

Fondamenti di Informatica T-1 (A.A. 2015/2016) - Ingegneria Informatica
Prof.ssa Mello
Prova Parziale d'Esame di Giovedì 8 Settembre 2016 – durata 1h
Totale 12 punti, sufficienza con 7

Compito A

ESERCIZIO 1 (6 punti)

Sia data una lista `l1` di interi strettamente positivi. Si realizzi la funzione ITERATIVA

```
list calcolaSomme(list l1);
```

che ritorni una nuova lista `l2` in cui, per ogni elemento `k` in `l1`, si inserisca in `l2` la somma dei numeri interi da 1 a `k` (`k` compreso).

A tal fine si implementi una funzione RICORSIVA:

```
int somma(int k)
```

che calcoli la somma dei numeri da 1 a `k` (`k` compreso)

Si realizzi infine una funzione `main()` che crei una lista `l1` ed utilizzi correttamente la funzione `calcolaSomme(l1)` precedente in modo tale da calcolare la lista `l2`. Ad esempio, data la lista `l1 = {4, 1, 5, 6, 6}` si dovrà ottenere la lista `l2 = {10, 1, 15, 21, 21}`

Le funzioni dovranno essere implementate utilizzando le primitive dell'ADT lista, includendo "`list.h`".

ESERCIZIO 2 (2 punti)

Si consideri la seguente funzione:

```
int h(int z) {
    int y = -3;

    if (y == z)
    {
        return 1;
    }
    else
    {
        return 1 - h(--z);
    }
}
```

Mostrare la sequenza dei record di attivazione e il valore di ritorno nel caso in cui la funzione sia invocata con parametri attuali `(-1)`.

ESERCIZIO 3 (3 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>

int f(int* a, int b, int* c){
    int count=0;
    while (b>=0 && (*c)!=0)
    {
        if (a[b] / (*(a+b+1)) < 1)
        {
            b++;
            (*c)--;
        }
        else
        {
            --b;
        }
        count++;
    }
    printf("%d %d\n",b, *c);
    return b+count;
}

int main(){
    int x[]={4,5,3,7,1};
    int y=0;
    int z=2;
    int res=f(x,y,&z);
    printf("%d %d %d\n",y,z,res);
    return 0;
}
```

ESERCIZIO 4 (1 punto)

Si illustri, anche con l'ausilio di un esempio, la differenza tra allocazione statica e dinamica di un array.

Soluzioni

ESERCIZIO 1

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include "list.h"

int somma(int n)
{
    if (n == 0)
    {
        return 0;
    }
    else
    {
        return n + somma(n-1);
    }
}

list calcolaSomme(list l1)
{
    list tmp = emptyList();
    list l2 = emptyList();

    // Si inseriscono gli elementi nell'ordine inverso
    while (l1 != NULL)
    {
        tmp = cons(somma(head(l1)), tmp);
        l1 = tail(l1);
    }

    // Ed è quindi poi necessario invertire la lista
    while(!empty(tmp)){
        l2 = cons(head(tmp), l2);
        tmp = tail(tmp);
    }

    return l2;
}

int main()
{
    list L1 = emptyList();
    list L2 = emptyList();
    list tmp;

    L1 = cons(6, L1);
    L1 = cons(6, L1);
    L1 = cons(5, L1);
    L1 = cons(1, L1);
    L1 = cons(4, L1);
```

```

L2 = calcolaSomme(L1);

showList(L1);
showList(L2);

while(!empty(L1)) { tmp = L1; L1 = tail(L1); free(tmp); }
while(!empty(L2)) { tmp = L2; L2 = tail(L2); free(tmp); }

return 0;
}

```

ESERCIZIO 2

La funzione restituisce valore 1.

RA	DL	-1
y = -3, z = -1		
RA	DL	0
y = -3, z = -2		
RA	DL	1
y = -3, z = -3		

ESERCIZIO 3

L'output prodotto è

```

1  0
0  0  4

```

Il programma `main` crea un array `x` di interi ed invoca la funzione `f`, utilizzando come parametri l'array stesso e le variabili `y=0` (passata per valore) e `z=2` (passata per riferimento). Il risultato è memorizzato in `res`.

La funzione `f` esegue una prima volta il ciclo `while` ed effettua la divisione tra interi tra il secondo ed il primo elemento dell'array (parametro `a` nella funzione), ovvero 4 e 5. Essendo il risultato 0, la variabile `b` viene incrementata, mentre il contenuto della variabile puntata da `c` viene decrementato. Quindi `b=1` e `*c=1`. Successivamente, viene incrementato `count` a 1. Il ciclo `while` viene eseguito una seconda volta, e stavolta si effettua la divisione tra gli interi 5 e 3. Essendo il risultato pari a 1, la variabile `b` viene decrementata. Quindi `b=0`. Anche la variabile `count` viene incrementata, ed assume valore 2. Il ciclo `while` viene eseguito una terza volta, e si esegue nuovamente la divisione tra 4 e 5. Essendo il risultato pari a 0, `b` viene nuovamente incrementata, mentre il contenuto della variabile puntata da `c` viene

decrementato. Quindi `b=1` e `*c=0`, mentre `count=3`.

A questo punto il ciclo `while` non esegue più, si stampano i valori `1` e `0`, e si ritorna al `main` il valore `b+count`, ovvero `4`.

Nel `main` vengono stampati i valori di `y` (non modificato, quindi `0`), `*z` (a sua volta `0`) e `res` (ovvero `4`).