APPELLO SCRITTO DI PROGRAMMAZIONE 1
CORSO DI LAUREA IN MATEMATICA
UNIVERSITÀ DEGLI STUDI DI MILANO

2013–2014

8.IX.2014

VINCENZO MARRA

Indice

Avvertenza. Il tema d'esame richiede lo sviluppo di un singolo programma che implementi una semplice versione del Gioco dell'Impiccato. (Il primo giocatore sceglie una parola. Il secondo giocatore tenta di indovinarne le lettere avendo a disposizione un numero limitato di tentativi. Sia il primo che il secondo giocatore saranno qui impersonati da un singolo utente.) La maggior parte del codice necessario sarà scritto nella funzione `main`, nel terzo e ultimo esercizio. Svilupperete alcune semplici funzioni ausiliarie nei primi due esercizi. Integrerete poi le funzioni ausiliarie nel programma finale, che consisterà di un unico file. Si raccomanda di leggere interamente il tema d'esame prima di cominciare a scrivere le soluzioni dei singoli esercizi, in modo da avere chiaro l'obiettivo finale.

Esercizio 1

Leggere una stringa e calcolarne la lunghezza.

Ultima revisione: 8 settembre 2014.

Punti: 4.

Scrivete una funzione di prototipo `int leggi(char *)` che legga una stringa di caratteri inserita dal terminale, e la memorizzi nel parametro passato in argomento. Usate la funzione `fgets()` del file di intestazione `stdio.h`. La funzione `leggi()` deve assicurarsi di sostituire, nella stringa letta, il carattere `'\n'` (il ritorno a capo digitato dall'utente) col carattere `'\0'`. La funzione restituisce il valore 0 se si è riscontrato un problema di I/O in lettura, e un valore diverso da zero se la lettura è andata a buon fine.

Scrivete poi una funzione di nome `lung()` e prototipo appropriato, che accetti una stringa in ingresso e ne restituisca la lunghezza (intesa come il numero di caratteri che compongono la stringa, escluso il carattere di terminazione `'\0'`). Se la stringa passata in argomento è `NULL`, la funzione restituisce 0.

Esercizio 2

Controllare il formato di una parola.

Punti: 8.

Nel Gioco dell'Impiccato che svilupperete nel prossimo esercizio, la parola da indovinare non è una stringa arbitraria, ma deve soddisfare le due condizioni seguenti.

(1) La parola contiene solo caratteri alfabetici.
(2) La parola contiene al più tante lettere maiuscole quante lettere minuscole.

Scrivete una funzione

```
int parolaok(char *)
```

che accetti in ingresso una stringa, e restituisca un valore diverso da zero se la stringa soddisfa le due condizioni precedenti, e zero altrimenti. Se l'argomento è `NULL` la funzione restituisce 0.

(*Suggerimento.* Per controllare la classe di un carattere, utilizzate le opportune funzioni definite nel file di intestazione `ctype.h` della libreria standard del C.)

Esercizio 3

Il Gioco dell'Impiccato.

Punti: 18 punti.

Scrivete una funzione `main` che implementi una versione semplificata del Gioco dell'Impiccato, come descritto qui di seguito. Usate le funzioni sviluppate negli esercizi precedenti nel modo opportuno.

All'avvio, il programma chiede all'utente di inserire la parola da indovinare. Il programma legge quindi una stringa dal terminale. (Assumete che la lunghezza massima ammissibile della stringa sia pari alla costante `BUFSIZ` definita in `stdio.h`, oppure usate un'altra costante che sia un limite superiore ragionevole, per esempio 256.) Se si verifica un errore di I/O il programma termina con un messaggio d'errore. Altrimenti, il programma controlla che la stringa immessa soddisfi i requisiti (1) e (2) prescritti dall'Esercizio 2. In caso negativo, il programma informa l'utente e chiede il reinserimento della stringa, fino a quando essa non soddisfi i requisiti. Si noti che la stringa acquisita deve essere terminata da `'\0'`, e non da `'\n'`.

Una volta acquisita la parola da indovinare, il gioco ha inizio. Il programma visualizza una stringa della medesima lunghezza della parola da indovinare, ma composta interamente di asterischi (`*`). Il programma chiede quindi all'utente la posizione del carattere che intende tentare di indovinare, e acquisisce il dato con[1] `scanf()`. Assumete che l'utente digiti esattamente un intero seguito dall'invio `'\n'`. (*Suggerimento.* Per eliminare il `'\n'` dallo stream d'ingresso standard, invocate `getchar()` dopo aver invocato `scanf()`.) Se la posizione è inesistente — ossia è un intero non compreso fra 1 e la lunghezza delle stringa — il programma chiede il reinserimento. Se la posizione esiste, ma il carattere è già stata indovinato in precedenza dall'utente, il programma avvisa l'utente e chiede il reinserimento. Altrimenti, se la posizione esiste e il corrispondente carattere non è ancora stato indovinato, il programma chiede all'utente di indovinare il carattere. Si assuma che l'utente digiti qui un singolo carattere seguito da invio: usate quindi `getchar()` per acquisire il carattere, seguito da una seconda invocazione per ripulire lo stream[2] da `'\n'`. Acquisito il carattere, il programma lo confronta col carattere nella posizione della stringa da indovinare già indicata dall'utente. Se i due caratteri coincidono — ossia, se l'utente ha indovinato il carattere — il programma avvisa l'utente e torna a mostrare la parola come sequenza di asterischi, mostrando però in chiaro i caratteri già indovinati. Se i due caratteri invece non coincidono, il programma avvisa l'utente di non aver indovinato, e il gioco continua col programma che chiede all'utente di indicare una nuova posizione all'interno della stringa. L'utente vince quando indovina tutte le lettere della parola. Appena ciò accade, il programma mostra la stringa in chiaro, informa l'utente della vittoria e termina. L'utente perde quando fa (strettamente) più di 3 tentativi errati relativi alla stessa lettera della parola, e ciò anche se i 3 tentativi sono fatti in modo non consecutivo. Il programma deve quindi tenere traccia di quanti tentativi l'utente ha già fatto per ciascuna lettera. (*Suggerimento.* A tal fine si usi un array di interi della stessa lunghezza della parola da indovinare.)

