Avvertenza: Si giustifichino tecnicamente tutte le risposte. In caso di discussioni poco formali o approssimative gli esercizi non verranno valutati pienamente.

1. Sia T un albero binario i cui nodi x hanno i campi *left*, *right*, *key*. L'albero si dice **k-limitato**, per un certo valore k, se per ogni nodo x la somma delle chiavi lungo ciascun cammino da x ad una foglia è minore o uguale a k.
   Scrivere una funzione **efficiente in C** *k-limitato(u,k)* che dato in input la radice u di un albero T e un valore k verifica se T è *k-limitato* e ritorna *1* se T è *k-limitato*, *0* altrimenti.
   Valutarne la complessità, **indicando eventuali relazioni di ricorrenza**.

2. Si deve organizzare una gara di programmazione. Ogni programmatore ha un punteggio che esprime la sua abilità (più alto è il punteggio migliore è il programmatore). Ogni programmatore è abbinato a un altro programmatore e la differenza fra i loro punteggi è detta "scarto".
   Scrivere un algoritmo **efficiente** *int scarto(int n, int punteggi[])* che dati n programmatori con n *pari* e i loro *punteggi* restituisce il **minimo scarto totale** (somma degli scarti delle varie coppie) che si può ottenere pianificando in modo ottimale le coppie nella gara. Calcolare e giustificare la complessità dell'algoritmo proposto.
   Si devono scrivere le eventuali procedure/funzioni ausiliarie utilizzate.

3. Si calcoli la complessità asintontica dei seguenti algoritmi (in funzione di n) e si stabilisca quale dei due è preferibile per n sufficiente grande:

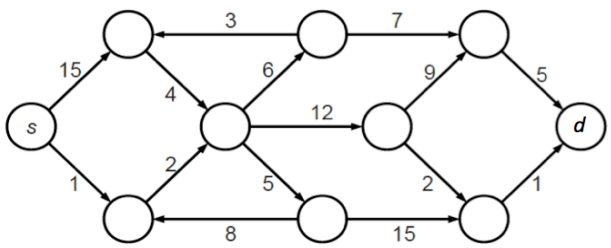
```
MyAlgorithm1( int n )
int
  a, i

if ( n > 1 ) then
   a = 0;
   for i = 1 to n
      a = a + (i+1)*(i+2)
   endfor
   for i = 1 to 4
      a = a + MyAlgorithm1(n/2)
   endfor
   return a
else
   return n-1
endif
```

```
MyAlgorithm2( int n )
int
  a, i, j

if ( n > 1 ) then
   a = 0
   for i = 1 to n
      for j = 1 to n
         a = a + (i+1)*(j+1)
      endfor
   endfor
   for i = 1 to 3
      a = a + MyAlgorithm2(n/2)
   endfor
   return a
else
   return n-1
endif
```

4. La rete ferroviaria italiana può essere descritta mediante un grafo orientato pesato $G = (V, E, w)$, dove i vertici rappresentato le stazioni, la presenza di un arco tra due vertici indica l'esistenza di una tratta ferroviaria diretta tra le corrispondenti stazioni e, per ogni arco $(u,v) \in E$, il peso $w(u,v)$ rappresenta la quantità di carburante necessaria per raggiungere la stazione v partendo da u. Si scriva un algoritmo che, dati in ingresso il grafo G, la quantità C di carburante inizialmente presente nel serbatoio della locomotiva di un treno, e due nodi s e d, restituisca TRUE se esiste un cammino che consente al treno di raggiungere la stazione d partendo dalla stazione s, e FALSE in caso contrario. Si discuta della correttezza e della complessità computazionale dell'algoritmo proposto e si simuli accuratamente la sua esecuzione sul seguente grafo, con $C = 18$.



Nota: nel caso in cui si utilizzi un algoritmo noto, si dimostri la correttezza dell'algoritmo in questione.