Università degli Studi di Napoli Federico II

Corso di Laurea in Informatica

Laboratorio di Programmazione (Gr.3)

Compito del 19/10/2022

Dott. Andrea Apicella

REGOLE

- *Della libreria standard del C, è accettato l'utilizzo **solo** dei seguenti file header:*
  - *stdio.h*
  - *stdlib.h*
  - *math.h*
  - *string.h*
  - *time.h*

  *L'utilizzo di qualsiasi altro file header e relative funzioni della libreria standard **non sarà accettato**;*
- *Seguire scrupolosamente le direttive nella traccia ed utilizzare eventuali nomi di variabili e/o funzioni dati senza variazioni (anche in termini di maiuscole/minuscole e caratteri _);*
- *Ogni file sorgente dovrà contenere nelle prime righe un commento nel formato:*

```
/* MATRICOLA: ...
COGNOME: ...
NOME: ... */
```

- *E' fortemente consigliato commentare il codice il più possibile;*
- *E' fortemente consigliato modulare il progetto su più file, nello specifico almeno 3:*
  1. *contenente i prototipi e le definizioni di eventuali strutture;*
  2. *contenente la definizione della funzione main(). Nel caso siano richiesti più main(), fare un file diverso per ognuno di essi;*
  3. *contenente le definizioni delle funzioni rimanenti.*

  Se lo si ritiene opportuno, è possibile separare i sorgenti in più di 3 file, motivandolo con dei commenti;
- ***NON** includere il file eseguibile ed eventuali file oggetto nella consegna.*
- *La consegna finale dovrà quindi essere un unico archivio zip contenente (almeno) 4 file:*
  - *i file sorgenti (almeno 3, come specificato sopra);*
  - *il file 'istruzioni.txt', contenente i comandi per la compilazione/linking come richiesti nella traccia.*

  Verificare che tutti i file siano stati correttamente compressi e caricati!

TRACCIA DEL 19/10/2022, TEMPO: 2 ore

Il nascente social network *Fakebook* contiene i *Post* degli utenti in specifiche strutture dati così formate:

- *msg*: messaggio di testo di al più 256 caratteri; per semplicità, si supponga che tale campo non contenga caratteri di a capo al suo interno;
- *n_like*: numero di utenti che apprezzano il post;

Per un singolo utente, le informazioni sono memorizzate in un file di testo nel seguente formato:

```
msg1
n_like1
msg2
n_like2
.
.
.
```

Esempio:

```
01/01/2010: buongiornissimo!1!1!11!!kaffè??
144
11/01/2010: non ho superato l'esame! :(
5
10/02/2010: neanche questa volta ho passato l'esame! >:(
2
16/04/2010: è la terza volta che rifaccio quest'esame!!! >:@
1
20/05/2024: Ho finalmente superato l'esame! :D
14
```

Punto 1:: implementare una struttura dati *Bacheca* con politica di accesso *coda* (*First In First Out*) in grado di memorizzare dei *Post*. Tale struttura dovrà fornire almeno le seguenti funzioni:

- *append(Bacheca, Post)*: inserisce il nuovo post all'interno della bacheca, rispettando la politica di accesso;
- *pop(Bacheca)*: restituisce il collegamento (e rimuove dalla bacheca) il prossimo post, rispettando la politica di accesso;
- *len(Bacheca)*: restituisce il numero di elementi contenuti in bacheca;
- *is_empty(Bacheca)*: restituisce 1 se la bacheca è vuota, 0 altrimenti.

Tale struttura dati dovrà essere implementata attraverso un array di puntatori a Post allocato dinamicamente.

Punto 2: scrivere una funzione *load_user_from_file(Bacheca,nomefile)* che carica tutti i dati contenuti nel file di testo di nome $nomefile$ all'interno della *Bacheca*. Il file di testo è fornito con la traccia, ed ha nome "utente42.txt". L'accesso alla bacheca deve essere effettuato sfruttando soltanto le funzioni definite al punto 1.

Punto 3: nella funzione *main()*, stampare tutti gli elementi della bacheca uno per volta, dando la possibilità all'utente di scegliere se:

- fermarsi;
- andare al prossimo post;
- aggiungere un like al post attuale;
- ”condividere” il post attuale. In questo contesto, ”condividere” il post equivale a salvarlo in coda in un file di testo di nome ”condivisi.txt”.

L'accesso alla bacheca deve essere effettuato sfruttando soltanto le funzioni definite al punto 1. I post dovranno essere ancora presenti all'interno della struttura dati Bacheca al termine dello scorrimento.

E' consigliato implementare una funzione *print(Post)* che stampi il contenuto di *un singolo* Post passato come argomento.

Punto 4: compilare ed eseguire il programma da linea di comando tramite *gcc*. Riportare su di un file di testo di nome 'istruzioni.txt' i comandi necessari per effettuare la generazione del file eseguibile. Le compilazioni dei file contenenti le funzioni diverse dal *main()* dovranno essere effettuate *distintamente*. La compilazione del file contenente il *main()* dovrà essere effettuata assieme al linking con tutti i rimanenti file oggetto. Il file eseguibile dovrà avere nome **utente**_numerodimatricola**.eseguibile**. Consegnare soltanto i file sorgenti, escludendo i file oggetto ed esegubili dalla consegna.