Fondamenti di Informatica L-A (A.A. 2003/2004) - Ingegneria Informatica
Prof.ssa Mello & Prof. Bellavista – Prova d'Esame di lunedì 13 settembre 2004 – durata 2h30m

ESERCIZIO 1 (14 punti)

Una compagnia aerea usa un sistema automatico di registrazione delle prenotazioni su file di tipo binario (al massimo 144 prenotazioni). Ogni posto disponibile su un aereo è caratterizzato dal numero di fila (da 1 a 24) e dal numero di sedia in tale fila (da 1 a 6), e viene rappresentato tramite una apposita struttura **`posto`** che contiene appunto i due interi **`fila`** e **`sedia`**. Ogni prenotazione è poi rappresentata tramite una struttura dati, di nome appunto **`prenotazione`**, composta dei seguenti campi:

- nome (array di non più di 64 caratteri)
- cognome (array di non più di 64 caratteri)
- codice aereo (un intero)
- una struttura posto (descritta sopra)

Si vuole fornire una nuova funzionalità che permetta di sapere quale posto ha prenotato un cliente, a partire dal suo nome e cognome. A tal scopo si realizzi quindi una funzione **`trova_posto()`**

```
posto trova_posto(FILE *f1, const char *nome, const char *cognome);
```

La struttura **`posto`** deve rappresentare il posto effettivamente prenotato dal cliente o, in caso il cliente non abbia prenotato nessun posto, deve indicare la fila 0 e la sedia 0. La funzione deve ovviamente leggere tutte le strutture **`prenotazione`** presenti sul file indicato, e selezionare solo la prenotazione corrispondente al nome-cognome specificati. Per determinare se nome e cognome corrispondono ad una certa prenotazione, si consideri l'uso della funzione di libreria **`strcmp`**, che ritorna 0 quando due stringhe sono uguali. (**punti 7**)
Il candidato realizzi poi un programma C che (1) chieda ad un utente nome e cognome; (2) utilizzando la funzione **`trova_posto`** definita in precedenza, legga i dati presenti nel file **`passeggeri.dat`**; (3) stampi a video la fila e la sedia corrispondente al nome–cognome inseriti, o stampi un opportuno messaggio di errore qualora il nominativo non sia presente nel file. Il candidato si ricordi inoltre di definire opportunamente le strutture **`posto`** e **`prenotazione`** come descritte in precedenza. **(punti 7)**

ESERCIZIO 2 (6 punti)

Si scriva una funzione **`hop()`** che date in ingresso due liste **`l1`** e **`l2`** di interi, restituisca una terza lista contenente gli interi di entrambe le liste secondo il seguente criterio: l'elemento della lista **`l1`** deve sempre essere inserito; l'elemento della lista **`l2`** deve essere inserito solo se l'elemento corrispondente di **`l1`** è pari, scartato altrimenti. Le liste **`l1`** e **`l2`** sono non ordinate, ma hanno la stessa lunghezza. Ad esempio, se **`l1=[1,2,6,7] e l2=[3,9,5,4]`**, il risultato deve essere la lista **`[1,2,9,6,5,7]`**.
La funzione **`hop()`** 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 NON essere 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 MAX 15

int inverti(char *, char []);
int C = MAX - 5;

int main () {
    char * a;
    char * b = "Schumacher";
    int i;

    a = malloc(sizeof(char) * MAX);
    C = inverti(a, b);
    printf("C vale adesso: %d\n",C);
    for (i=0; i<C; i++) printf("%c", a[i]);
    return 0;
}

int inverti(char * x, char y[] ) {
     int limit = C, i = 0;

     while (limit > 0) {
          x[i] = y[limit-1];
          i++; limit--;
     }
     return i;
}
```

ESERCIZIO 4 (4 punti)

Si consideri la seguente funzione **FIBBIA()**:

