PROGRAMMAZIONE 1 e LABORATORIO (A,B) - a.a. 2014-2015
Prova scritta del 18/12/2014
SOLUZIONI PROPOSTE

ESERCIZIO 1
Dato il tipo degli alberi binari

```
type 'a btree = Void  | Node of 'a * 'a btree * 'a btree
```

si definisca in `CAML` una funzione `pfoglie` con tipo

```
pfoglie : ('a -> bool) -> 'a btree -> int
```

in modo che `(pfoglie p bt)` restituisca il numero dei nodi foglia di `bt` la cui etichetta soddisfa `p`.

Soluzione

```
let rec pfoglie p bt = match bt with
  Void  ->  0
| Node(x, Void, Void) -> if p x then 1 else 0
| Node(x, lbt, rbt) when lbt<>Void or rbt<>Void -> (pfoglie p lbt) + (pfoglie p rbt)
```

ESERCIZIO 2
Definire in CAML una funzione

```
foo : int list -> int
```

in modo che `(foo lis)` restituisca `-1` se `lis` contiene solo elementi negativi e restituisca il primo elemento non negativo di `lis` altrimenti. Ad esempio:

```
foo [-1;4;-2;5;0;-5] = 4
foo [-10;-2;-8] = -1
```

Facoltativo: Definire la funzione `foo` senza utilizzare ricorsione esplicita.

Soluzione
Soluzione ricorsiva

```
let rec foo lis =
     match lis with
       [x] -> if x<0 then -1 else x
     | x::y::ys -> if x<0 then foo (y::ys) else x
```

Soluzione con foldr

```
let foo lis = let f x y = if x<0 then y else x
                 in match lis with
                 _::_ -> foldr f -1 lis;;
```

ESERCIZIO 3
Indicare il tipo delle seguenti funzioni

(i) `let f x y = (x 2) y;;` (ii) `let g (x,y) z = (x y) + (z x);;`

(iii)
```
let rec h x y = match y with
    [] -> true
  | z::zs -> (z x) & (h x zs);;
```

Soluzione

```
let f x y = (x 2) y;;
f : (int -> 'a -> 'b) -> 'a -> 'b

let g (x,y) z = (x y) + (z x);;
g : ('a -> int) * 'a -> (('a -> int) -> int) -> int

let rec h x y  = match y with
        [] ->  true
      | z::zs -> (z x) & (h x zs);;
h : 'a -> ('a -> bool) list -> bool
```

ESERCIZIO 4
Definire una funzione ricorsiva f da coppie di naturali in naturali che soddisfi la proprietà

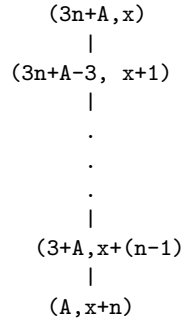
$$\forall n, m \in \mathbb{N}.\ f(n, m) = n + m$$

in modo che la relazione di precedenza indotta su $\mathbb{N} \times \mathbb{N}$ sia la seguente

$$\forall n, m, n', m' \in \mathbb{N}. \langle n, m\rangle \prec_f \langle n', m'\rangle \equiv (n' = n + 3 \wedge m = m' + 1)$$

Facoltativo: Dimostrare per induzione ben fondata la correttezza della definizione data.

Soluzione
Una possibile rappresentazione grafica della relazione indicata è la seguente



dove `x,n` indicano generici numeri naturali e `A` è `0`, `1` o `2`.

Dunque:

$$f(n, m) = \begin{cases} n + m & \text{se } n < 3 \\ f(n - 3, m + 1) + 2 & \text{se } n \geq 3 \end{cases}$$

Omettiano la domostrazione dei casi base. Dimostriamo il caso induttivo:

$$\forall n, m \in \mathbb{N}.\ f(n, m + 1) = n + m + 1 \Longrightarrow f(n + 3, m) = n + 3 + m$$

$f(n + 3, m)$
= {def. di f, secondo caso}
$f(n, m + 1) + 2$
= {Ip. induttiva}
$n + m + 1 + 2$
= {calcolo}
$n + 3 + m$

ESERCIZIO 5
Date le seguenti definizioni:

```
struct el {int info; struct el *next;};

typedef struct el ElementoDiLista;
typedef ElementoDiLista *ListaDiInteri;
```

scrivere in `C` una procedura che, dati in ingresso attraverso opportuni parametri una lista di interi e un valore `x`, elimina, se esiste, il primo elemento della lista seguito da un elemento che contiene `x`.

Soluzione

```
void proc (ListaDiInteri *lis, int x)
   { if (*lis != NULL)
        if (*lis->next != NULL)
           { ListaDiInteri prec=NULL, corr=*l;
             int trovato = 0;
             while (corr->next != NULL && !trovato)
                if (corr -> next -> info == x)
                     trovato=1;
                else {prec = corr; corr = corr -> next;}
             if (trovato)
                 { if (prec==NULL)
                         then *l=*l->next;
                         else prec->next=corr->next;
                   free(corr);

                 }
           }
   }
```