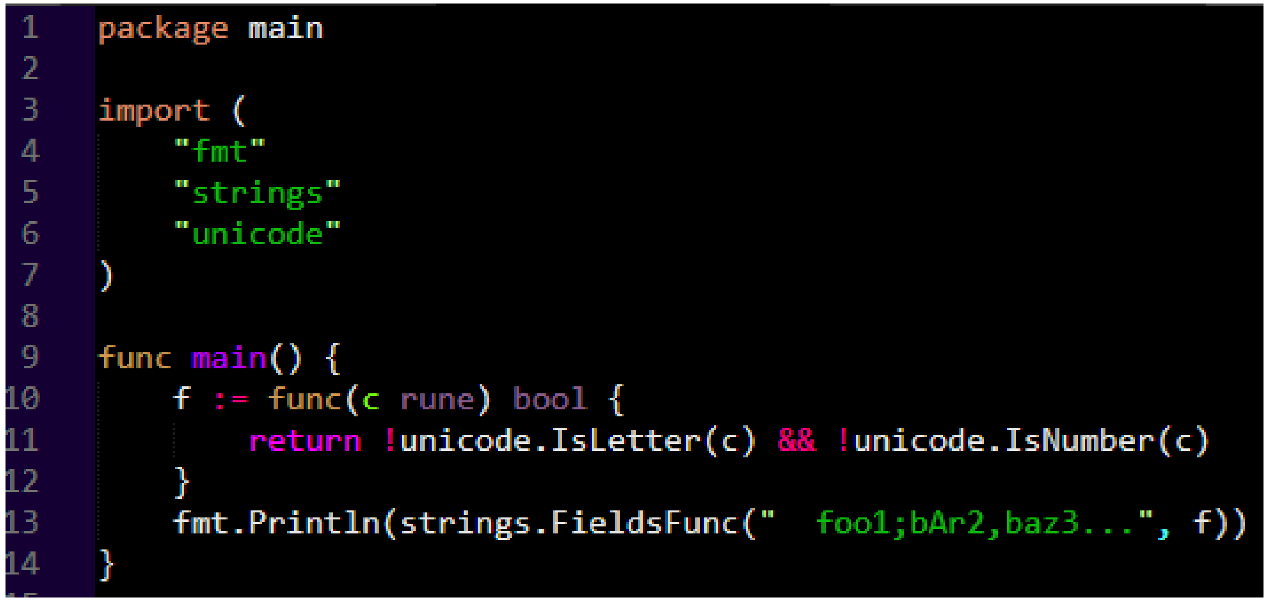
strings package



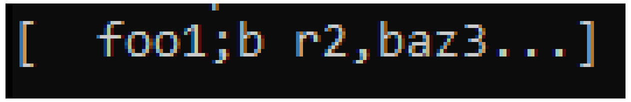
```go
func FieldsFunc(s string, f func(rune) bool) []string
```



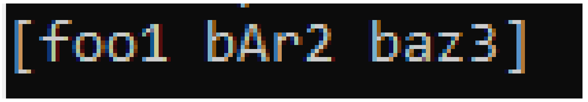
```go
package main

import (
	"fmt"
	"strings"
	"unicode"
)

func main() {
	f := func(c rune) bool {
		return unicode.IsUpper(c)
	}
	fmt.Println(strings.FieldsFunc("  foo1;bAr2,baz3...", f))
}
```



```go
package main

import (
	"fmt"
	"strings"
	"unicode"
)

func main() {
	f := func(c rune) bool {
		return !unicode.IsLetter(c) && !unicode.IsNumber(c)
	}
	fmt.Println(strings.FieldsFunc("  foo1;bAr2,baz3...", f))
}
```

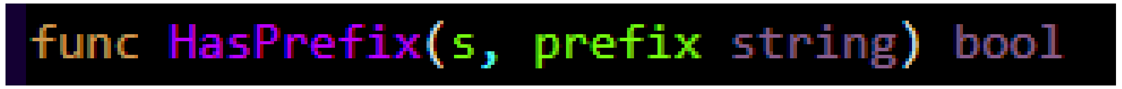


```
[  foo1;b r2,baz3...]
```

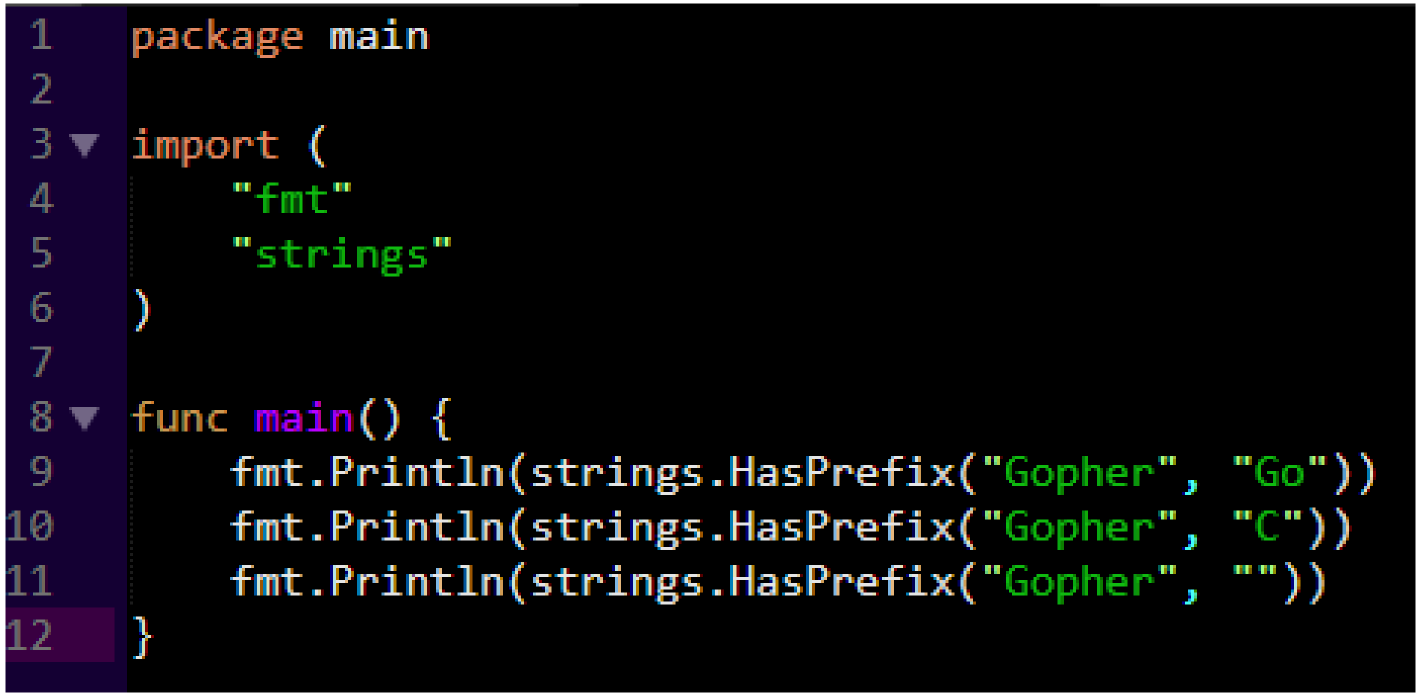


```
[foo1 bAr2 baz3]
```

strings package

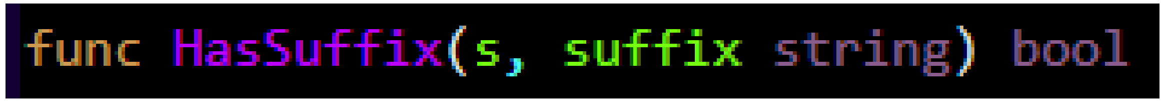


```
func HasPrefix(s, prefix string) bool
```

