Laboratorio di Elementi di Bioinformatica

Laurea Triennale in Informatica
(codice: E3101Q116)

AA 2016/2017

Linguaggio Ruby:
input/output

Docente del laboratorio: Raffaella Rizzi

Funzioni di standard input/ output

- **`gets`**: legge una riga dallo standard input (il terminatore è il carattere di *newline*)
- **`puts`**: valuta e converte in stringhe gli argomenti e le stampa in standard output, producendo un *newline* dopo ogni argomento
- **`print`**: funziona come `puts` senza però stampare il *newline* dopo ogni stringa

Funzioni di standard input/ output

- **`gets`**: legge una riga dallo standard input (il terminatore è il carattere di *newline*)
- **`puts`**: valuta e converte in stringhe gli argomenti e le stampa in standard output, producendo un *newline* dopo ogni argomento
- **`print`**: funziona come `puts` senza però stampare il *newline* dopo ogni stringa

NB. Per redirigere un file *infile* in standard input basta specificarlo nella riga di comando dopo il nome dello script Ruby

>`ruby` *script.rb infile*

Funzioni di standard input/ output

- **`gets`**: legge una riga dallo standard input (il terminatore è il carattere di *newline*)
- **`puts`**: valuta e converte in stringhe gli argomenti e le stampa in standard output, producendo un *newline* dopo ogni argomento
- **`print`**: funziona come `puts` senza però stampare il *newline* dopo ogni stringa

NB. Per redirigere un file *outfile* in standard output basta specificarlo nella riga di comando dopo il simbolo >

>`ruby` *script.rb infile* > *outfile*

Input/Output da file (classe `File`)

Classe: `File`

Costruttore:

```
file_obj = File.new(filename, flag)
```

dove:

`file_obj` ➔ riferimento all'oggetto `File`

`filename` ➔ nome del file

`flag` ➔ flag di creazione dell'oggetto `File`:

- "r": solo lettura (il file deve esistere)
- "w": solo scrittura (se il file esiste già, ne viene creato uno nuovo)
- "r+": lettura e scrittura (il file deve esistere)
- "w+": lettura e scrittura (se il file esiste già, ne viene creato uno nuovo)
- "a": solo scrittura in *append* (se il file non esiste viene creato)
- "a+": lettura scrittura in *append* (se il file non esiste viene creato)

Metodi (di istanza) della classe `File`

- **`gets`**: legge una riga dal file (usando come terminatore di riga il *newline*)
- **`puts`**: valuta e converte in stringhe gli argomenti e le stampa su file, producendo un *newline* dopo ogni argomento
- **`print`**: funziona come `puts` senza però stampare il *newline* dopo ogni argomento.
- **`each_line`**: iteratore che divide il file in righe (separandole tramite il *newline*) e passa ogni riga come parametro a un blocco
- **`each_line(`*`separator`*`)`**: iteratore che divide il file in parti (usando *`separator`* come separatore) e passa ogni parte come parametro a un blocco
- **`close`**: chiude il file

Metodi (di istanza) della classe `File`

- **`gets`**: legge una riga dal file (usando come terminatore di riga il *newline*)
- **`puts`**: valuta e converte in stringhe gli argomenti e le stampa su file, producendo un *newline* dopo ogni argomento
- **`print`**: funziona come `puts` senza però stampare il *newline* dopo ogni argomento.
- **`each_line`**: iteratore c
  tramite il *newline*) e pas
  blocco
- **`each_line(`*`separator`*`)`**: iteratore che divide il file in parti (usando *`separator`* come separatore) e passa ogni parte come parametro a un blocco
- **`close`**: chiude il file

NB. Tutto quello che viene letto da file è una stringa.

Metodo (di classe) della classe `File`

In alternativa a usare il costruttore per istanziare un ogetto `File` si può usare il metodo di classe `open` che prende come parametri il nome del file e il flag di apertura, e invoca un blocco passandogli come parametro il riferimento all'oggetto creato

```
File.open(filename, flag) do |file_obj|
            …
    end
```

NB: il file deve essere gestito all'interno del blocco in quanto viene automaticamente chiuso alla sua uscita

Metodo (di classe) della classe `File`

Il metodo `readlines` della classe `File` permette di specificare come parametro il nome di un file e restituisce un array contenente tutte le sue righe

```
rows = File.readlines("inputfile.txt")

rows.each {|row| puts row}
```