Prova Pratica - Prima Prova in Itinere

Il progetto deve essere svolto da **un** solo studente.

È altamente **sconsigliato** di copiare codice on-line o da altri elaborati, poiché questo provoca l'annullamento della consegna con conseguente esclusione dalla prove in itinere e dagli esoneri scritti.
È, invece, **consigliato** di manipolare il codice visto a lezione e disponibile sul sito della parte pratica del corso.

La scelta del progetto 1, 2 o 3 avverrà secondo questo algoritmo:

1. Prendi la seconda lettera del tuo cognome e trasformala nel suo valore intero ($a = 97$, $b = 98$, … , $z = 122$)
2. Calcola il numero relativo al tuo progetto calcolando $x \; mod \; 3 + 1$

Ad esempio, per il cognome "Italiano", il progetto da svolgere sarà il numero 3, poiché $t = 116$ e $(116 \; mod \; 3) + 1 = 2 + 1$

Documentazione da consegnare:

- breve relazione contenente
    - descrizione dell'algoritmo
    - analisi del tempo di esecuzione teorico e sperimentale
- codice sorgente
    - implementazione dell'algoritmo
    - demo dell'algoritmo

Tutto il materiale dovrà essere caricato su un repository online (ad esempio GitHub, Dropbox, Drive,...), il cui **link** dovrà essere inviato via email con tutti i tutor in cc entro il 15 dicembre 2017.

L'elaborato sarà valutato secondo metriche di:

- correttezza dell'algoritmo
- chiarezza del codice
    - commenti
    - modularità
- completezza della relazione

Problema 1 - Binary search tree con lazy deletion

Un albero binario di ricerca con lazy deletion è composto da nodi che possono essere segnati come “eliminati”. Siano definite le seguenti funzioni:

- **bool delete(key k)**: Se il nodo con chiave k è presente nell’albero e non è segnato come “eliminato”, segna il nodo come eliminato e ritorna True. Se il nodo con chiave k non è presente nell’albero, o è presente ma è stato già precedentemente segnato come eliminato, ritorna False.
- **insert(key k, value v)**: Inserisce una nuova coppia (k,v) nell’albero binario. Se è possibile inserire il nodo con chiave k nella posizione di un nodo segnato come eliminato, sostituisce il vecchio nodo con il nuovo.
- **search(chiave k)**: Ritorna il nodo con chiave k se questo è presente nell’albero e non è segnato come eliminato.

Basandosi sul codice presentato a lezione e presente sul sito del corso, implementare un dizionario che fa uso della struttura dati sopra descritta.

Nota: Nella relazione si discuta in quali casi è preferibile l’uso della lazy deletion.

Problema 2 - Concatenazione di alberi AVL

Siano dati due alberi AVL A e B tali che le chiavi di uno siano tutte strettamente minori delle chiavi dell’altro. Progettare e implementare un algoritmo che restituisca un nuovo albero AVL ottenuto come concatenazione di A e B.

Problema 3 - Trovare un elemento in un array

Sia dato un array A di n interi tale che n sia multiplo di 7.

Progettare e implementare un algoritmo che trovi, se esiste, un numero x che soddisfi entrambe le seguenti proprietà:

- x appartiene al primo settimo dell'array, cioè
  $x = A[i]\ t.\ c.\ 1 \leq i \leq n/7$
- x deve trovarsi nell'ultimo settimo dell'array ordinato, cioè
  $x = A[j]\ t.\ c.\ 6n/7 \leq i \leq n$