



Laboratorio di
program-
mazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

Laboratorio di programmazione

Corsi di laurea triennale in Informatica, Informatica musicale e Informatica per la comunicazione digitale

Turno A (A-Can) - Turno B (Cao-D)

Docenti: Anna Morpurgo - Andrea Trentini
Tutor: Leonardo Albani - Federico Bruzzzone

Dipartimento di Informatica
Università degli Studi di Milano

A.A. 2023-2024



Laboratorio di
program-
mazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

Laboratorio 04

2/11/2023



I caratteri in Go

Laboratorio di
program-
mazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

- Non esiste un vero tipo per i caratteri in Go.
- Per gestire i **caratteri** Go mette a disposizione degli **alias** predefiniti **di int**:
 - **byte**, un alias di uint8, per i caratteri ASCII
 - **rune**, un alias di int32, per i caratteri Unicode (UTF-8)
- In quanto tipi numerici, sono ammesse le **operazioni aritmetiche** (utili ad es. per criptare)
- È importante ricordarsi che byte e rune sono alias di int, indistinguibili dai tipi int corrispondenti, soprattutto quando occorre **leggere o stampare** caratteri.



Lettura e stampa di caratteri

Laboratorio di
program-
mazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

Lettura

Con la funzione `fmt.Scan` non è possibile leggere caratteri.

Occorre usare `fmt.Scanf`

```
var ch rune  
fmt.Scanf("%c", &ch)
```

`%c` indica di prendere il prossimo carattere (runa in questo caso, ma lo stesso vale per i byte) in input (anche se è un carattere di spazio, tab, a capo) e di salvarlo nella variabile indicata (`ch`)



Lettura e stampa di caratteri

Laboratorio di
program-
mazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

Stampa

Si può usare Printf, oppure Print/Println utilizzando string():

```
var c rune = 'ò'  
fmt.Printf("%c è una runa\n", c)  
fmt.Println(string(c), "è una runa")
```

produce:

ò è una runa

ò è una runa

Lo stesso vale per i byte.



Sequenze di escape

Laboratorio di
program-
mazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

Codifica di caratteri particolari:

- tab (indentazione): `\t`
- newline (a capo): `\n`
- backslash(`\`): `\\`
- double quotes (`"`): `\"`

Esempio:

```
var c = '\t'
```



Il pacchetto unicode

Laboratorio di
program-
mazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

Funzioni per verificare se il carattere è:

- una lettera: `unicode.IsLetter(r rune)`
- una cifra: `unicode.IsDigit(r rune)`
- un carattere “bianco”: `unicode.IsSpace(r rune)`
- un carattere minuscolo: `unicode.IsLower(r rune)`
- un carattere maiuscolo: `unicode.IsUpper(r rune)`

Il valore restituito è di tipo `bool`.

Funzioni per cambiare il *case* di un carattere:

- `unicode.ToUpper(r rune)`
- `unicode.ToLower(r rune)`

Il valore restituito è di tipo `rune`.



Documentazione di Golang

Laboratorio di
program-
mazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

<https://golang.org/pkg/>



Documents

Packages

The Project

Help

Blog

Play

Search



Packages

Standard library
Other packages
Sub-repositories
Community

Standard library ▾

Name

archive
tar
zip
bufio
builtin
bytes
compress
bzip2
flate
gzip
lzw
zlib
container
heap
list
ring
context

Synopsis

Package tar implements access to tar archives.

Package zip provides support for reading and writing ZIP archives.

Package bufio implements buffered I/O. It wraps an io.Reader or io.Writer object, creating another object (Reader or Writer) that also implements the interface but provides buffering and some help for textual I/O.

Package builtin provides documentation for Go's predeclared identifiers.

Package bytes implements functions for the manipulation of byte slices.

Package bzip2 implements bzip2 decompression.

Package flate implements the DEFLATE compressed data format, described in RFC 1951.

Package gzip implements reading and writing of gzip format compressed files, as specified in RFC 1952.

Package lzw implements the Lempel-Ziv-Welch compressed data format, described in T. A. Welch, "A Technique for High-Performance Data Compression", Computer, 17(6) (June 1984), pp 8-19.

Package zlib implements reading and writing of zlib format compressed data, as specified in RFC 1950.

Package heap provides heap operations for any type that implements heap.Interface.

Package list implements a doubly linked list.

Package ring implements operations on circular lists.