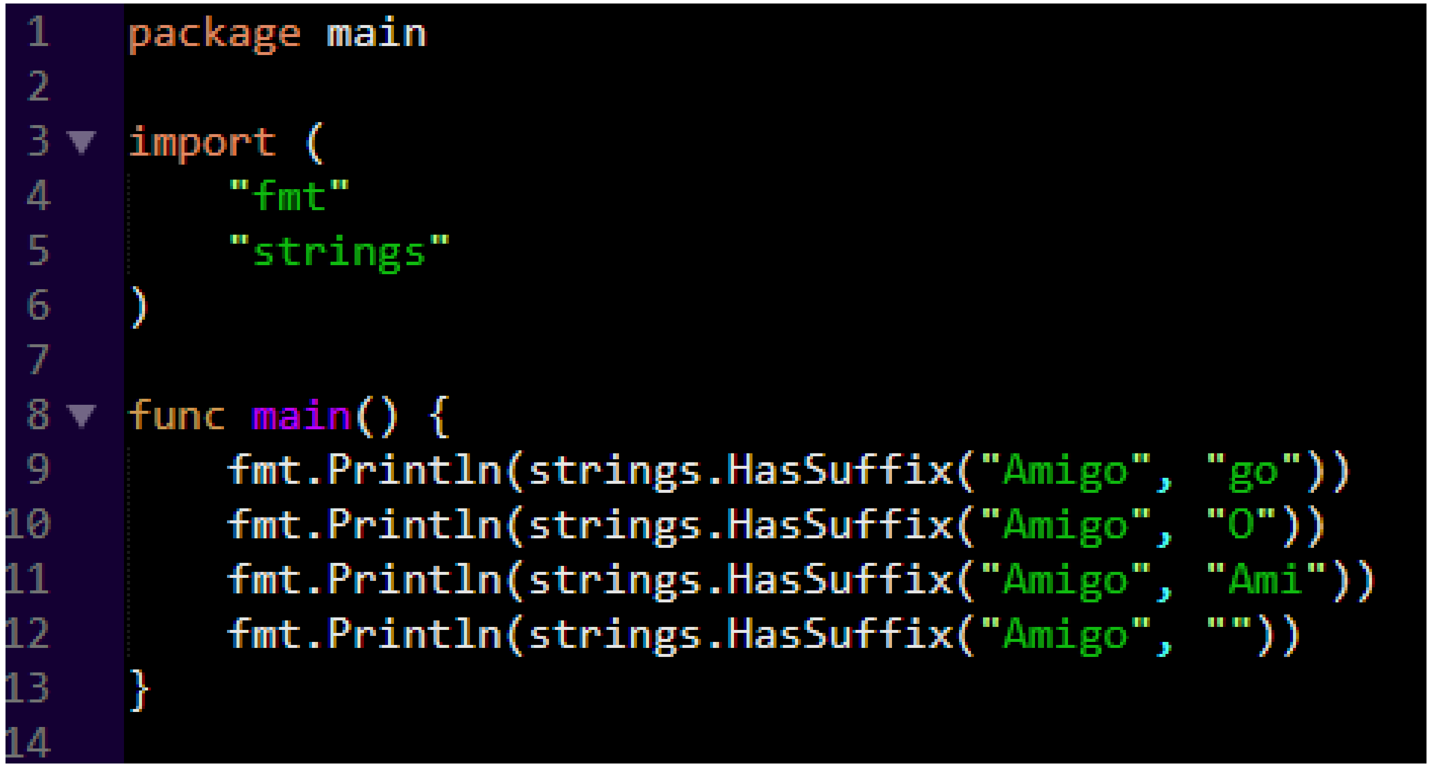


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	fmt.Println(strings.HasPrefix("Gopher", "Go"))
	fmt.Println(strings.HasPrefix("Gopher", "C"))
	fmt.Println(strings.HasPrefix("Gopher", ""))
}
```

strings package

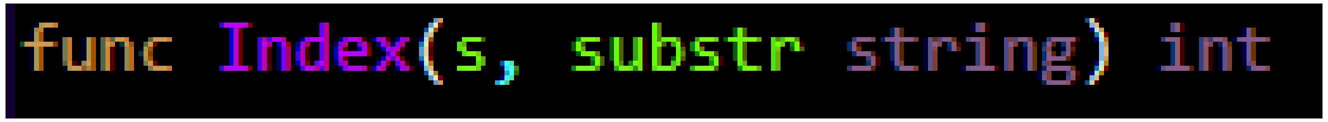


```go
func HasSuffix(s, suffix string) bool
```

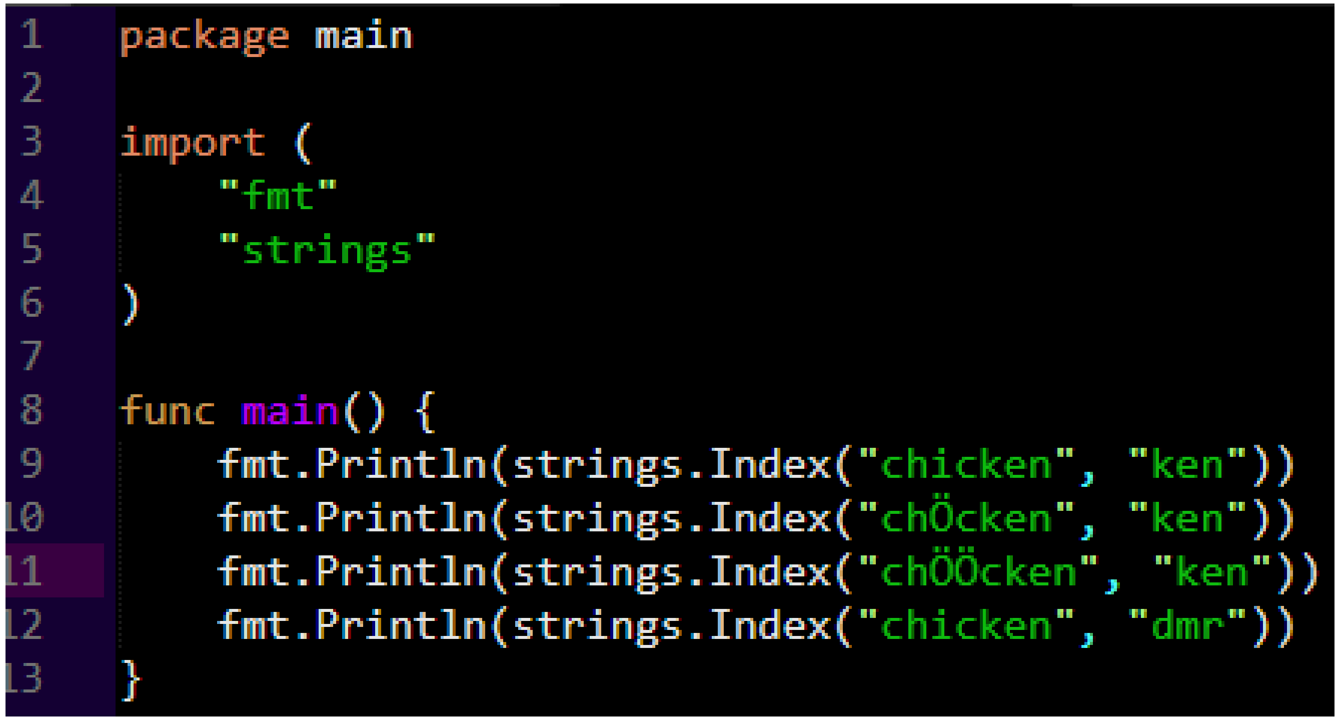


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	fmt.Println(strings.HasSuffix("Amigo", "go"))
	fmt.Println(strings.HasSuffix("Amigo", "O"))
	fmt.Println(strings.HasSuffix("Amigo", "Ami"))
	fmt.Println(strings.HasSuffix("Amigo", ""))
}
```

