Fondamenti di Informatica L-A (A.A. 2007/2008) - Ingegneria Informatica
Prof.ssa Mello & Prof. Bellavista – Prova d'Esame di Giovedì 13 Dicembre 2007 – durata 2h
Compito A

ESERCIZIO 1 (10 punti)

Una banca salva in una struttura dati di nome **`cliente`** le informazioni relative alle azioni possedute dai propri clienti. In tale struttura viene memorizzato il numero di conto corrente (un intero detto **`ccNum`**), un array di interi detto **`codiciTitoli`**, contenente identificatori unici di titoli azionari, e un intero **`dim`** che rappresenta la dimensione (logica) del vettore dei codici titoli (cioè il numero di titoli azionari posseduti dal cliente).

a) Il candidato realizzi una funzione:

```
int * rischio(cliente c[], int dim1, int tr[], int dim2, int *dim)
```

che, ricevuti come parametri un vettore di strutture dati **`c`** (di tipo **`cliente`**) e la sua dimensione **`dim1`**, ed un vettore di codici di titoli (a rischio elevato) **`tr`** e di dimensione **`dim2`**, selezioni i clienti a rischio. La funzione deve restituire in un apposito vettore allocato dinamicamente (non necessariamente della dimensione strettamente necessaria) tutti i numeri di conti correnti tali che nella rispettiva struttura dati **`cliente`** vi siano registrati almeno due titoli azionari a rischio tra quelli indicati in **`tr`**. Tramite il parametro **`dim`** passato per riferimento, la funzione deve restituire la dimensione logica del vettore restituito come risultato. Si riporti, per completezza, anche la definizione della struttura dati **`cliente`**.

b) Si supponga di avere a disposizione, già implementate, le funzioni:

```
cliente * leggiClienti(FILE * fp, int * dim);
int * leggiMIB(char * filename, int * dim);
```

Tali funzioni leggono da file i dati relativi ai clienti e i dati relativi ai titoli a rischio, restituendo in un caso un array di strutture dati **`cliente`** (sotto forma di un puntatore a **`cliente`**), e nell'altro un array di interi contenente i codici dei titoli azionari a rischio. Entrambe le funzioni restituiscono la dimensione logica dei rispettivi vettori tramite il parametro **`dim`** passato per riferimento. Il candidato realizzi un programma che, usando le funzioni appena descritte e la funzione definita al punto precedente, stampi a video l'elenco dei conti correnti che hanno investimenti in titoli azionari rischiosi.

ESERCIZIO 2 (10 punti)

Si supponga di avere a disposizione, già definiti, l'ADT lista per interi (denominato come al solito **`list`**, con relative primitive) e un nuovo ADT, denominato **`list_l`**, che rappresenta il concetto usuale di lista, definita però per elementi che a loro volta sono liste di interi. Il candidato supponga di possedere le usuali primitive (denominate però **`empty_l(…)`**, **`emptylist_l(…)`**, **`cons_l(…)`**, **`head_l(…)`**, **`tail_l(…)`**) definite opportunamente per il nuovo ADT **`list_l`**. Il candidato definisca una funzione *ricorsiva*

```
list_l exact(list_l ll)
```

che, ricevuta in ingresso una lista **`ll`** i cui elementi sono a loro volta liste di interi, restituisca in uscita una nuova lista (di tipo **`list_l`**) contenente tutte le sotto-liste di interi tali che il seguente criterio sia soddisfatto: è da includersi ogni lista di interi la cui lunghezza sia pari all'ultimo intero registrato in tale lista. Ad esempio, se invocata con parametro **`ll= [ [9,2,3,4], [5,6,7,12] ]`**, la funzione deve restituire come risultato la lista **`[[9,2,3,4]]`**. Infatti la prima sotto-lista ha quattro elementi, e l'ultimo valore incluso nella sotto-lista è proprio 4. A tal scopo si utilizzino solo le operazioni primitive degli ADT **`list`** e **`list_l`**.

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>

int dim = 0;

int sec(char * a, char b[]) {
      char * c;
      while (a[dim] != '\0') {
            c = b;
            while (*c != '\0') {
                  if (*c == a[dim])
                        return dim;
                  c = c + 1;
            }
            dim++;
      }
      return -1;
}

int main(void) {
      int dim = 1, pos;
      char s1[] = "Paperoga";
      char s2[] = "Ape";
      char * s3;

      pos = sec(s1, s2);
      s3 = s1 + pos + dim;
      while (*s3 != '\0') {
            printf("%c", *s3);
            s3++;
      }
      return (0); }
```

ESERCIZIO 4 (4 punti)

Data la funzione:

```
int doppio(float a, int b){
      if ( (a==b) || (a>b))
            return 0;
      else
            return doppio(a+1, b-2) + doppio(a+1, b-1) + 3;
}
```

mostrare la sequenza dei record di attivazione nel caso in cui la funzione sia invocata con parametri attuali (0, 3.3).

ESERCIZIO 5 (2 punti)

Un elaboratore rappresenta i numeri interi su 8 bit tramite la notazione in complemento 2. Indicare come viene svolta la seguente operazione aritmetica e determinarne il risultato traslandolo poi in decimale per la verifica:

47+(-101)

ESERCIZIO 1

