LA RUOTA DELLA FORTUNA

Un progetto di:

Davide Valeriani

Matricola 190883

Corso di laurea in Ingegneria Informatica

Esame di Fondamenti di Informatica A

Proff. Massimo Bertozzi e Eduardo Calabrese

Anno accademico 2007/2008

File costituenti il progetto:

- ruotafortuna.cpp (file sorgente)
- ruotafortuna.exe (file eseguibile)
- parole.txt (elenco frasi)

Descrizione generale

Il programma implementa il popolare gioco della ruota della fortuna, inventato negli Stati Uniti e arrivato in Italia nel 1989 in un quiz televisivo condotto da Mike Bongiorno. Lo scopo del gioco è indovinare la frase misteriosa prima degli altri giocatori. Durante il proprio turno, ogni giocatore può girare la ruota e scegliere una consonante che pensa possa trovarsi nella frase, oppure comprare una vocale, dare la risposta o fare altre azioni.
L'esecuzione del programma può avvenire in due modi: con o senza parametri. Lanciando l'eseguibile senza parametri, il programma cercherà nella cartella corrente il file "parole.txt" e, nel caso non lo trovasse, verrà terminata l'esecuzione con un messaggio d'errore. Il file "parole.txt", allegato al programma, contiene alcune frasi predefinite che i concorrenti dovranno indovinare.
È stata prevista anche la possibilità di passare al programma un file con un elenco di frasi "personalizzate", cioè scelte dall'utente. Per fare questo, è necessario passare al programma il nome del file di testo (compresa l'estensione) che contiene le frasi, che deve essere memorizzato nella stessa cartella dell'eseguibile. In questo modo, il programma caricherà le frasi da quel file, invece che da quello predefinito.
Ulteriori parametri passati al programma verranno ignorati.
Appena avviato il programma, vengono richiesti i nomi dei partecipanti alla sessione di gioco. Il numero dei partecipanti deve essere, ovviamente, almeno uno, ma non ha limiti fissati, se non per quanto riguarda la memoria disponibile sul sistema.
Una volta inseriti i nomi dei partecipanti, che verranno memorizzati tramite una struttura a lista circolare, il gioco inizia.
Il primo nome inserito è il nome del giocatore che inizierà il gioco. Il programma sceglierà una frase casuale dall'elenco (già caricato in memoria dinamica), la trasformerà in una serie di underscore ("_") e la visualizzerà. Poi, comparirà un menu, attraverso il quale il giocatore può scegliere cosa vuole fare. Il menu si compone di 9 voci:

1) *Gira la ruota*: permette di girare la ruota e di inserire una consonante che si pensa possa essere nella frase
2) *Compra una vocale*: permette di scegliere una vocale, pagando 1000 punti
3) *Dai la risposta*: permette di dare la risposta, ovvero di inserire la frase da indovinare
4) *Guarda la soluzione*: permette di visualizzare la soluzione; il punteggio del giocatore che sceglie questa opzione verrà dimezzato
5) *Genera una nuova frase*: permette di abbandonare la frase attuale e generarne una nuova; il punteggio del giocatore che sceglie questa opzione verrà dimezzato
6) *Guarda il tuo punteggio*: permette di visualizzare il punteggio del giocatore di turno
7) *Guarda la classifica*: permette di guardare l'elenco dei giocatori, ordinati secondo il proprio turno (prima il giocatore di turno, poi quello successivo e così via)
8) *Leggi le istruzioni*: permette di visualizzare il regolamento del gioco
9) *Esci dal programma*: termina l'esecuzione del programma.

Nel caso si decida di uscire dal programma, viene invocata una funzione che permette di deallocare la memoria che era stata precedentemente allocata per contenere l'elenco dei nomi e quello delle frasi.