Si completi infine il programma dando l'utente la possibilità di sostituire il carattere `*` usato per mascherare la parola da indovinare con un altro carattere qualsiasi. Più in dettaglio, l'utente ha la possibilità di passare in argomento dalla riga di comando al programma il carattere da usare in luogo dell'asterisco. Se vi è più di un argomento sulla riga di comando, i rimanenti sono ignorati; del primo argomento è estratto solo il primo carattere, che prenderà il posto di `*` durante il gioco. Non vi sono restrizioni relative alla classe del carattere che l'utente può specificare. Se non vi sono argomenti sulla riga di comando, il programma utilizza `*`.

Si vedano le Figg. 1–3 per qualche esempio di esecuzione.

---

[1] Un'implementazione più completa della lettura dell'intero si può fare tramite un'invocazione di `leggi()`, o direttamente di `fgets()`, seguita da un'opportuna analisi della stringa acquisita. Una tale implementazione più raffinata non è richiesta.

[2] Un commento analogo al precedente si applica qui: non è richiesta un'implementazione più sofisticata dell'acquisizione del carattere.

```
Vincenzos-MacBook-Air:Soluzione enzo$ ./a.out
(Uso il carattere di default * per la maschera.)

                ***************************
                ** Gioco dell'Impiccato **
                ***************************

Inserisci la parola da indovinare: Po

**

Posizione? 1
Lettera? P
Hai indovinato!

P*

Posizione? 2
Lettera? o
Hai indovinato!
Hai vinto! La parola e': Po.
Vincenzos-MacBook-Air:Soluzione enzo$
```

FIGURA 1. Primo esempio di esecuzione dell'intero programma.

Istruzioni per la consegna

- Funzioni da consegnare (in un unico file sorgente).

  `main()`
  `leggi()`
  `lung()`
  `parolaok()`
  Tutte le eventuali altre funzioni ausiliarie che abbiate scritto.

- Sito per la consegna:

  `https://upload.mat.unimi.it`

- Autenticarsi con le proprie credenziali di posta elettronica d'ateneo.
- Consegnare solo il codice sorgente (file `*.c`), non l'eseguibile.

(V. Marra) DIPARTIMENTO DI MATEMATICA *Federigo Enriques*, UNIVERSITÀ DEGLI STUDI DI MILANO, VIA CESARE SALDINI, 50, I-20133 MILANO
E-mail address: vincenzo.marra@unimi.it

```
(Uso il carattere di default * per la maschera.)

                ************************
                ** Gioco dell'Impiccato **
                ************************

Inserisci la parola da indovinare: Po

**

Posizione? 1
Lettera? Q
Non hai indovinato.

**

Posizione? 2
Lettera? o
Hai indovinato!

*o

Posizione? 1
Lettera? A
Non hai indovinato.

*o

Posizione? 1
Lettera? A
Non hai indovinato.
Hai esaurito il numero di tentativi disponibili. Hai perso!
```

FIGURA 2. Secondo esempio di esecuzione dell'intero programma.

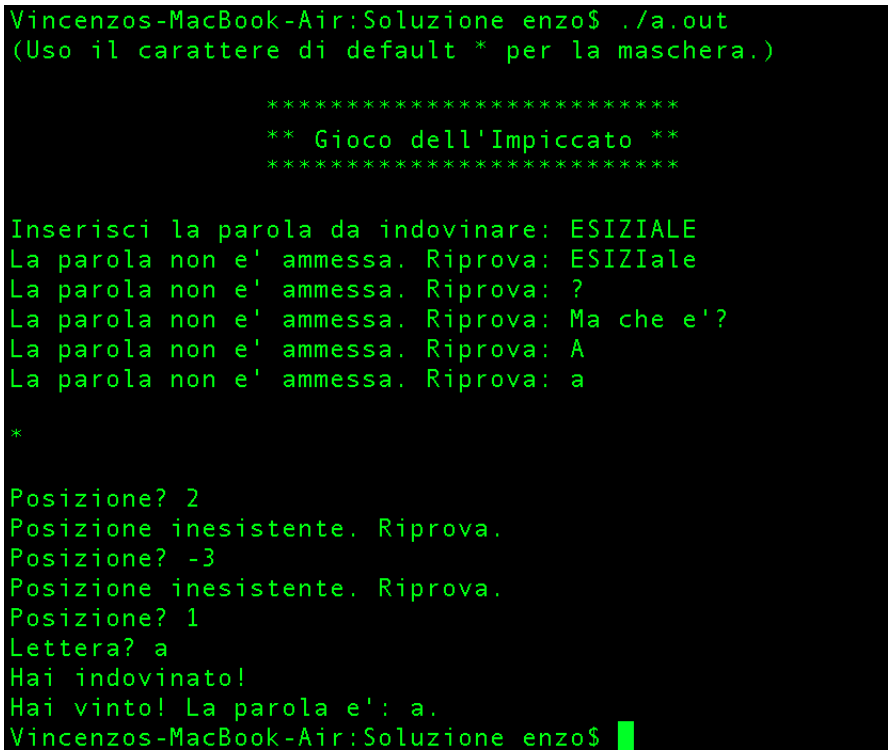

FIGURA 3. Terzo esempio di esecuzione dell'intero programma.