Fondamenti di Informatica L-A (A.A. 2006/2007) - Ingegneria Informatica
Prof.ssa Mello & Prof. Bellavista – Prova d'Esame di Giovedì 29 Marzo 2007 – durata 2h
Compito B

ESERCIZIO 1 (9 punti)

Si considerino due stringhe ben formate **s1** e **s2,** contenenti dei caratteri ripetuti più volte. Il candidato realizzi una funzione:

```
char * diff(char *s1, char *s2, int *newDim)
```

che, ricevuti come parametri tali stringhe, restituisca come risultato un nuovo vettore di caratteri allocato dinamicamente (puntatore a **char**). Tale vettore deve contenere i caratteri presenti nella stringa **s1** , ma non in **s2**; inoltre i caratteri che eventualmente sono ripetuti in **s1** devono comparire una sola volta nel risultato. Il candidato abbia cura di allocare dinamicamente la memoria sufficiente (si consideri che, al massimo, il vettore risultante sarà grande come **s1**). Tramite il parametro **newDim**, la funzione deve anche restituire la dimensione (eventualmente "logica") del vettore risultato. Per determinare la dimensione di una stringa, si faccia eventualmente uso della funzione di libreria **strlen(…).**

Ad esempio, se **s1="abcdabi"** e **s2="ailia"**, il risultato deve essere un vettore di dimensione logica 3, con contenuto **['b', 'c', 'd']**.

ESERCIZIO 2 (9 punti)

Si supponga di avere a disposizioneADT lista per interi (denominato come al solito **list**, con relative primitive). Si definisca una funzione ricorsiva:

```
list differenti(list l1, list l2)
```

che, ricevute in ingresso le liste **l1** e **l2** (entrambe contenenti eventuali valori ripetuti), restituisca in uscita una lista contenente, per ogni elemento di **l1**, il numero di elementi di **l2** differenti dall'elemento di **l1** preso in considerazione. Ad esempio, se invocata con parametro **l1 = [1, 2, 3]** e **l2 = [1, 2, 1, 1]**, la funzione deve restituire come risultato la lista **[1, 3, 4]**. A tal scopo si utilizzino le sole operazioni primitive degli ADT **list**.

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 p = 0;

void deCrypt(char * s, int pos) {
     char ch; int i=0;

     ch = s[pos];
     if (ch != '\0') {
          if (ch >='0' && ch <='9') {
               i = ch-'0';
               ch = s[i];
               printf("%c", ch);
          }
          if (i<pos)  deCrypt(s, pos+1);
          else deCrypt(s, i+1);
     }
     else
          printf("\n");
}

int main(void) {
     char s[] = "abc25i08o";
     deCrypt(s, p);
     return (0); }
```

ESERCIZIO 4 (5 punti)

Si considera la seguente funzione:

```
int strange(float x, char y) {
      x=x-y;
      if ((x+y)/2 >=0) return strange(x+1, y) + strange(x, y);
      else return y;
}
```

Si dica che cosa restituisce se invocata con parametri **`strange(2.0, 3)`** e si mostrino i record di attivazione relativi.

ESERCIZIO 5 (3 punti)

Si definiscano brevemente i vari componenti che costituiscono un moderno compilatore e le varie operazioni che vengono svolte da ciascuno di essi. Inoltre, quali sono le principali differenze q tempo di esecuzione fra un programma compilato e uno interpretato?

ESERCIZIO 1

```
char * diff(char * s1, char * s2, int * newDim) {
      int i, j, trovato;
      char * result;

      *newDim = 0;
      result = (char *) malloc(sizeof(char) * strlen(s1));

      for (i=0; s1[i]!='\0'; i++) {
            trovato=0;
            for (j=0; j<*newDim && !trovato; j++) {
                  if (result[j]==s1[i]) trovato=1;
            }
            for (j=0; s2[j]!='\0' && !trovato; j++) {
                  if (s1[i] == s2[j]) trovato = 1;
            }
            if (!trovato) {
                  result[*newDim] = s1[i];
                  *newDim = *newDim + 1;
            }
      }
      return result;
}
```

ESERCIZIO 2

```
list differenti(list l1, list l2) {
      int i = 0;
      list temp;

      if (empty(l1)) return emptylist();
      else {
            temp = l2;
            while (! empty(temp)) {
                  if (head(l1) != head(temp))
                        i++;
                  temp = tail(temp);
            }
            return cons(i, differenti(tail(l1), l2) );
      }
}
```

ESERCIZIO 3

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

Nella funzione **`main(…)`** viene creata una stringa ben formata, contenente sia caratteri che numeri, e viene poi invocata la funzione **`deCrypt(…)`**. Tale funzione ricorsiva comincia con l'assegnare a **`ch`** il valore in indice **`pos`** (passato come parametro). Se tale valore non è una cifra viene ignorato e la funzione si re-invoca ricorsivamente con indice **`pos+1`**. Se invece viene salvato un carattere rappresentante una cifra, in **`ch`** viene posto il valore dell'ulteriore carattere presente a tale indice **`i`** nella stringa (considerata come array di caratteri). Tale carattere viene poi stampato e la funzione si re-invoca ricorsivamente, a partire dalla posizione successiva il maggiore tra gli indici **`pos`** ed **`i`** (appena determinato).

ESERCIZIO 4