Fondamenti di Informatica e Laboratorio T-AB e Fondamenti di Informatica T1
Ingegneria Elettronica e Telecomunicazioni e
Ingegneria dell'Automazione
a.a. 2010/2011

Lab 15
Gestione file di testo

Esercizio 1

- Realizzare un programma che, aperto un file di testo di nome “Prova.txt” in modalità “scrittura”, provveda a leggere da input delle parole separate da spazi (stringhe di al più 63 caratteri) e le scriva nel file di testo.
- Il programma termina quando l’utente inserisce la parola “fine”. Si abbbia cura di chiudere il file prima di terminare definitivamente il programma.
- Si controlli il corretto funzionamento del programma accedendo direttamente al file di testo con un editor (ad es. Notepad).

Esercizio 1 - Soluzione

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main(void) {
    FILE * fp;
    char s[64];

    if ((fp=fopen("prova.txt", "w")) == NULL)
      exit(1);

    do {
      scanf("%s", s);
      if (strcmp("fine", s) != 0)
       fprintf(fp, "%s ", s);
    } while (strcmp("fine", s) != 0);

    fclose (fp);
    system("PAUSE");
    return (0);
}
```

Esercizio 2

Sia dato il file di testo "dati.txt" contenente i dati relativi agli studenti immatricolati al primo anno della Facoltà di Ingegneria

In particolare, le informazioni sono memorizzate nel file "dati.txt" come segue: ognuna delle linee del file contiene i dati relativi ad un nuovo studente; in particolare:

1 Matricola: un intero che indica il numero di matricola dello studente

2 CdL: un intero che indica il corso di laurea (CdL) dello studente (es. 2145)

Esercizio 2

Sia dato un secondo file di testo, “indirizzi.txt” che contiene, invece, l’indirizzo di ogni studente, e in particolare, in ogni linea, separati da uno spazio:

- Matricola: il numero di matricola dello studente (un intero)
- Nome: il nome dello studente, al più 20 caratteri senza spazi
- Cognome: il cognome dello studente, al più 30 caratteri senza spazi
- Via: una stringa di al più 30 caratteri senza spazi, che riporta la via di residenza dello studente
- Città: una stringa che riporta la città di residenza dello studente, al più 30 caratteri senza spazi
- CAP: un intero che rappresenta il codice di avviamento postale dello studente

Esercizio 2

Si scriva un programma in linguaggio C che:

1. A partire dai file "dati.txt" e “indirizzi.txt” **costruisca una tabella T** contenente, per ogni studente, Matricola, Nome, Cognome, Via, Città, CAP e CdL

Si ricorda l’esistenza della procedura di libreria void rewind (FILE *f) che riporta la testina di lettura a inizio file

Esercizio 2

2. A partire dalla tabella T, e dato da input un intero C che rappresenta un CdL, **stampi la percentuale di studenti** (rispetto al numero totale delle matricole) **iscritti al corso C** [Ad esempio, se il numero totale delle matricole è 1000, e quello degli studenti iscritti a C è 200, il programma stamperà “20%”]

3. Scriva su un terzo file di testo “bologna.txt”, nome, cognome e numero di matricola di tutti gli studenti che abitano a Bologna

Esercizio 2 – Soluzione

```
typedef struct {
    unsigned int matr;
    unsigned int CDL;
} dati;

typedef struct {
    unsigned int matr;
    char nome[21];
    char cognome[31];
    char via[31];
    char citta[31];
    unsigned int CAP;
} indirizzo;
```

```
typedef struct {
        unsigned int matr;
        char nome[21];
        char cognome[31];
        char via[31];
        char citta[31];
        unsigned int CAP;
        unsigned int CDL;
} elemento;

typedef elemento tabella[10];
```

Esercizio 2 – Soluzione

```
elemento riempiel(dati d, indirizzo i){
     elemento e;
     e.matr=d.matr;
     e.CDL=d.CDL;
     strcpy(e.nome, i.nome);
     strcpy(e.cognome, i.cognome);
     strcpy(e.via, i.via);
     strcpy(e.citta, i.citta);
     e.CAP=i.CAP;
     return e;
}


int main() {
     dati D;
     indirizzo I;
     elemento E;
     tabella T;
     FILE *f1, *f2;
     int i, trovato, ins=0, totC;
     unsigned int C;
```

Esercizio 2 – Soluzione