strings package

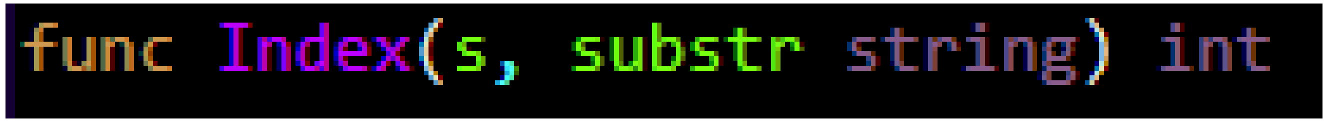


```go
func Index(s, substr string) int
```

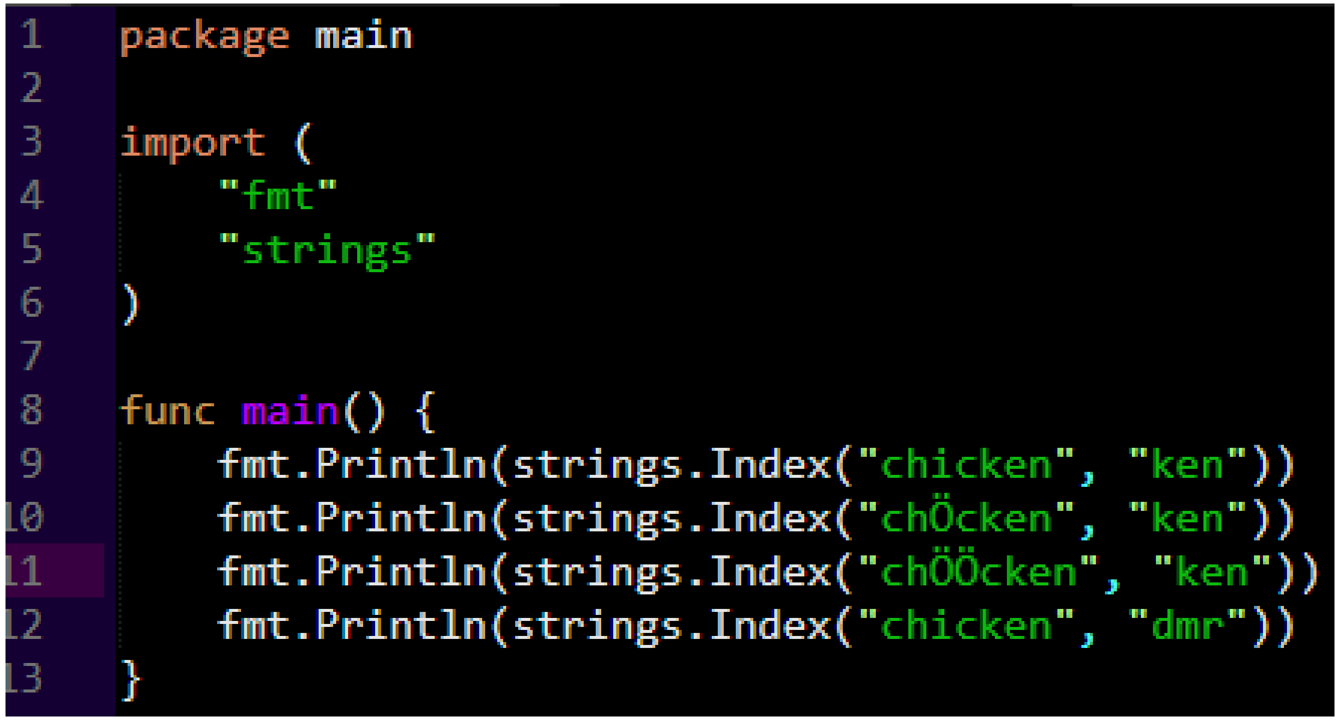


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	fmt.Println(strings.Index("chicken", "ken"))
	fmt.Println(strings.Index("chÖcken", "ken"))
	fmt.Println(strings.Index("chÖÖcken", "ken"))
	fmt.Println(strings.Index("chicken", "dmr"))
}
```

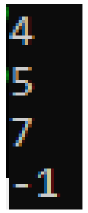
strings package



```go
func Index(s, substr string) int
```

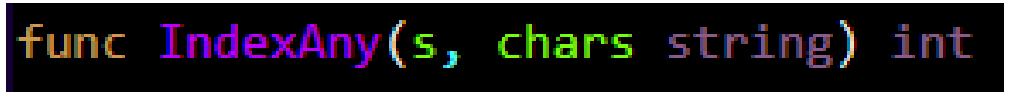


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	fmt.Println(strings.Index("chicken", "ken"))
	fmt.Println(strings.Index("chÖcken", "ken"))
	fmt.Println(strings.Index("chÖÖcken", "ken"))
	fmt.Println(strings.Index("chicken", "dmr"))
}
```

