Domanda 37 Scrivere una funzione `IsABR(A)` che dato in input un array di interi `A[1..n]`, interpretato come albero binario (come nel caso degli heap, ogni `A[i]` è un nodo con figlio sinistro e destro `A[2i]` e `A[2i+1]`) verifica se `A` è un albero binario di ricerca. Valutarne la complessità.

Esercizio 11 Sia dato un albero i cui nodi contengono una chiave intera $x.key$, oltre ai campi $x.l$, $x.r$ e $x.p$ che rappresentano rispettivamente il figlio sinistro, il figlio destro e il padre. Si definisce *grado di squilibrio* di un nodo il valore assoluto della differenza tra la somma delle chiavi nei nodi foglia del sottoalbero sinistro e la somma delle chiavi dei nodi foglia del sottoalbero destro. Il grado di squilibrio di un albero è il massimo grado di squilibrio dei suoi nodi.

Fornire lo pseudocodice di una funzione `sdegree(T)` che calcola il grado di squilibrio dell'albero T (si possono utilizzare funzioni ricorsive di supporto). Valutare la complessità della funzione.

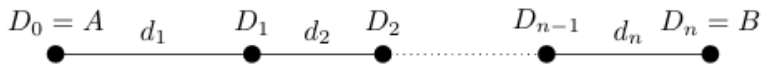
Domanda 23 Scrivere una funzione `IsMaxHeap(A)` che dato in input un array di interi `A[1..n]` che verifica se `A` è organizzato a max-heap e ritorna un corrispondente valore booleano. Valutarne la complessità.

Domanda 14 Dare una soluzione asintotica per la ricorrenza $T(n) = 3T(n/2) + n(n+1)$.

Domanda 17 Dare la definizione di $\Omega(f(n))$. Mostrare che se $f(n) = \Omega(g(n))$ e $g(n) = \Omega(h(n))$ allora $f(n) = \Omega(h(n))$.

9 Greedy

Esercizio 33 Si supponga di voler viaggiare dalla città A alla città B con un'auto che ha un'autonomia pari a d km. Lungo il percorso si trovano $n-1$ distributori $D_1, \ldots, D_{n-1}$, a distanze di $d_1, \ldots, d_n$ km ($d_i \leq d$) come indicato in figura



L'auto ha inizialmente il serbatoio pieno e l'obiettivo è quello di percorrere il viaggio da A a B, minimizzando il numero di soste ai distributori per il rifornimento.

i. Introdurre la nozione di soluzione per il problema e di costo della una soluzione. Mostrare che vale la proprietà della sottostruttura ottima e individuare una scelta che gode della proprietà della scelta greedy.

ii. Sulla base della scelta greedy individuata al passo precedente, fornire un algoritmo greedy `stop(d,n)` che dato in input l'array delle distanze `d[1..n]` restituisce una soluzione ottima.

iii. Valutare la complessità dell'algoritmo.

Esercizio 1 (9 punti) Realizzare una funzione `avgTree(T)` che dato un albero binario `T` con chiavi numeriche, verifica se, per ogni nodo che abbia discendenti, la chiave del nodo è maggiore o uguale della media delle chiavi dei discendenti e ritorna conseguentemente un valore booleano (la radice dell'albero è `T.root` e ogni nodo `x` ha i campi `x.left` `x.right` e `x.key`).

Domanda B (7 punti) Dare la definizione di albero binario di ricerca. Specificare l'albero ottenuto inserendo, con la procedura vista a lezione, a partire da un albero vuoto, i nodi aventi le seguenti chiavi:

$$3, 5, 8, 6, 9, 7, 4$$

Dall'albero così ottenuto si cancelli il nodo con chiave 5 e si indichi l'albero ottenuto. Sia per gli inserimenti che per la cancellazione, motivare sinteticamente il risultato ottenuto.

Domanda A (7 punti) Definire formalmente la classe $O(f(n))$. Dimostrare che la seguente ricorrenza ha soluzione $T(n) = O(n)$

$$T(n) = \frac{1}{3}T(n-1) + 2n + 1$$

Esercizio 1 (9 punti) Realizzare una funzione booleana `checkSum(A,B,C,n)` che dati tre array `A[1..n]`, `B[1..n]` e `C[1..n]` con valori numerici, verifica se esistono tre indici i, j, k con $1 \leq i, j, k \leq n$ tali che $A[i] + B[j] = C[k]$. Valutarne la complessità.