```
/*domanda 1: costruzione della tabella */

     f1=fopen("dati.txt", "r");
     f2=fopen("indirizzi.txt", "r");

     while (fscanf(f1,"%u%u", &D.matr, &D.CDL)>0) {
           trovato=0;
           rewind(f2);
           while(fscanf(f2, "%d %s %s %s %s %d",
                            &I.matr, I.nome, I.cognome,
                            I.via, I.citta, &I.CAP)==6
                       && !trovato)
                 if (I.matr==D.matr) {
                       trovato=1;
                       E=riempiel(D, I);
                       T[ins]=E;
                       ins++;
                 }
      }

     fclose(f1);fclose(f2);
     ...
```

Esercizio 2 – Soluzione

```
    /*domanda 2: stampa della percentuale degli
   iscritti a un corso dato*/
    printf("Inserire il corso C: ");
    scanf("%u", &C);
    totC=0;
    for(i=0; i<ins; i++)
      if(T[i].CDL==C)
           totC++;

    printf("\n Iscritti al corso %u: %f \%\n",
           C, (float)totC*100/ins);


    /*domanda 3: scrittura di "bologna.txt" */
    f1=fopen("bologna.txt", "w");
    for (i=0; i<ins; i++)
       if (strcmp("bologna", T[i].citta)==0)
           fprintf(f1,"%s %s %u\n",T[i].nome,T[i].cognome,T[i].matr);
    fclose(f1);
    return 0;
}
```

Esercizio 3

Sono dati due file di testo cineprogramma.txt e sale.txt che contengono, rispettivamente, il programma settimanale dei film in proiezione e le descrizioni delle sale in città. Più precisamente, ogni riga di cineprogramma.txt contiene, nell'ordine:

- titolo del film (non più di 30 caratteri senza spazi), uno e un solo spazio di separazione
- nome della sala (non più di 20 caratteri senza spazi), uno e un solo spazio di separazione
- 3 orari di inizio proiezione (3 numeri interi separati da caratteri '-'), terminatore di riga

mentre ogni riga di sale.txt contiene, nell'ordine:

- nome della sala (non più di 20 caratteri senza spazi), uno e un solo spazio di separazione
- costo del biglietto (numero reale), terminatore di riga

Esercizio 3

cineprogramma.txt:

```
TheKingdom Nosadella 18-20-22
Dogville Fellini 17-20-22
OttoEMezzo Capitol 17-20-23
BreakingWaves Odeon 15-19-23
```

sale.txt:

```
Capitol 6.00
Fellini 5.50
Modernissimo 6.00
Nosadella 6.50
```

Esercizio 3

1) Si scriva una procedura load() che riceva come parametri di ingresso due puntatori a file di testo e restituisca come parametri di uscita **un vettore y contenente strutture film** (titolo film, costo biglietto) **e il numero degli elementi N inseriti in y**

Per semplicità si supponga che tutte le sale contenute nel primo file siano presenti anche nel secondo, e una sola volta

Si ricorda inoltre l'esistenza della procedura di libreria void rewind (FILE *f) che riporta la testina di lettura a inizio file

Esercizio 3

2) Si scriva un programma C che, utilizzando la procedura load() precedentemente definita, **inserisca in un vettore prezzi** (supposto di dimensione massima DIM=100) **le strutture film di cui sopra**, derivanti dai file cineprogramma.txt e sale.txt

Il programma deve inoltre stampare a terminale tutti gli elementi di prezzi il cui costo del biglietto è **inferiore alla media di tutti i costi caricati nel vettore**

Esercizio 3 - Soluzione

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define DIM 100
typedef struct {
    char titolo[31];
    float prezzo;
} film;

void load(FILE * f1, FILE * f2, film * y, int * num) {
    char titolo[31], sala1[21], sala2[21];
    int orario; float p; *num=0;

    while (fscanf(f1,"%s %s %d-%d-%d\n", titolo, sala1, &orario,
                                          &orario, &orario) != EOF) {
       while ((fscanf(f2,"%s %f\n",sala2,&p) != EOF)&&
                            (strcmp(sala1,sala2))); //nessuna azione
// supponiamo per semplicità che la sala sia sempre compresa in f2
       strcpy(y[*num].titolo, titolo);
       y[*num].prezzo=p;
       (*num)++;
       rewind(f2); }
}
```

Esercizio 3 - Soluzione

```
int main() {
   FILE *f1, *f2; int N, i;
   float somma,media; film prezzi[DIM];

   if ((f1=fopen("cineprogramma.txt", "r"))==NULL) {
     printf("Non sono riuscito ad aprire file1 in lettura!");
     exit(1);
   }
   if ((f2=fopen("sale.txt", "r"))==NULL) {
     printf("Non sono riuscito ad aprire file2 in lettura!");
     exit(1);
   }

   load(f1,f2,prezzi,&N);
   fclose(f1); fclose(f2);
   ...
```