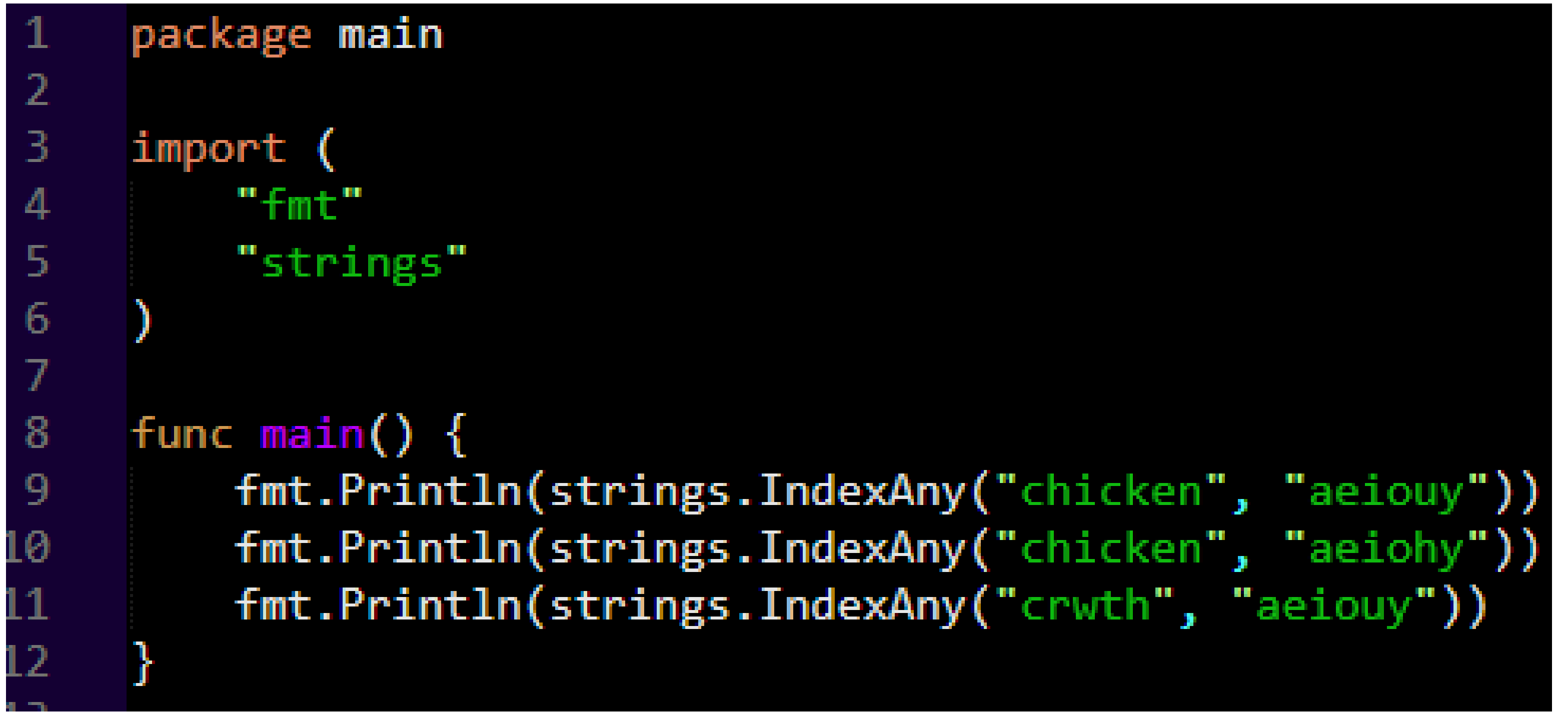


```
4
5
7
-1
```

strings package

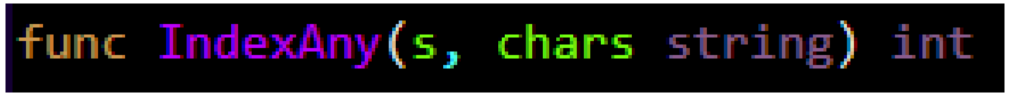


```go
func IndexAny(s, chars string) int
```

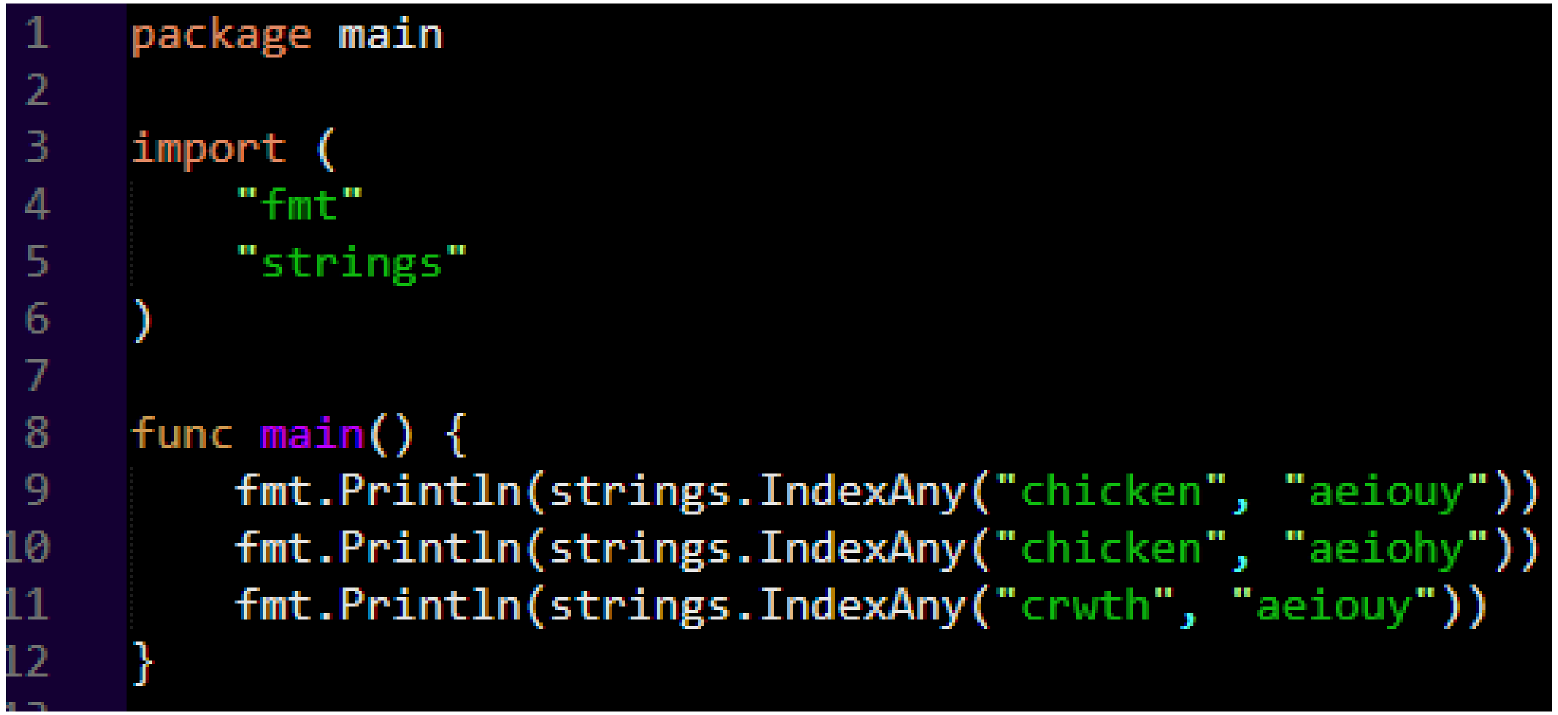


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	fmt.Println(strings.IndexAny("chicken", "aeiouy"))
	fmt.Println(strings.IndexAny("chicken", "aeiohy"))
	fmt.Println(strings.IndexAny("crwth", "aeiouy"))
}
```

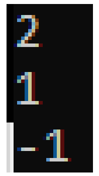
strings package



```go
func IndexAny(s, chars string) int
```

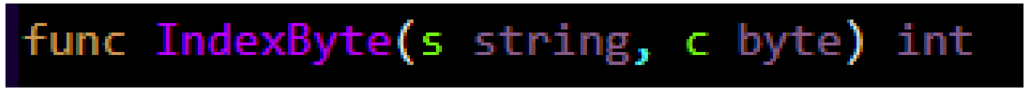


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	fmt.Println(strings.IndexAny("chicken", "aeiouy"))
	fmt.Println(strings.IndexAny("chicken", "aeiohy"))
	fmt.Println(strings.IndexAny("crwth", "aeiouy"))
}
```