```
int FIBBIA(int a, int b){
      a=a-b;
      if ((a/b)>0) return FIBBIA(a, b) + a;
      else return 0; }
```

Si scriva il risultato della funzione quando invocata come **FIBBIA(7, 2)** e si disegnino i corrispondenti record di attivazione.

ESERCIZIO 5 (3 punti)

Un elaboratore rappresenta i numeri interi su 8 bit dei quali 7 sono dedicati alla rappresentazione del modulo del numero e uno al suo segno. Indicare come viene svolta la seguente operazione aritmetica e determinarne il risultato traslandolo poi in decimale per la verifica:

38 - 91

SOLUZIONE

ESERCIZIO 1

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

#define DIM 144

typedef struct {
      int fila;
      int sedia;
} posto;

typedef struct {
      char nome[65];
      char cognome[65];
      int cod_volo;
      posto p;
} prenotazione;

posto trova_posto(FILE *f1, const char *nome, const char *cognome) {
      int i;
      int letti = 0;
      prenotazione pren[DIM];
      posto result;
      result.fila = 0;
      result.sedia = 0;

      letti = fread(pren, sizeof(prenotazione), DIM, f1);
      for (i=0; i<letti; i++) {
            if ( (strcmp(pren[i].nome, nome) == 0) &&
                  (strcmp(pren[i].cognome, cognome) == 0) ) {
                        result = pren[i].p; i=letti;}
      }
      return result; }

int main() {
FILE *f1;
char nome[65]; char cognome[65];
posto result;

if ((f1=fopen("passeggeri.dat", "rb"))==NULL) {
      printf("Non sono riuscito ad aprire file1 in lettura!"); exit(1); }

printf("Nome da cercare: "); scanf("%s", nome);
printf("Cognome da cercare: "); scanf("%s", cognome);

result = trova_posto(f1, nome, cognome);
fclose(f1);

if ((result.fila==0) && (result.sedia==0))
      printf("Il passeggero %s %s non ha prenotato un posto.\n", nome, cognome);
else
      printf("Il passeggero %s %s ha prenotato il posto in fila %d, sedia %d",
      nome, cognome, result.fila, result.sedia);

return 0; }
```

ESERCIZIO 2

```
list hop(list l1, list l2) {
      if (empty(l1)) return l1;
      else if ((head(l1)%2) == 0)
            return cons(head(l1), cons(head(l2), hop(tail(l1),tail(l2))));
            else return cons(head(l1), hop(tail(l1), tail(l2)));
}
```

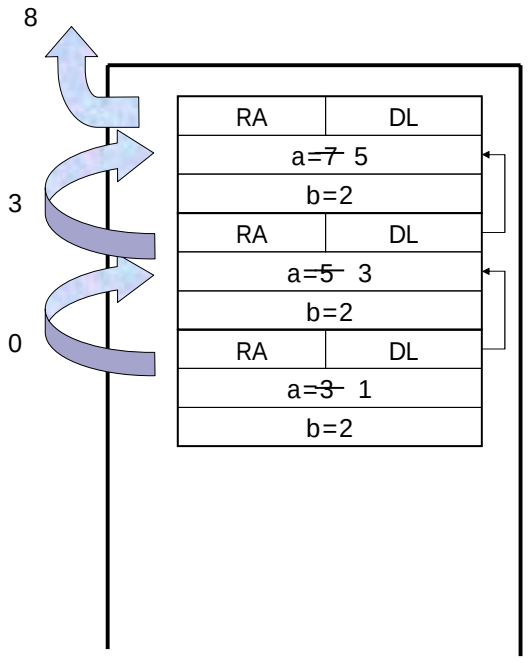
ESERCIZIO 3

Il programma è corretto sintatticamente e stampa:
C vale adesso: 10
rehcamuhcS

Infatti la stringa b viene inizializzata a "Schumacher", terminatore compreso. Successivamente viene allocato dinamicamente uno spazio di MAX byte, dove MAX è definito tramite una direttiva al pre-processore e vale 15; l'indirizzo iniziale di tale spazio viene assegnato ad a. Quindi viene invocata la procedura inverti() con parametri a e b. La funzione si limita a copiare i valori dal vettore b al vettore a, invertendoli nell'ordine (a tal scopo usa due indici, che incrementa/decrementa ad ogni ciclo). Da notare che non viene copiato il terminatore di stringa di b: quindi al termine della funzione a non è una stringa ben formata. Ciò non importa particolarmente, poiché a viene stampata tramite un ciclo, che si ferma proprio all'ultimo carattere, e quindi non c'è pericolo di accedere a zone di memoria non correttamente inizializzate o prive di significato. Infine viene stampato a video il valore di C, variabile definita a livello globale, ancora uguale a 10, come prima dell'invocazione della funzione.

ESERCIZIO 4

La funzione restituisce il valore 8, attraverso la seguente sequenza di record di attivazione:



ESERCIZIO 5

```
38->        0   0100110
91->        0   1011011
```

Tra i moduli dei numeri si esegue una sottrazione ottenendo: `1  0110101`
che vale -53 in base dieci.