Fondamenti di Informatica L-A (A.A. 2003/2004) - Ingegneria Informatica
Prof.ssa Mello & Prof. Bellavista – Prova d'Esame di Lunedì 12/01/2004 – durata 2h:30m
COMPITO A

ESERCIZIO 1 (12 punti)

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), terminatore di riga.

Ogni cliente compare nel file **`anagrafe.txt`** 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.

Ad esempio:

`anagrafe.txt`	**`fatture.txt`**
1 Chesani Bologna 2 Bellavista Bologna 3 Mello Bologna	1 23 54.00 p 1 24 102.00 n 3 25 27.00 p 1 26 88.00 n .....

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 **(6 punti)**.

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 **(6 punti)**.

ESERCIZIO 2 (7 punti)

Si scriva una funzione **`f()`** che data in ingresso una lista ordinata di interi **`l1`**, restituisca in uscita una nuova lista, ottenuta da **`l1`** eliminando tutti gli elementi ripetuti. Ad esempio, se invocata con **l1=[1, 3, 3, 3, 5, 8, 8]**, la funzione **`f()`** deve restituire la lista **[1, 3, 5, 8]**.
La funzione **`f()`** può essere realizzata in modo ricorsivo o iterativo, utilizzando il tipo di dato astratto **`list`** e le operazioni primitive sul tipo **`list`** definite durante il corso (che quindi possono essere NON riportate nella soluzione).

ESERCIZIO 3 (6 punti)
Il seguente programma C compila correttamente? In caso affermativo, quali sono i valori stampati a tempo di esecuzione (si motivi opportunamente la risposta data)?

```
#include <stdio.h>
#include <stdlib.h>
#define DIM 9

int Funz(char *, int *, int *);
int N = DIM - 7;

main () {
      char s[] = "Paperone"; int i=3;
      N = (++N)-2;
      N = Funz(s, &N, &i);
      printf("N vale adesso: %d\n",N);
      {     char * ss = malloc(DIM);
            for (i=0; i<DIM-1; ++i) {
                  *(ss+i) = s[i];
                  printf("%c", *(ss+i)); }
      }
}

int Funz(char y[], int *N, int *M) {
      int i;
      (*N)++;
      for (i=(*N)-1; i<*N; i++)
            y[i] = y[*M];
      return *N; }
```

Esercizio 4 (6 punti)
Si consideri la seguente funzione **F**:

```
float F(float x){
      x=x-1;
      if (x>0) return F(x)+F(x-2);
      else return 1;}
```

Si scriva il risultato della funzione quando invocata come **F(3)** e si disegnino i corrispondenti record di attivazione.

Esercizio 5 (2 punti)
Si consideri la grammatica **G** con scopo **S** e simboli terminali **{e, m, c, *, =, ^, 2}**

```
S::= Z X | Y W | X
X::= Y X | W Z | Z
Y::= W Z W | W W | Z Z
W::= e | m | c
Z::= * | ^ | = | 2
```

Si dica se la stringa **`e = m c 2`** è sintatticamente corretta rispetto a tale grammatica e se ne mostri la derivazione *left most*.

SOLUZIONE COMPITO A

ESERCIZIO 1

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define DIM 100

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

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);
        }
}

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);

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

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

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

ESERCIZIO 2

```
list f(list l1) {
      if (empty(l1)) return l1;
      else
            if (empty(tail(l1)) return l1;
            else {
                  if (head(l1) == head(tail(l1))) return f(tail(l1));
                  else return cons(head(l1), f(tail(l1)); }
}
```

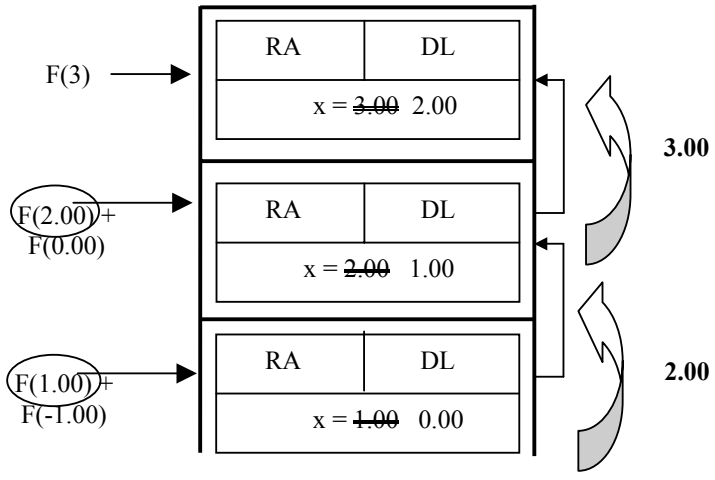
ESERCIZIO 3

Il programma è corretto sintatticamente e stampa:
N vale adesso: 2
Peperone

Infatti la stringa s viene inizializzata a "Paperone", terminatore compreso. Successivamente viene effettuato un semplice calcolo su N, dichiarato globalmente. Prima dell'invocazione di Funz(), N vale 1. La funzione Funz() è quindi invocata con parametri il vettore di caratteri s, e i valori 1 e 3. La funzione Funz() si limita a sostituire alcuni caratteri in determinate posizioni; in particolare il ciclo sostituisce il carattere in posizione 1 con il carattere in posizione 3. Il valore restituito è il secondo parametro incrementato di 1, ovvero 2. Infine il programma alloca memoria per una stringa grande DIM, copia i caratteri dal vettore s alla zona allocata e puntata da ss, e di volta in volta stampa anche tali caratteri.

ESERCIZIO 4

La funzione restituisce il valore 3.00. Supponendo che la valutazione degli addendi nella somma venga fatta a partire da sinistra, si ottiene prima:



poi l'eliminazione dell'ultimo record, e un nuovo ulteriore record di attivazione per F(-1.00). Quindi viene fatta la prima somma con risultato 2.00, restituita indietro dove viene fatta l'ultima invocazione di F(0.00), con somma finale restituita alla prima invocazione di F().

ESERCIZIO 5

La stringa è sintatticamente corretta. Derivazione *left most*:

S → X → Y X → W Z W X → e Z W X → e = W X →
e = m X → e = m W Z → e = m c Z → e = m c 2