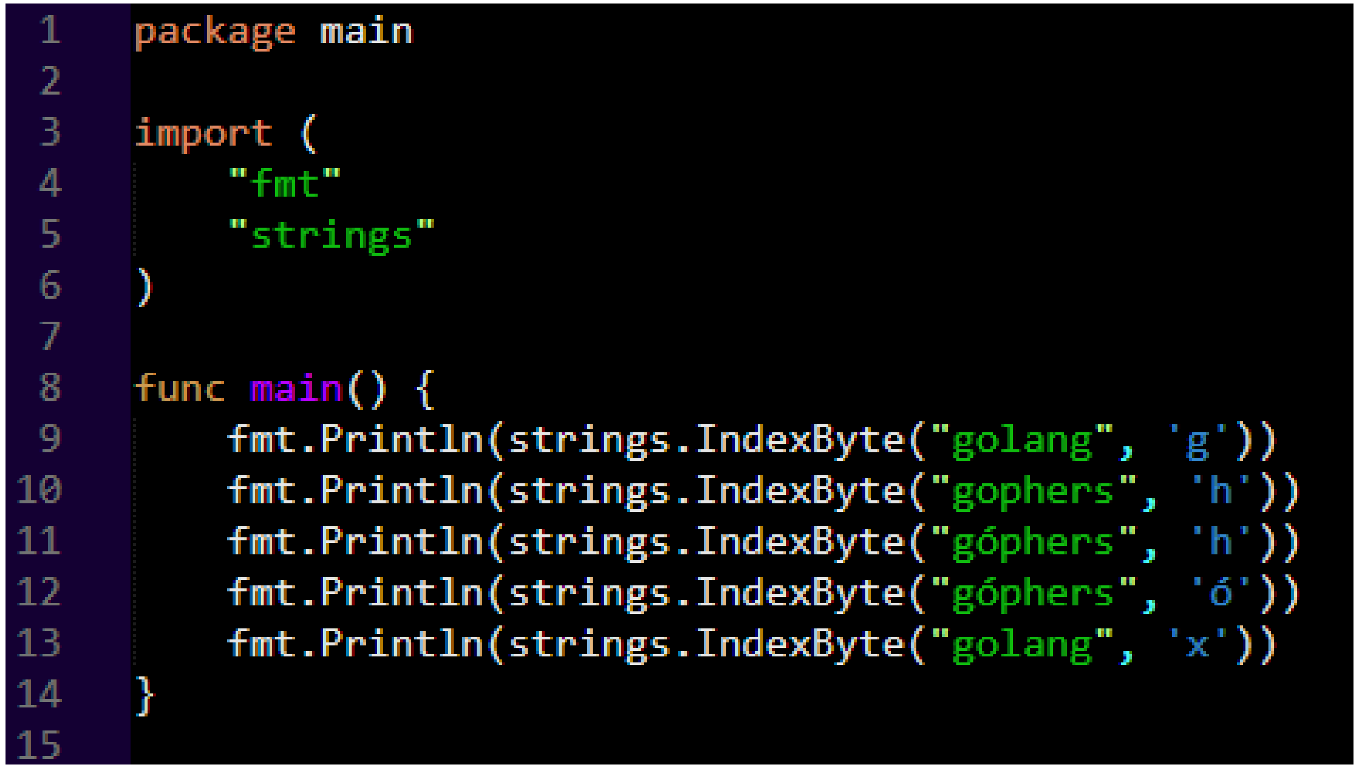


```
2
1
-1
```

strings package

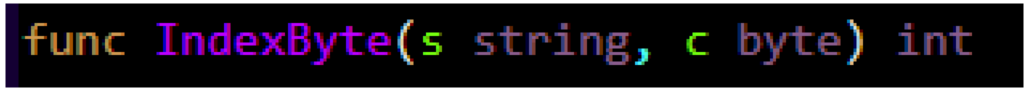


```go
func IndexByte(s string, c byte) int
```

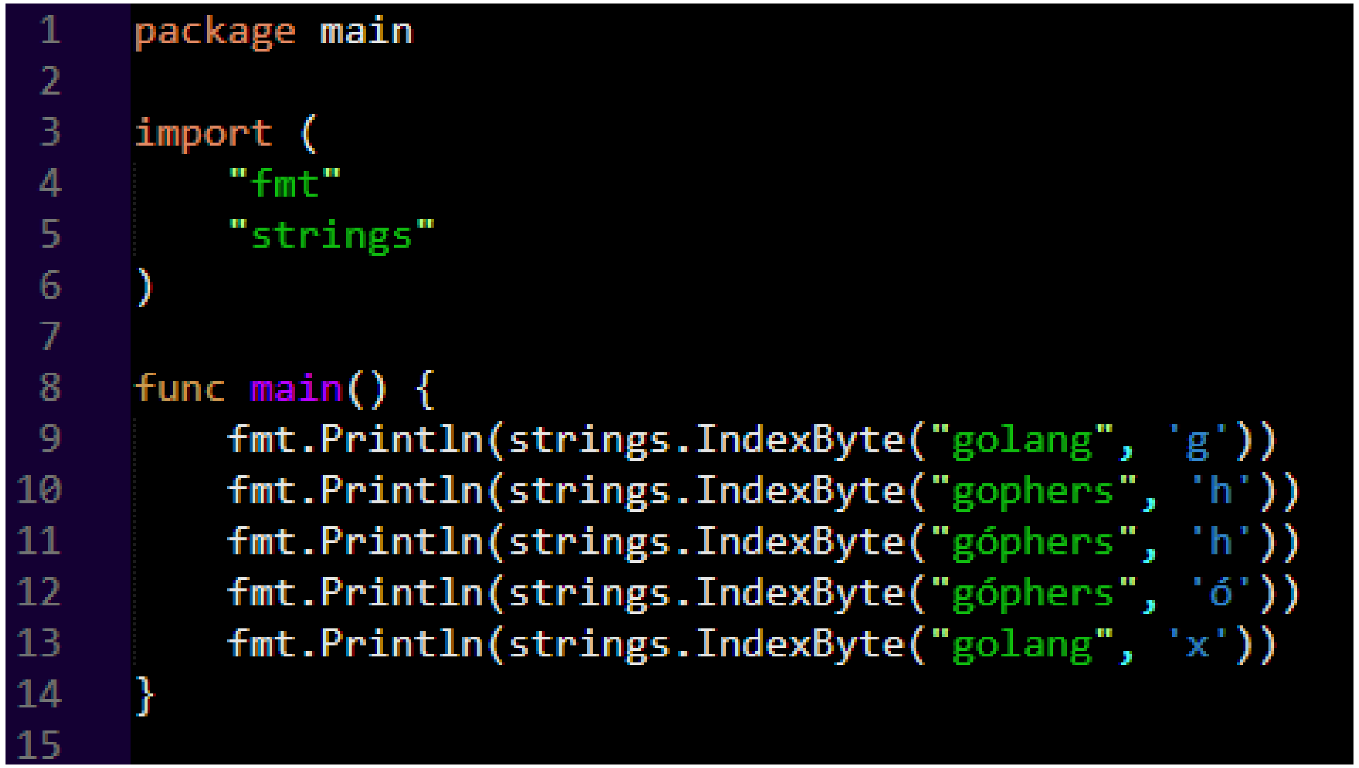


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	fmt.Println(strings.IndexByte("golang", 'g'))
	fmt.Println(strings.IndexByte("gophers", 'h'))
	fmt.Println(strings.IndexByte("gópher s", 'h'))
	fmt.Println(strings.IndexByte("góphers", 'ó'))
	fmt.Println(strings.IndexByte("golang", 'x'))
}

```

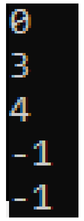
strings package



