FONDAMENTI DI PROGRAMMAZIONE A

Tempo a disposizione: 1 ora 40 minuti

Nome ..... Cognome ..... Matricola .....

Esercizio 1 (6pt).

Scrivere una funzione di tipo `int` chiamata `compact` che, presi come parametri un array di numeri interi `a` e il numero `n` di elementi in `a` ed un array di numeri interi `b`, memorizza nell'array `b` gli elementi di `a` senza ripetizioni (se un elemento è presente più di una volta dovrà essere memorizzato la prima volta che compare). La funzione deve ritornare il numero di elementi di `a` senza ripetizioni. Per esempio:

```
a = {1, 2, 7, 1, 2, 4, 1, 7, 5}   // (n = 9)
b = {1, 2, 7, 4, 5} // la funzione ritorna 5
```

Esercizio 2 (9pt). Sia `intv` il tipo di una struttura dati `struct` formata da due campi di tipo intero `inf` e `sup` che rappresentano rispettivamente il limite inferiore e quello superiore di un intervallo chiuso `[inf, sup]`. Si implementi una funzione `add` che accetta come parametri una struttura `i` di tipo `intv` e un intero `x` e aggiunge, se possibile, l'elemento `x` all'intervallo `i`, modificando `i` stesso. Tale operazione è possibile quando (*i*) `x` è contenuto nell'intervallo `i` (e in questo caso `i` rimane invariato) oppure (*ii*) quando `x` coincide con il predecessore di `inf` o con il successore di `sup` dell'intervallo `i` (in questo caso `i` viene modificato aggiornando rispettivamente `inf` o `sup`). In questi casi la funzione deve ritornare `true` altrimenti ritorna `false`.

Per esempio, se `i = [3, 7]` e `x = 8` il nuovo intervallo sarà `i = [3, 8]` e la funzione ritorna `true` , mentre se `x = 9` l'intervallo rimane invariato e la funzione ritorna `false`

Esercizio 3 (15pt). Scrivere una funzione di tipo `bool` di nome `equals` che, presi come parametri due liste concatenate semplici `l1` e `l2` i cui elementi hanno campo informazione di tipo `int`, ritorna `true` se le due liste sono uguali, `false` altrimenti. Gestire opportunamente i casi in cui `l1` o `l2` siano vuote. (**+2pt se la funzione è ricorsiva**)