```
typedef struct {
      int ccNum;
      int * codiciTitoli;
      int dim;
} cliente;

int * rischio(cliente c[], int dim1, int tr[], int dim2, int *dim) {
      int i, j, k, trovato=0;

      int * result = (int *) malloc(sizeof(int)*dim1);
      *dim = 0;

      for (i=0; i<dim1; i++) {
            trovato=0;
            for (j=0; j<c[i].dim && trovato<2; j++)
                  for (k=0; k<dim2 && trovato<2; k++)
                        if (c[i].codiciTitoli[j] == tr[k])
                              trovato++;
            if (trovato>=2) {
                  result[*dim] = c[i].ccNum;
                  *dim = *dim + 1;
            }
      }
      return result;
}

cliente * leggiClienti(FILE * fp, int * dim) {
      cliente * result;
      *dim = 0;
      result = (cliente *) malloc(sizeof(cliente) * 256);
      while (!feof(fp) && *dim<256)
            if (fread(result+(*dim), sizeof(cliente), 1, fp) == 1) *dim = *dim + 1;
      return result;
}

int * leggiMIB(char * filename, int * dim) {
      FILE * fp;
      int * result;
      int codice;
      float andamento;
      *dim = 0;
      if ((fp = fopen(filename, "rt")) == NULL)
            exit(-1);
      result = (int *) malloc(sizeof(int)*1024);
      while (!feof(fp) && *dim<256)
            if (fscanf(fp, "%d %f", &codice, &andamento) == 2)
                  if (andamento < -10.0) {
                        result[*dim] = codice;
                        *dim = *dim + 1;
                  }
      fclose(fp);
      return result;
}
```

```
int main(void) {
      FILE * fp;
      cliente * elencoClienti;
      int dimClienti;
      int * elencoTitoli;
      int dimTitoli;
      int * clientiRischio;
      int dimRischio, i;

      if ((fp = fopen("clienti.bin", "rb")) == NULL)
            exit(-1);
      elencoClienti = leggiClienti(fp, &dimClienti);
      fclose(fp);

      elencoTitoli = leggiMIB("mib.txt", &dimTitoli);

      clientiRischio = rischio(elencoClienti, dimClienti,
                                    elencoTitoli, dimTitoli, &dimRischio);
      for (i=0;i <dimRischio; i++)
            printf("Cliente: %d\n", clientiRischio[i]);
      return (0);
}
```

ESERCIZIO 2

```
list_l exact(list_l master)
{
      int i = 0, last=0;
      list temp;

      if (empty_l(master))
            return emptylist_l();
      else {
            temp = head_l(master);
            while (!empty(temp)){
                  if (empty(tail(temp)))
                        last = head(temp);
                  temp = tail(temp);
                  i++;
            }
            if (i == last)
                  return cons_l(head_l(master), exact(tail_l(master)));
            else return exact(tail_l(master));
      }
}
```

ESERCIZIO 3

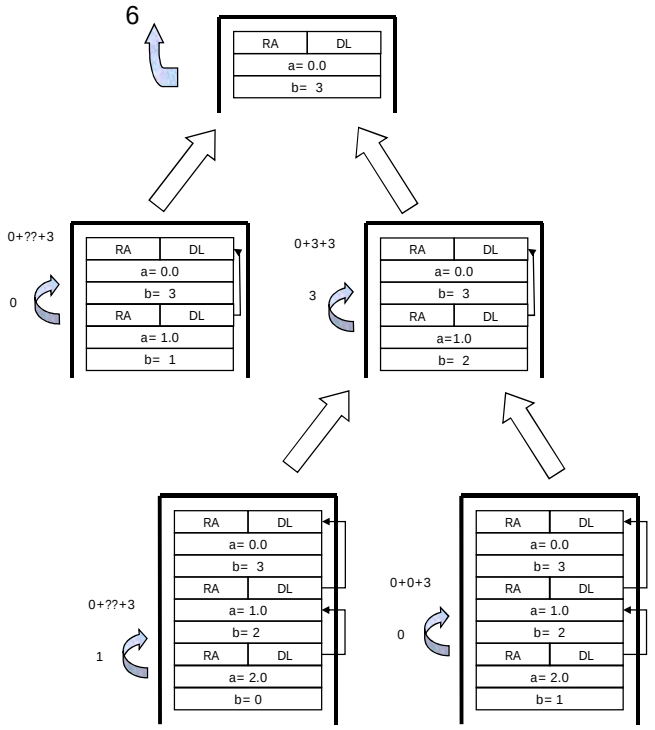
Il programma è corretto sintatticamente e la sua esecuzione produce la stampa:

eroga

Nella funzione main() vengono assegnate alle variabili s1 e s2 due stringhe, e poi viene invocata la funzione sec(). Tale funzione cerca la prima occorrenza di un qualunque carattere di b in a: non appena trova tale occorrenza, restituisce l'indice di tale carattere in b. Poiché la funzione viene invocata con parametri reali a="Paperoga" e b="Ape", la prima occorrenza di un carattere di b in a è il carattere 'p' all'indice 2 rispetto al vettore a.

main() assegna al puntatore c il vettore s3, incrementato del valore restituito dalla funzione sec(), cioè 2, più il valore della variabile locale dim, cioè 1. Infine la stringa identificata dal puntatore c viene stampata a video: essa non è altro che la stringa s1, stampata a partire dall'elemento di indice 3.

ESERCIZIO 4



ESERCIZIO 5

```
47  -> 00101111                    00101111+
101 -> 01100101                    10011011=
       10011010                    ---------
-101-> 10011011                   11001010 ->-54
                                   00110101
                                   00110110 ->54
```