```go
func IndexByte(s string, c byte) int
```

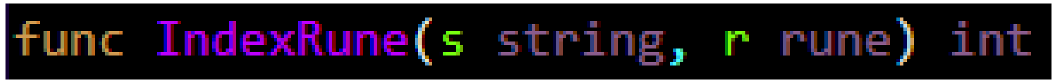


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	fmt.Println(strings.IndexByte("golang", 'g'))
	fmt.Println(strings.IndexByte("gophers", 'h'))
	fmt.Println(strings.IndexByte("góphers", 'h'))
	fmt.Println(strings.IndexByte("góphers", 'ó'))
	fmt.Println(strings.IndexByte("golang", 'x'))
}
```

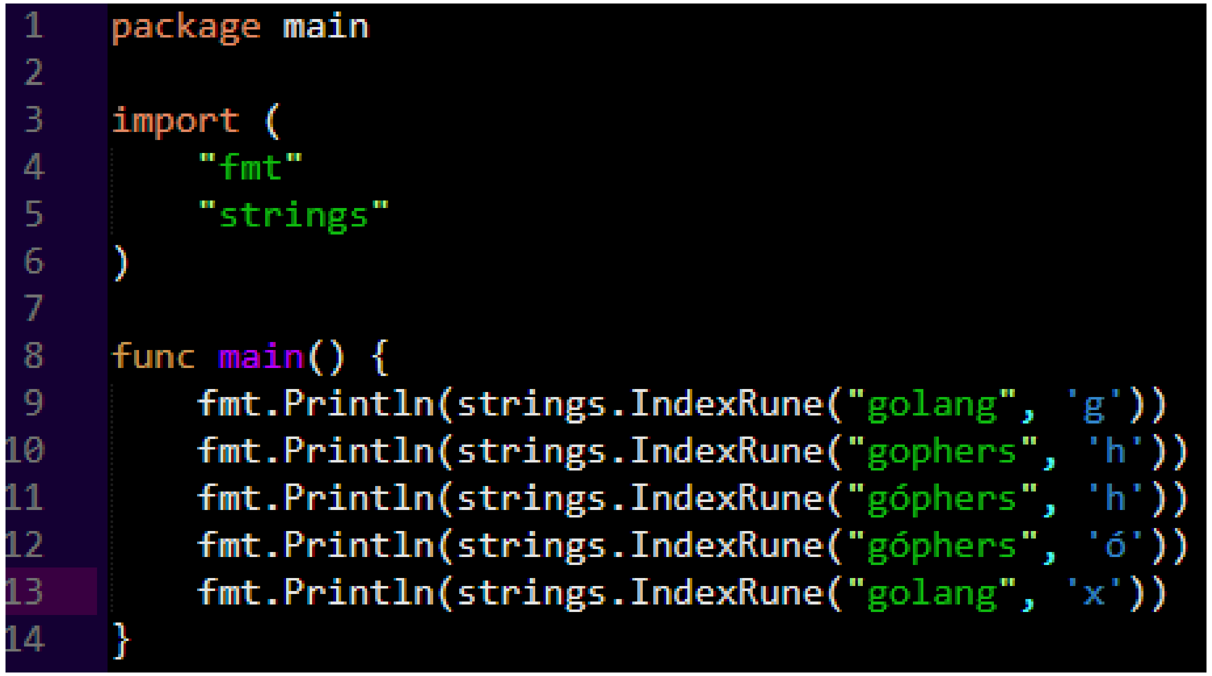


```
0
3
4
-1
-1
```

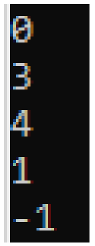
strings package



```go
func IndexRune(s string, r rune) int
```

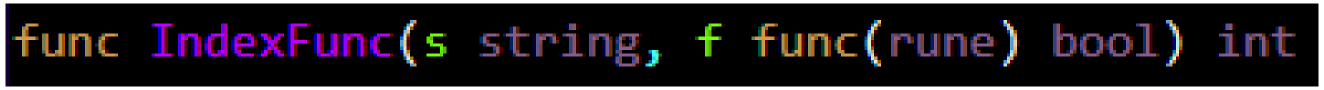


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	fmt.Println(strings.IndexRune("golang", 'g'))
	fmt.Println(strings.IndexRune("gophers", 'h'))
	fmt.Println(strings.IndexRune("góphers", 'h'))
	fmt.Println(strings.IndexRune("góphers", 'ó'))
	fmt.Println(strings.IndexRune("golang", 'x'))
}
```

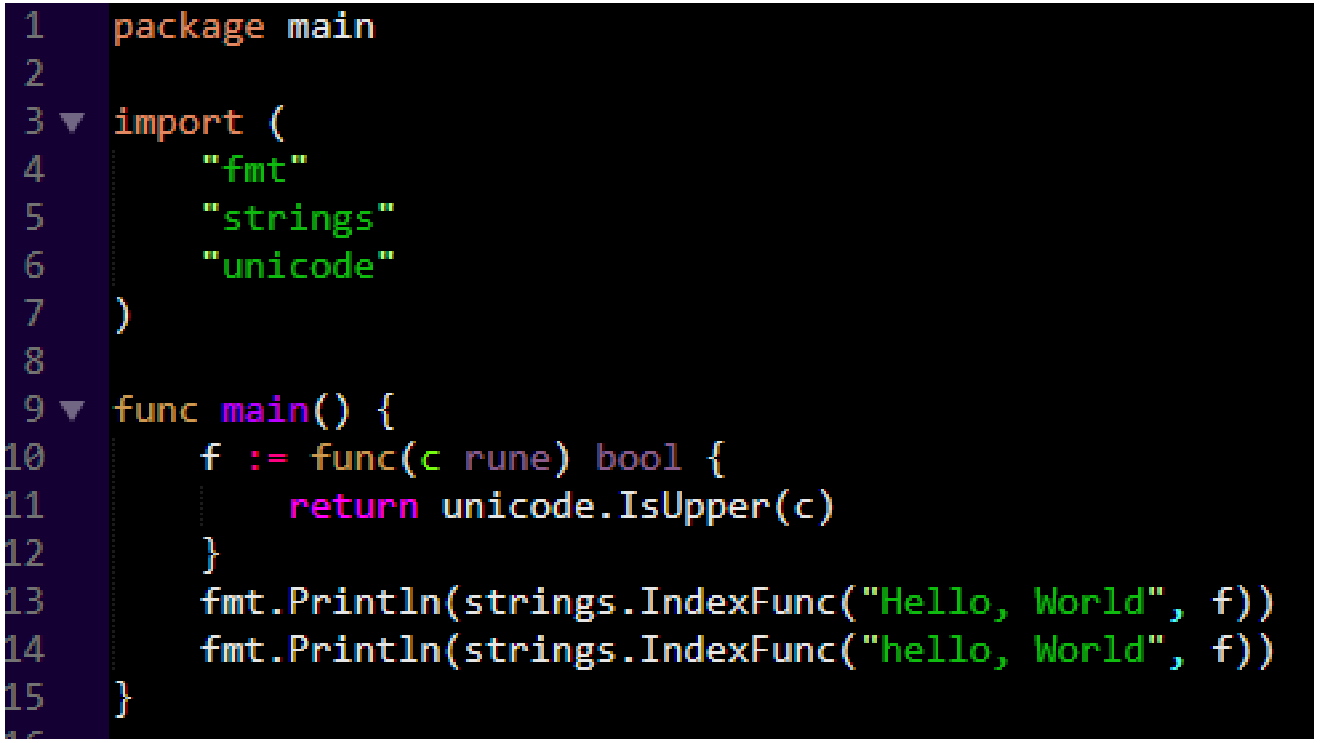


