Algoritmi e Strutture Dati
Esercitazione 4 (Laboratorio)

Esercizio 1

1) Implementare in C la funzione

- `AlberoBinario creaPreHeap(int* a, int n)`

che, preso in input un vettore `a` di `n` interi, restituisce un albero completo fino al penultimo livello contenente tutti e soli gli elementi del vettore di input come nodi. (NOTA: è necessario definire anche la struttura dati per la rappresentazione dell'albero binario).

2) Implementare in C la funzione

- `int* heapSort(int* a)`

che realizza l'algoritmo Heap Sort.

3) Fornire un'implementazione alternativa di `int* heapSort(int[] a)` che operi esclusivamente sull'array di input, senza fare uso di alcuna struttura collegata ausiliaria. È ammesso l'uso di un numero costante di variabili ausiliarie.

Esercizio 2

1) Implementare in C la funzione

- `int* vettorePosizionale(AlberoBinario a)`

che, preso in input un albero binario con nodi etichettati da interi, in rappresentazione collegata, ne restituisce il corrispondente *vettore posizionale*.

Si ricorda che il vettore posizionale associato ad un albero è un vettore tale che:

- l'elemento in posizione 1 corrisponde al nodo radice dell'albero;
- se l'elemento in posizione i corrisponde al nodo v dell'albero, allora gli elementi in posizione $2i$ e $2i + 1$ corrispondono, rispettivamente, al figlio sinistro e al figlio destro di v.