I/O AVANZATO

APERTURA DI FILE (pacchetto os)

LEGGERE `os.Open(filename string)` (*File, error)

nil ↓

CREARE `os.Create(filename string)` (*File, error)

nil ↓

LETTURA

```
f, err := os.Open("pippo.txt")
if err != nil {
    [ CASO VARIE ]
}
defer f.Close()
```

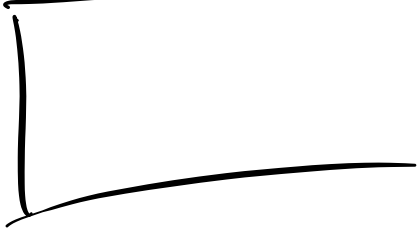
```
func (f *File) Chdir() error
func (f *File) Chmod(mode FileMode) error
func (f *File) Chown(uid, gid int) error
func (f *File) Close() error
func (f *File) Fd() uintptr
func (f *File) Name() string
func (f *File) Read(b []byte) (n int, err error)
func (f *File) ReadAt(b []byte, off int64) (n int, err error)
func (f *File) ReadDir(n int) ([]DirEntry, error)
func (f *File) ReadFrom(r io.Reader) (n int64, err error)
func (f *File) Readdir(n int) ([]FileInfo, error)
func (f *File) Readdirnames(n int) (names []string, err error)
func (f *File) Seek(offset int64, whence int) (ret int64, err error)
func (f *File) SetDeadline(t time.Time) error
func (f *File) SetReadDeadline(t time.Time) error
func (f *File) SetWriteDeadline(t time.Time) error
func (f *File) Stat() (FileInfo, error)
func (f *File) Sync() error
func (f *File) SyscallConn() (syscall.RawConn, error)
func (f *File) Truncate(size int64) error
func (f *File) Write(b []byte) (n int, err error)
func (f *File) WriteAt(b []byte, off int64) (n int, err error)
func (f *File) WriteString(s string) (n int, err error)
type FileInfo
```

```
var buffer []byte
buffer = make([]byte, 1)

for {
    n, err := f.Read(buffer)
    if n == 0 {
        break
```

```
    }
    if err != nil {
        fmt.Println("AUTOOO")
        break
    }
    fmt.Printf("%c", buffer[0])
}
```

SCRITTURA

```
f, err := os.Create("pippo.txt")
if err != nil {
    
}
defer f.Close()
var b byte[]
b = []byte("Pado Boldi")
n, err := f.Write(b)
if err != nil {
    
}
```

ESERCIZIO

- Scrivete (e usate) una funzione che dato il nome di un file (contenente testo ASCII) restituisca una `map[rune]float64` in cui le chiavi sono i caratteri che compaiono nel file e i valori sono le loro frequenze relative.
- Scrivere una funzione che data due mappe come sopra dice

se sono più o meno
uguali

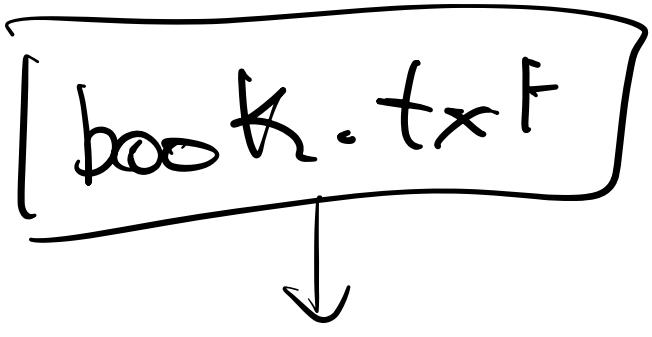
- Scrivete un programma
che prende sulla riga
di comando due nomi
di due file e tenta
di capire se sono
scritti nella stessa lingua

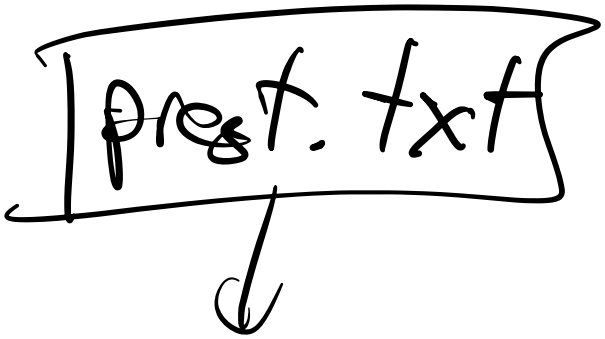
type Book {
author, title string
anno int
id int
:

}

type Prestito {
id_book int
data_prestito data
persona string

}





CSV

TSV

DSV

JSON

XML

LETTURA FILE CSV

```
f, err := os.Open("book.csv")
defer f.Close()
csvReader := csv.NewReader(f)
for {
    fields, err := csvReader.
                   Read()
    if fields == nil {
```

break

}

usi fields

}

fwb.

FPrintln

FPrint

FPrintf

FScan

FScanf

bufio.

```
f, _ := os.Open("pippo.txt")
scanner := bufio.NewScanner(f)
for scanner.Scan() {
        riga := scanner.Text()
        ...
}
```