```
0
3
4
1
-1
```

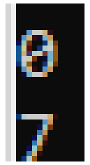
strings package



```
func IndexFunc(s string, f func(rune) bool) int
```

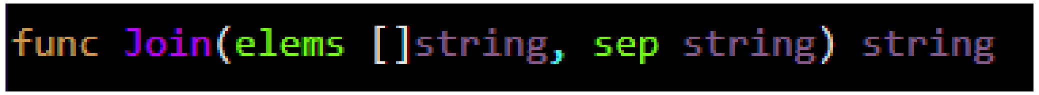


```go
package main

import (
	"fmt"
	"strings"
	"unicode"
)

func main() {
	f := func(c rune) bool {
		return unicode.IsUpper(c)
	}
	fmt.Println(strings.IndexFunc("Hello, World", f))
	fmt.Println(strings.IndexFunc("hello, World", f))
}
```

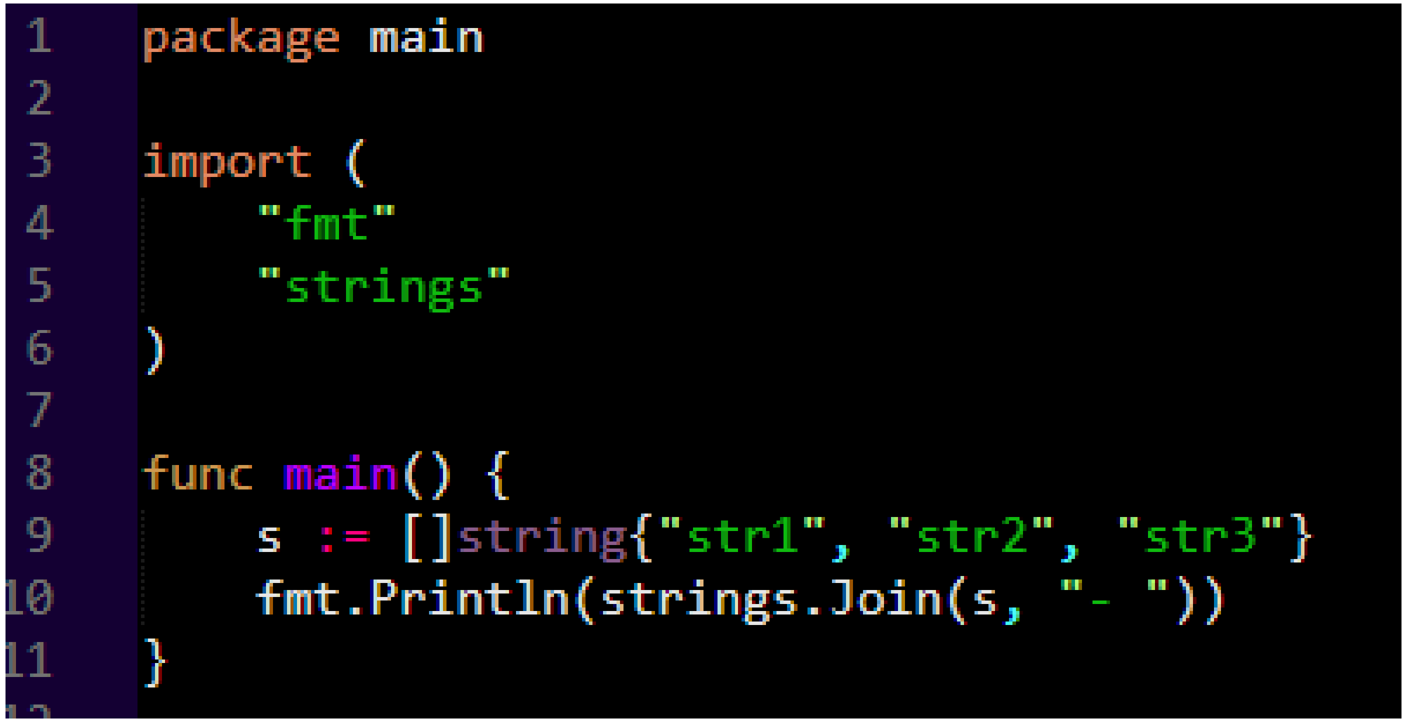


```
0
7
```

strings package

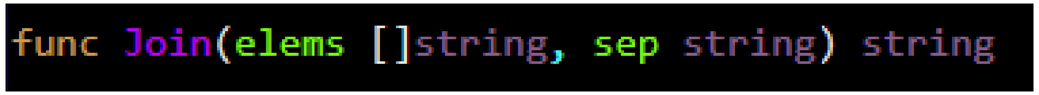


```go
func Join(elems []string, sep string) string
```

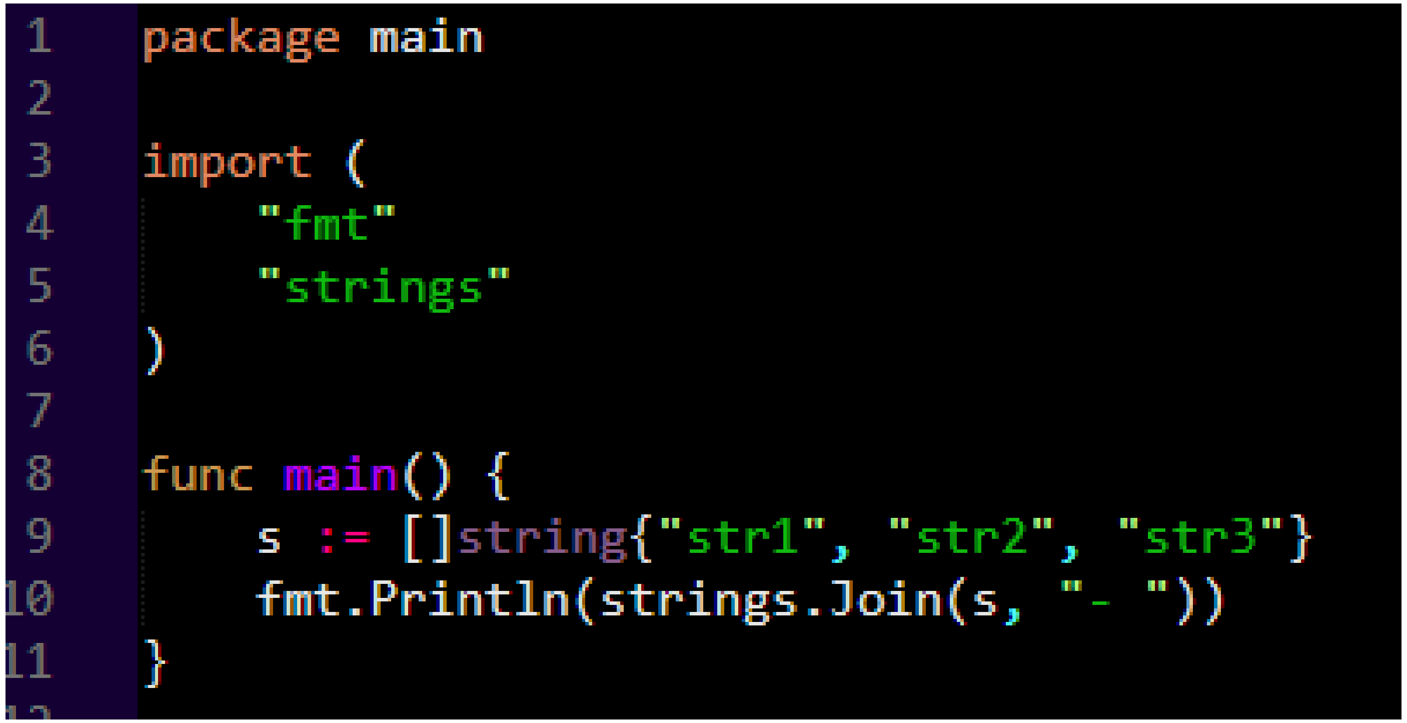


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	s := []string{"str1", "str2", "str3"}
	fmt.Println(strings.Join(s, "- "))
}
```

