Istruzioni per la consegna e la discussione del progetto di ASD (A.A. 25-26)

Istruzioni

Tutti i progetti di Algoritmi e Strutture Dati prevedono la realizzazione di una libreria (insieme di funzioni, tipi e costanti) in C. È consentito l'uso di funzionalità C++ solo limitatamente all'input/output da tastiera e/o file.

Accanto alla libreria realizzata, è **obbligatorio** sviluppare una funzione main che ne mostri degli esempi d'uso.

Il progetto è **personale**.

Cosa deve essere consegnato:

- un file zip contenente il codice sorgente e le istruzioni per compilarlo (meglio se presente un makefile)

Il progetto sarà discusso online. Condizioni necessarie ad una valutazione positiva sono la **compilazione** e la **corretta esecuzione** del programma. Si consiglia di svolgere la dimostrazione dell'esecuzione su una macchina di propria scelta (es., il proprio laptop).

Il progetto può essere discusso in una qualsiasi delle date previste (una per ciascun appello) indicate nel sito del corso. Il voto ottenuto sarà valido per l'intero anno accademico (in tutti gli appelli disponibili fino a novembre), **indipendentemente** da quando viene discusso. La consegna del file deve avvenire prima della data scelta per la discussione.

Il progetto **non è di libera scelta** ma viene assegnato compilando questo form (necessario autenticarsi con le credenziali istituzionali @uniroma1.it)

L'assegnazione del progetto è consultabile qui (necessario autenticarsi con le credenziali istituzionali @uniroma1.it), la corrispondenza con l'indice del progetto è riportata nella prossima pagina di questo documento.

Il file .zip del progetto deve essere nominato con il numero di matricola (es. 123456.zip)
e consegnato usando questo form (necessario autenticarsi con le credenziali istituzionali @uniroma1.it)

Progetti disponibili
(Assegnati automaticamente)

1. Libreria per la gestione di un Dizionario rappresentato come Albero AVL

2. Libreria per la gestione di un Dizionario rappresentato come Hash Table, con risoluzione delle collisioni mediante liste di collisione

3. Libreria per la gestione di un Dizionario rappresentato come Hash Table, con risoluzione delle collisioni mediante indirizzamento aperto

4. Libreria per l'ordinamento di array di interi, contenente i seguenti algoritmi:
    a. InsertionSort
    b. SelectionSort
    c. MergeSort
    d. QuickSort

5. Libreria per l'implementazione del tipo astratto Grafo rappresentato mediante liste di adiacenza e implementazione ricorsiva di una funzione per la verifica che un nodo sorgente, fornito in input, sia connesso ad un nodo destinazione, fornito in input.

6. Implementazione dell'algoritmo di Bellman-Ford per il calcolo dell'albero dei cammini minimi in un grafo generico. (Deve includere le strutture per la rappresentazione del grafo e le funzioni necessarie all'effettiva esecuzione del programma).

7. Implementazione dell'algoritmo di Bellman-Ford per il calcolo dell'albero dei cammini minimi in un grafo aciclico. (Deve includere le strutture per la rappresentazione del grafo e le funzioni necessarie all'effettiva esecuzione del programma, incluso il calcolo dell'ordinamento topologico).

8. Implementazione dell'algoritmo di Dijkstra. (Deve includere le strutture per la rappresentazione del grafo e le funzioni necessarie all'effettiva esecuzione del programma).

9. Libreria per l'implementazione del tipo astratto Albero rappresentato mediante liste di successori e implementazione di una funzione ricorsiva per il conteggio dei nodi dell'albero. (Deve includere le strutture per la rappresentazione dell'albero e le funzioni necessarie all'effettiva esecuzione del programma).

10. Libreria per l'implementazione del tipo astratto Grafo rappresentato mediante matrice di adiacenza e implementazione di una funzione che verifichi se il grafo sia fortemente connesso. (Deve includere le strutture per la rappresentazione del grafo e le funzioni necessarie all'effettiva esecuzione del programma).

11. Libreria per l'implementazione del tipo astratto Grafo rappresentato mediante liste di adiacenza e implementazione ricorsiva di una funzione che, preso in input un insieme S di nodi del grafo, restituisca il sottografo indotto da S (Deve includere le

strutture per la rappresentazione del grafo e le funzioni necessarie all’effettiva esecuzione del programma).