ESERCITAZIONE 17

FONDAMENTI DI INFORMATICA

2 dicembre 2021
michele.chiari@polimi.it

Esercizio 1

Liste

Dati i tipi definiti di seguito:

```
typedef struct N {
  int val;
  struct N *next;
} Node;
typedef Node *Ptr_node;
```

implementare le seguenti funzioni:

- `Ptr_node list_create(int v, Ptr_node t)` – la funzione crea (mediante `malloc()`) e restituisce un puntatore ad un nuovo nodo di una lista; il nodo contiene il valore v e punta a t come elemento successivo.
- `Ptr_node nth_element(Ptr_node list, int n)` – la funzione deve restituire il puntatore all'n-esimo nodo della lista (n=0 è il primo nodo, n=1 è il secondo nodo, ecc.). Se la lista ha meno di n+1 nodi, deve restituire NULL;
- `Ptr_node list_from_array(int v[], int n)` – la funzione deve creare una nuova lista contenente gli n elementi dell'array `v`;
- `Ptr_node list_concat(Ptr_node L1, Ptr_node L2)` – la funzione deve restituire una lista contenente i nodi di `L1` seguiti da quelli di `L2`. N.B., la funzione non deve creare nuovi nodi, ma utilizzare quelli già presenti nelle due liste.

Esercizio 2

Lista Somma
Scrivere un sottoprogramma che, ricevuti in ingresso una lista e due numeri N e M, restituisca la somma di tutti gli elementi della lista il cui valore è compreso fra N e M.
Ad esempio, con
1 3 5 7 9 11, N= 6, M = 10
Si ottiene
7 + 9 = 16

Esercizio 3

Primo Pari

Definire una funzione `primoPari` che, data una lista, restituisca il puntatore al primo elemento pari nella lista (restituisce NULL se la lista è vuota o non contiene elementi pari).

Esercizio 4

Ripetizioni

Scrivere un sottoprogramma che, ricevuta in ingresso una lista per la gestione di numeri interi, restituisce la lista in modo tale che tutti gli elementi adiacenti uguali siano ridotti ad un solo elemento.