Aspetti di carattere tecnico

A parte i tipi di dato semplici, il programma si basa su una lista circolare semplice contenente i nomi dei giocatori con il relativo punteggio, e su un array con memoria allocata dinamicamente per contenere le frasi caricate dal file.
La lista circolare semplice è stata implementata con una struttura (struct) composta da un campo nome, un campo punteggio e un campo puntatore. È stata scelta una lista circolare in quanto permette di gestire automaticamente i turni di gioco (arrivata alla fine della lista, si riparte dal primo giocatore).
L'array dinamico, invece, è stato scelto in quanto non si conosce a priori la lunghezza delle frasi ed era necessario un array per implementare correttamente la scelta casuale di una frase, ottenuta generando un indice casuale e accedendo direttamente alla frase presente nella posizione corrispondente dell'array. Per scelta progettuale, è stato fissato un limite massimo di 80 frasi da caricare in memoria. Le ulteriori frasi eventualmente presenti nel file di testo verranno ignorate.

Funzioni utilizzate

Per questioni di leggibilità e ottimizzazione del codice, il programma è stato suddiviso in diverse funzioni, descritte qui di seguito.

- ✓ **bool carica(char*[], int&, char[]="parole.txt");**
  Invocata appena inizia l'esecuzione del programma, apre il file contenente le frasi e carica fino ad un massimo di 80 frasi in un array dinamico. Il nome del file contenente le frasi può essere specificato al momento dell'esecuzione del programma, passandolo come parametro. In assenza di parametri, il nome del file predefinito è "parole.txt". La funzione restituisce un valore booleano, che indica se è stato possibile o meno aprire il file.
- ✓ **void scegli(char*[], char[], int);**
  Sceglie casualmente una frase dall'array delle frasi, utilizzando la funzione predefinita rand che genera un numero casuale e calcolando il resto della divisione tra questo numero e il numero di frasi caricate, così da generare un indice compreso tra 0 e il numero delle frasi – 1. Una volta scelta, la frase viene resa maiuscola per una più semplice gestione dei confronti tra la frase da indovinare e la risposta del giocatore, e memorizzata in un'apposita variabile. La funzione non restituisce nessun valore.
- ✓ **void mistero(char[], char[]);**
  Genera, dalla frase scelta, lo "scheletro" della frase da indovinare, sostituendo ogni lettera da un underscore ("_"). Se un carattere non è una lettera (spazi, punteggiatura…), viene lasciato invariato. La funzione non restituisce nessun valore.
- ✓ **int risposta(void);**
  Stampa a video il menu di scelta, acquisisce la scelta dell'utente e la restituisce come valore della funzione. Nel caso in cui la risposta non sia corretta, viene chiesto di reinserirne una corretta (compresa tra 1 e 9).

- ✓ **int lettera(char[], char[], bool&);**

Acquisisce una consonante dall'utente (controllando che questo non inserisca una vocale o un carattere non previsto), la inserisce nello "scheletro" della frase da indovinare, e restituisce un punteggio calcolato moltiplicando un valore casuale tra 100 e 1000 (generato dalla funzione *giralaruota*) per il numero di occorrenze di quella lettera nella frase. Nel caso in cui la lettera non ci sia o sia già stata inserita, il giocatore passa il turno, settando la variabile booleana *cambio* a *true*.

- ✓ **void vocale(char[], char[], int&, bool&);**

Simile alla precedente, ma questa volta acquisisce una vocale. Dopo aver controllato che l'utente disponga dei punti necessari ad acquistare una vocale (settati dalla costante *costvoc*), gli sottrae i punti, acquisisce una vocale (con i controlli di routine), la sostituisce nello "scheletro" della frase e, nel caso non sia presente, setta la variabile *cambio* a *true* per far passare il turno al giocatore. La funzione non restituisce nessun valore.

- ✓ **bool indovinato(char[], char[]);**

Controlla che ci sia rimasta almeno una consonante o una vocale da indovinare, altrimenti segnala che la frase è stata indovinata. Il controllo è stato implementato semplicemente confrontando lo "scheletro" della frase con la frase scelta precedentemente. Se almeno una lettera è uguale a "_" restituisce *false*, altrimenti restituisce *true*.