Package context defines the Context type, which carries deadlines, cancellation signals, and other request-scoped values across API boundaries and between processes.





Le stringhe

Laboratorio di
program-
mazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

- Una stringa Go è una sequenza di caratteri UTF-8 (in codice ASCII a 1 byte quando possibile, in codice UTF-8 a 2-4 byte quando necessario)
- Una stringa Go è quindi una sequenza di caratteri (rune), ciascuno di **lunghezza variabile** (da 1 a 4 byte)
- Lo *zero value* di una stringa è la stringa vuota "" (**non** lo spazio " ")
- Per accedere al byte *i*-esimo della stringa `str`: `str[i]`



Le stringhe

Laboratorio di
program-
mazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

Operazioni con le stringhe

- lunghezza in byte: `len(str)`
- lunghezza in rune: `utf8.RuneCountInString(str)` del pacchetto unicode
- concatenazione: `str1 + str2`
- sottostringa : `str[i:j]` (i-esimo byte incluso, j-esimo escluso)

Attenzione però che una stringa non può essere modificata:

`str[i] = 'a'` non è ammesso.

È invece possibile avere un riassegnamento:

```
str = "ciao"
```

```
str = "hello"
```



Le stringhe

Laboratorio di
program-
mazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

Due modi di scorrere una stringa:

- Per byte: il **for ternario** itera sugli indici e restituisce elementi di tipo **byte** (`str[i]`)

```
for i:=0; i < len(str); i++ {  
    fmt.Print(string(str[i]))  
}
```
- Per rune: il **for-range** itera sui caratteri Unicode della stringa e restituisce elementi di tipo **rune** (`char`)

```
for pos, char := range str {  
    fmt.Print(pos, string(unicode.ToLower(char)))  
}
```

*Attenzione: char è qui una **copia** della runa in quella posizione; il valore nella stringa non può essere modificato facendo assegnamenti a char nel blocco del for.*



Costruzione della conoscenza strategica: composizione

Laboratorio di
program-
mazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

Una volta identificati i sottoproblemi in un problema e le relative soluzioni, spesso occorre che le soluzioni dei sottoproblemi vengano **composte** insieme nel programma.

Le soluzioni dei sottoproblemi possono essere composte in tre modi diversi:

- **concatenate** una dopo l'altra, eventualmente nell'ordine necessario per risolvere il problema;
- **fuse** in modo che le parti comuni siano svolte insieme (es. $\max + \min$; $\text{somma} + \text{conteggio}$, ...);
- **annidate** una nell'altra, ad esempio quando l'elaborazione da fare a ogni iterazione richiede a sua volta un ciclo.



Composizione: concatenazione

Laboratorio di
programmazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

Specifiche: dato un numero intero in forma di stringa, stampare le cifre che sono maggiori della media delle cifre del numero.

```
func main() {  
    var numero string  
    fmt.Scan(&numero)  
    dim := len(numero)  
    var sum byte  
    for i := 0; i < dim; i++ {  
        n := numero[i] - '0'  
        sum += n  
    }  
    media := sum/byte(dim) + '0'  
    for i := 0; i < dim; i++ {  
        if numero[i] < media {  
            fmt.Println(string(numero[i]))  
        }  
    }  
}
```



Composizione: fusione

Laboratorio di
programmazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

Specifiche: dato un numero intero in forma di stringa, stampare la somma delle cifre e la cifra più grande.

```
func main() {  
    var numero string  
    fmt.Scan(&numero)  
    dim := len(numero)  
  
    var sum, max byte  
    for i := 0; i < dim; i++ {  
        n := numero[i] - '0'  
        sum += n  
        if n > max {  
            max = n  
        }  
    }  
    fmt.Println("somma:", sum)  
    fmt.Println("max:", max)  
}
```



Composizione: annidamento

Laboratorio di
programmazione - Ed. 1
- 2023/24

A. Morpurgo

Laboratorio 04

Argomenti

Caratteri

Stringhe

Composizione

Specifiche: data una serie di 10 numeri interi, stampare il primo numero la cui somma delle cifre è pari.

```
func main() {  
    const DIM = var numero string  
    for i := 0; i < DIM; i++ {  
        fmt.Scan(&numero)  
        dim := len(numero)  
        var sum byte  
        for i := 0; i < dim; i++ {  
            n := numero[i] - '0'  
            sum += n  
        }  
        if sum%2 == 0 {  
            fmt.Println(numero)  
            break  
        }  
    }  
}
```