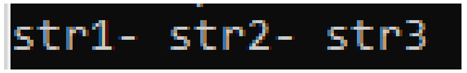
strings package



```go
func Join(elems []string, sep string) string
```

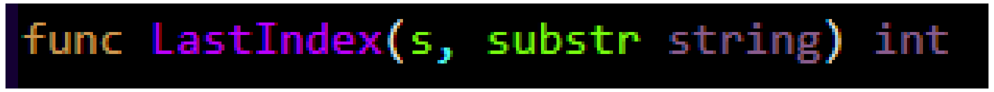


```go
package main

import (
	"fmt"
	"strings"
)

func main() {
	s := []string{"str1", "str2", "str3"}
	fmt.Println(strings.Join(s, "- "))
}
```

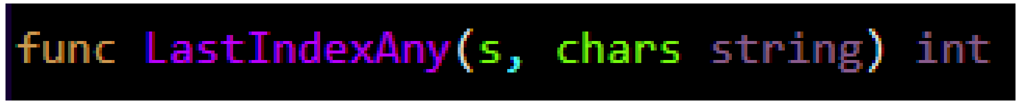


```
str1- str2- str3
```

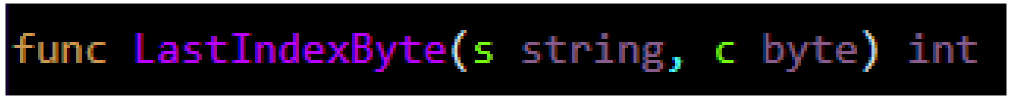
strings package



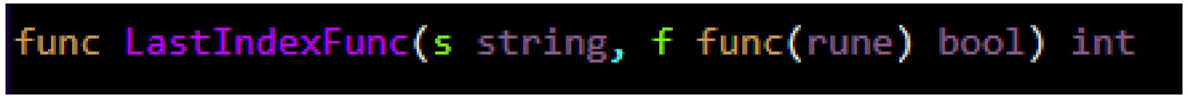
```go
func LastIndex(s, substr string) int
```



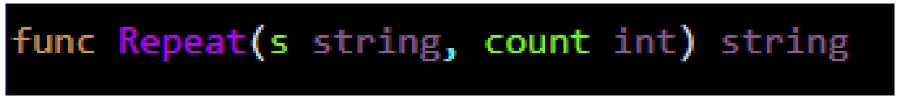
```go
func LastIndexAny(s, chars string) int
```



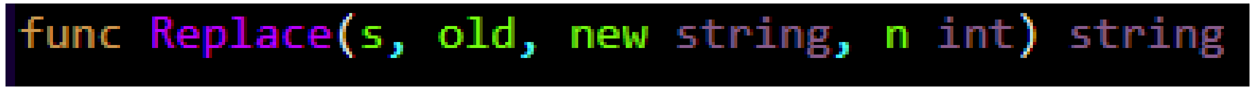
```go
func LastIndexByte(s string, c byte) int
```



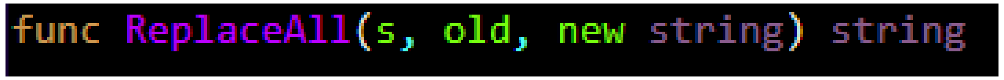
```go
func LastIndexFunc(s string, f func(rune) bool) int
```



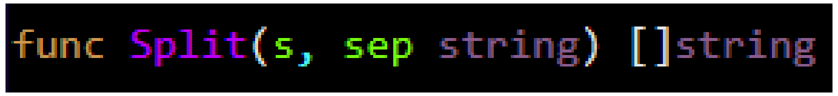
```go
func Repeat(s string, count int) string
```



```go
func Replace(s, old, new string, n int) string
```

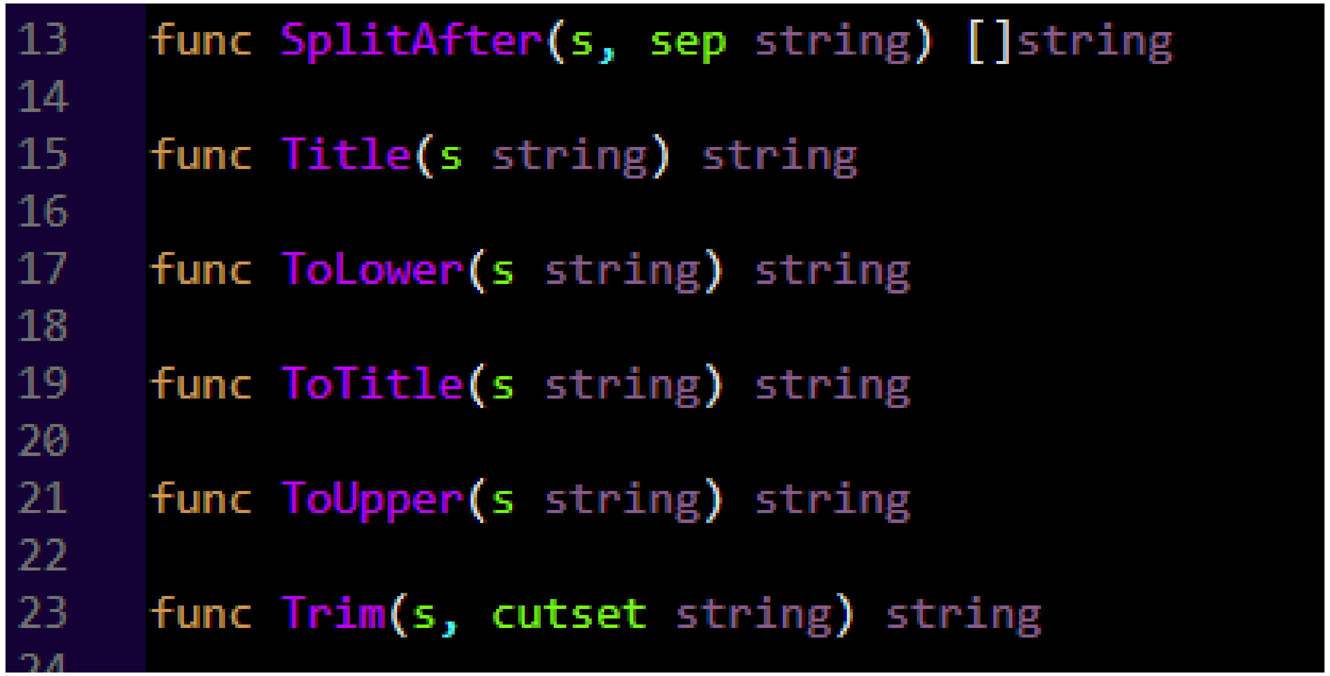


```go
func ReplaceAll(s, old, new string) string
```



```go
func Split(s, sep string) []string
```

strings package



```
func SplitAfter(s, sep string) []string

func Title(s string) string

func ToLower(s string) string

func ToTitle(s string) string

func ToUpper(s string) string

func Trim(s, cutset string) string
```

Thank You