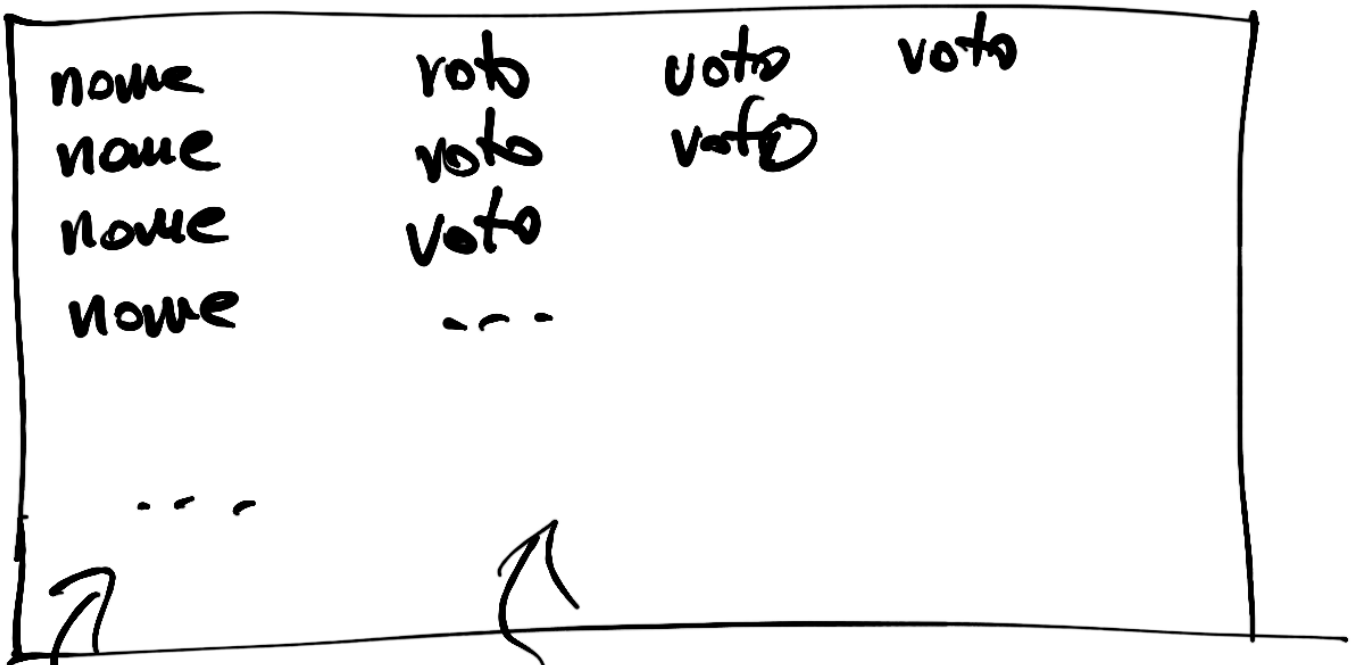
ESERCIZIO

esami.txt



nomi distinti

Voti in 30-esimi
(32 = 30 e lode)

csv

tsv

Parsing

Espressioni regolari

ESPRESSIONI REGOLARI

a b

= {ab}

zacca

= {zacca}

P = a . b

= {aab, abb, acb, a☺b,
a1b, a7b, ... }

a.. b ...

= {aaabccca, abbbzca, ...}

T = cannabis alba

P.FindString(T) ⟶ "a|b"

1 o più volte

$a+b$

$= \{ab, aab, aaab, aaaab, \ldots\}$

0 più volte

$a*b$

$= \{b, ab, aab, aaab, \ldots\}$

$(ab)*$

$= \{\epsilon, ab, abab, ababab, \ldots\}$

$a|b$

$= \{a, b\}$

$(a|b)+$

$= \{a, b, ab, aa, ba, bb, \ldots\}$

(0|1|2|3|4|5|6|7|8|9)*(0|2|4|6|8)

[0123456789]*[02468]

[0-9]*[02468]

```
func parseLine(line string)
(name string, sores []int, err error){
name RE := regexp.MustCompile(
    "([a-z]|[A-Z]|[0-9]|_)+")
sore RE := regexp.MustCompile(
    "[0-9]+")

name = nameRE.FindString(line)

name Borders := nameRE.
        FindStringIndex(line)

scoreString := scoreRE.FindAllString
        (line[nameBorders[1]:], -1)

for _, x := range scoreStrings {
        n, err1 := strconv.Atoi(x)
        if err1 != nil {
                err = err1
                return
        }
        scores = append(scores, n)
}
```

```
type examMap map[string][]int

func readFile (filename string)
  (em examMap, err error) {
  f, e := os.Open(filename)
  if e != nil {
      err = e
      return
  }
  defer f.Close()
  scanner := bufio.NewScanner(f)
  for scanner.Scan() {
      line := scanner.Text()
      name, scores, e := parseLine(line)
      if e != nil {
          err = e
          return
      }
      em[name] = scores
  }
```

```
func avg(scores []int) float64 {
    var c float64
    for _, v := range scores {
        c += math.Min(30.0,
            float64(v))
    }
    return c/float64(len(scores))
}

func to110(x float64) float64 {
    return 110.0 * x/30.0
}
```

```
func produceReport(em examMap,
    filename string) {
  f, _ := os.Create(filename)
  defer f.Close()
  for name, scores := range em {
    av := avg(scores)
    av110 := to110(av)
    fmt.Fprintf(f,
      "%30s\t%.2f\t%.2f\n",
      name,
      av,
      av110
    )
  }
}
```

```
func main() {
    scoreFilename := os.Args[1]
    reportFilename := os.Args[2]
    em, err := readFile(scoreFilename)
    if err != nil {
        fmt.Println(err)
        os.Exit(1)
    }
    produceReport(em, reportFilename)
}
```