Laboratorio di Linguaggi e Programmazione Orientata agli Oggetti

31 ottobre 2019

a.a. 2019/2020

Definire in OCaml le seguenti funzioni.

1. `prod : int list -> int`
   `prod` l restituisce il prodotto di tutti i numeri interi contenuti in l.

2. `gen_sum_list : ('a -> int) -> 'a list -> int`
   `gen_sum_list` è la versione per liste della funzione `gen_sum` vista a lezione: `gen_sum_list` f `[`v_1`;...;`v_n`]` restituisce $fv_1 + \ldots + fv_n$.

3. `member : 'a -> 'a list -> bool`
   `member` e l restituisce `true` se e solo se e è un elemento di l.

4. `insert : 'a -> 'a list -> 'a list`
   `insert` e l restituisce la lista ottenuta aggiungendo e in fondo a l se e non appartiene a l; restituisce l altrimenti.

5. `reverse : 'a list -> 'a list`
   `reverse` l restituisce la lista ottenuta rovesciando l.

6. `odd : 'a list -> 'a list`
   `odd` l restituisce la lista ottenuta da l tenendo solo gli elementi di posizione dispari (considerando che le posizioni cominciano da 1).

   Esempio: `odd [1;2;3;4;5] = [1;3;5];;`

7. `ord_insert : 'a -> 'a list -> 'a list`
   `ord_insert` e l restituisce la lista ordinata in modo crescente e senza ripetizioni ottenuta aggiungendo e a l, assumendo che l sia ordinata in modo crescente e senza ripetizioni.

8. `merge : 'a list * 'a list -> 'a list`

   `merge` (l_1, l_2) restituisce la lista ordinata in modo crescente e senza ripetizioni ottenuta fondendo assieme le liste ordinate in modo crescente e senza ripetizioni l_1 ed l_2.

   Esempio: `merge ([1;3;5],[2;4;6]) = [1;2;3;4;5;6];;`

9. `curried_merge : 'a list -> 'a list -> 'a list`
   `curried_merge` è la versione curried di `merge`; definire la funzione senza usare `merge`.