- ✓ **int rispondi(char[], char[], bool&);**

Acquisisce una stringa dall'utente e la confronta con la parola da indovinare. Come primo controllo, confronta la lunghezza delle due stringhe e, nel caso in cui siano diverse, significa che l'utente non ha indovinato la frase; altrimenti, procede confrontando carattere per carattere. Nel caso in cui la risposta sia sbagliata, la variabile *cambio* viene settata a *true* e la funzione restituisce un numero negativo che verrà sottratto al punteggio del giocatore come penalità. Nel caso in cui la risposta sia corretta, viene generato un valore casuale tra 100 e 1000 e viene moltiplicato per il numero di lettere rimaste da indovinare; il risultato viene restituito dalla funzione e verrà aggiunto al punteggio del giocatore.

- ✓ **void soluzione (char[], bool&, int&);**

Dopo aver richiesto una conferma all'utente, visualizza la frase "in chiaro" e dimezza il punteggio del giocatore di turno. La funzione non restituisce nessun valore.

- ✓ **void nuovaparola (char&, bool&, bool&, int&);**

Dopo aver richiesto una conferma all'utente, setta alcune variabili in modo che, quando il controlla torna al main, viene generata una nuova parola; infine, il punteggio del giocatore viene dimezzato. La funzione non restituisce nessun valore.

- ✓ **void inseriscinomi(player*&);**

Acquisisce i nomi dei partecipanti alla sessione di gioco, li rende maiuscoli e li memorizza in una lista circolare, invocando più volte la funzione *inscirc*. La funzione non restituisce nessun valore.

- ✓ **int giralaruota(void);**

Genera e restituisce un valore casuale compreso tra 100 e 1000, utilizzando la funzione *rand*.

- ✓ **void pulisci_schermo(void);**

Stampa a schermo una serie di *endl* per cancellare tutto ciò che c'è scritto sullo schermo. La funzione non restituisce nessun valore.

- ✓ **void preminvio(void);**

Blocca l'esecuzione del programma finché l'utente non preme *invio*. La funzione non restituisce nessun valore.

- ✓ **void istruzioni(void);**

Stampa a video il regolamento del gioco. Data la lunghezza, la visualizzazione è stata spezzata in due parti: dopo la prima parte, viene chiesto all'utente di premere invio per visualizzare la seconda parte. La funzione non restituisce nessun valore.

- ✓ **void inscirc(player*&, player*&, char[]);**

Inserisce un nome nella lista circolare semplice contenente i nomi dei giocatori e i relativi punteggi. Se la lista è vuota, inserisce in testa, altrimenti inserisce l'elemento dopo l'ultimo e lo collega alla testa della lista. La funzione non restituisce nessun valore.

- ✓ **void stampaclassifica(player*);**

Stampa a video l'elenco dei giocatori con i relativi punteggi, partendo dal giocatore di turno fino ad arrivare alla fine della lista circolare, identificata dal ritorno al nodo del giocatore di turno. La funzione non restituisce nessun valore.

- ✓ **void liberamemoria (player*&, char*[], int);**

Dealloca le locazioni di memoria occupate dalla lista circolare e dall'array dinamico, prima di terminare il programma.

Note dell'autore

La fase più impegnativa di questo progetto è stata l'ideazione, in quanto è stato difficile trovare un'idea di cosa realizzare, che non fosse troppo banale. Una volta scelto di realizzare il gioco della ruota della fortuna, arrivato circa a metà della scrittura del codice, ho dovuto fermarmi per circa una settimana per cercare un errore che si presentava quando inserivo la risposta alla frase da indovinare, giusta o sbagliata che fosse, perché il programma mi entrava in un ciclo infinito e si bloccava. Dopo giorni di prove e analisi del codice, sono riuscito a trovare l'errore (che, come sempre accade in questi casi, era banale), che consisteva nell'omissione (per una svista) di un parametro che era da passare alla funzione *cin.ignore*.

Una volta terminata la stesura del codice, che in totale mi ha impegnato circa un mese, nell'ultima settimana ho provveduto ad ottimizzare il programma, ad esempio aggiungendo la funzione *liberamemoria*. Infine, ho provveduto a redigere questo documento, che spero possa chiarire, assieme ai commenti che ho inserito nel codice, com'è stato progettato e realizzato questo programma.

Rimango a disposizione per chiarimenti.

Davide Valeriani
davide.valeriani@studenti.unipr.it