Esercizio 3 – Soluzione

```
   ...
   somma=0;
   for (i=0; i<N; i++) somma=somma+prezzi[i].prezzo;
   media=somma/N;

   for (i=0; i<N; i++)
     if (prezzi[i].prezzo<media)
          printf("Il costo del biglietto per il film %s è
                      %f\n", prezzi[i].titolo,
   prezzi[i].prezzo);
}
```

Esercizio 4

Sono dati due file di testo anagrafe.txt e fatture.txt che contengono, rispettivamente, i dati anagrafici di alcuni clienti e l'elenco delle fatture

Più precisamente, ogni riga di anagrafe.txt contiene, nell'ordine:

- Codice Cliente (numero intero) , uno e un solo spazio di separazione
- Nome del cliente (non più di 30 caratteri senza spazi), uno e un solo spazio di separazione
- Città (non più di 20 caratteri senza spazi), uno e un solo spazio di separazione

Ogni cliente compare nel file di anagrafe una ed una sola volta

Ogni riga di fatture.txt contiene, nell'ordine:

- Codice Cliente (numero intero), uno e un solo spazio di separazione
- Numero della fattura (numero intero), uno e un solo spazio di separazione
- Importo della fattura (numero reale), uno e un solo spazio di separazione
- Un carattere ('p' se la fattura è stata pagata, 'n' altrimenti), terminatore di riga

Esercizio 4

anagrafe.txt:

```
1 Chesani Bologna
2 Bellavista Bologna
3 Mello Bologna
```

fatture.txt:

```
1 23 54.00 p
1 24 102.00 n
3 25 27.00 p
1 26 88.00 n
```

Esercizio 4

1) Si scriva una procedura load() che riceva come parametri di ingresso due puntatori a file di testo e restituisca come parametri di uscita **un vettore y contenente strutture debito** (nome cliente, importo) e il numero degli elementi N inseriti in y: questo vettore deve contenere solo i dati relativi a **fatture non pagate**

Si ricorda inoltre l'esistenza della procedura di libreria void rewind (FILE *f) che riporta la testina di lettura a inizio file

Esercizio 4

2) Si scriva un programma C che, utilizzando la procedura **load()** precedentemente definita, inserisca in un vettore **debitori** (supposto di dimensione massima **DIM=100**) le strutture **debito** di cui sopra, derivanti dai file **anagrafe.txt** e **fatture.txt**

Il programma chieda poi all'utente il nome di un cliente e stampi il numero di fatture (non pagate) intestate a tale cliente e la somma totale degli importi dovuti

Esercizio 4 - Soluzione

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define DIM 100

typedef struct {
   char nome[31];
   float importo;
} debito;
```

Esercizio 4 - Soluzione

```
void load(FILE * f1, FILE * f2, debito * y, int * num) {
   int codice1, codice2;
   char nome[31], citta[21], pagato;
   int num_fattura; float importo;

   *num=0;
   while(fscanf(f1,"%d %s %s\n", &codice1, nome, citta) != EOF){
         while ( fscanf(f2,"%d %d %f %c\n", &codice2,
               &num_fattura,&importo, &pagato) != EOF )

               if ( (codice1==codice2) && (pagato == 'n') ) {
                     strcpy(y[*num].nome, nome);
                     y[*num].importo = importo;
                     (*num)++;
               }
         rewind(f2);
   }
}
```

Esercizio 4 - Soluzione

```
int main() {
   FILE *f1, *f2; int N, i, count; float somma;
   char nome[31]; debito debitori[DIM];

   if ((f1=fopen("anagrafe.txt", "r"))==NULL) {
     printf("Non sono riuscito ad aprire file1 in lettura!");
     exit(1);
   }
   if ((f2=fopen("fatture.txt", "r"))==NULL) {
     printf("Non sono riuscito ad aprire file2 in lettura!");
     exit(1);
   }

   load( f1, f2, debitori, &N);
   fclose(f1);fclose(f2);
   ...
```

Esercizio 4 - Soluzione

```
   ...
   printf("Inserire nome cliente: ");
   scanf("%s", nome);

   somma = 0; count = 0;
   for (i=0; i<N; i++) {
     if (! strcmp(nome, debitori[i].nome)){ /* strcmp==0 */
       somma = somma + debitori[i].importo;
       count++;
     }
   }

   printf("%d fatture per l'ammontare di: %f", count, somma);
}
```