Algoritmi e Strutture Dati

a.a. 2023/24

Prima prova intermedia del 08/01/2024

Cognome: _____ Nome: _____

Matricola: _____ E-mail: _____

1. Si consideri un albero ternario completo in cui ogni nodo ha i seguenti campi: (i) `key` chiave intera, (ii) `fruitful` valore booleano, (iii) `left` puntatore al figlio sinistro, (iv) `center` puntatore al figlio centrale, (v) `right` puntatore al figlio destro.
   a. Si scriva una procedura **efficiente** in C o C++ che assegni `True` al campo `fruitful` del nodo se e solo se la somma delle chiavi dei nodi di **ciascuno** dei sottoalberi radicati nei figli è maggiore di una costante `k` fornita in input. Il prototipo della procedura è:
      `void set_fruitful(PTNode r, int k)`
   b. Valutare la complessità della procedura, **indicando eventuali relazioni di ricorrenza e mostrando la loro risoluzione.**
   c. Specificare il linguaggio di programmazione scelto.

2. Questa settimana Carlotta ha ricevuto del denaro dai suoi genitori e vuole spenderlo tutto acquistando libri. Per finire un libro Carlotta impiega una settimana e poiché riceve denaro ogni due settimane, ha deciso di acquistare due libri, così potrà leggerli fino a quando riceverà altri soldi. Desidera spendere tutti i soldi così vorrebbe scegliere due libri i cui prezzi sommati sono pari ai soldi che ha ricevuto.
   Data la quantità di soldi che Carlotta ha a disposizione e un array contenente i prezzi dei libri (tutti distinti), restituire le coppie di prezzi di libri che soddisfano la condizione. Le coppie di prezzi devono contenere prima il prezzo più basso e poi quello più alto.
   a. Scrivere una funzione **efficiente** il cui prototipo è il seguente:
      `int libriSelezionati(array prezzolibri, double soldi, array ris)`

      La funzione restituisce la dimensione dell'array `ris`. **Si devono scrivere eventuali funzioni/procedure ausiliari utilizzate.**
   b. Valutare e giustificare la complessità della funzione.

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 3
      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 7
      a = a + MyAlgorithm2(n/3)
   endfor
   return a
else
   return n-1
endif
```

Si forniscano giustificazioni *formali*. In caso contrario l'esercizio non verrà valutato pienamente, anche in presenza di risposte corrette.