Άσκηση 1

Τα επίπεδα σε ένα δέντρο με ρίζα αριθμούνται με τον ακόλουθο τρόπο:

- το επίπεδο στο οποίο βρίσκεται η ρίζα έχει τον αριθμό 0 και
- αν ένας κόμβος βρίσκεται στο επίπεδο i τότε τα παιδιά του βρίσκονται στο επίπεδο $i + 1$.

Ένα δέντρο με ρίζα λέγεται πλήρες αν όλα τα επίπεδά του είναι γεμάτα, εκτός ίσως από το τελευταίο επίπεδο. Το επίπεδο αυτό μπορεί να είναι γεμάτο από τα αριστερά μέχρι ενός σημείου.

Συγκεκριμένα για τριαδικά δέντρα, το πλήθος των κόμβων που βρίσκονται στο τελευταίο επίπεδο δεν πρέπει οπωσδήποτε να είναι πολλαπλάσιο του 3. Αυτό σημαίνει ότι αν u είναι ο δεξιότερος κόμβος του προτελευταίου επιπέδου που έχει παιδιά, τότε αυτός μπορεί να έχει 1 ή 2 ή 3 παιδιά. Όλοι οι κόμβοι που βρίσκονται στο προτελευταίο επίπεδο αριστερά του u έχουν ακριβώς 3 παιδιά, ενώ όλοι οι κόμβοι που βρίσκονται στο προτελευταίο επίπεδο δεξιά του u είναι φύλλα (δηλαδή δεν έχουν παιδιά).

Άσκηση 6

Ορθότητα αλγορίθμου FACTORIAL

Ας αποδείξουμε την ορθότητα του παρακάτω αλγορίθμου που υπολογίζει το παραγοντικό ενός θετικού ακεραίου.

```
 1: function FACTORIAL(n)
Input: n > 0
 2:     p ← 1
 3:     i ← 2
 4:     while i ≤ n do
 5:         p ← p · i
 6:         i ← i + 1
 7:     end while
 8:     return p
 9: end function
```

Η αναλλοίωτη που θα χρησιμοποιήσουμε για το βρόχο 4-7 είναι η εξής:

$$p = (i-1)! \wedge i \leq n+1 \tag{1}$$

- Αρχικά αποδεικνύουμε ότι η Εξ. 1 ισχύει την πρώτη φορά που ελέγχεται η συνθήκη του **while**. Πραγματικά, τότε ισχύει $p = 1$ και $i = 2$. Άρα προφανώς $p = (i-1)!$ και $i \leq n+1$ (αφού $n > 0$).

- Στη συνέχεια αποδεικνύουμε ότι αν η Εξ. 1 ισχύει πριν από κάποια επανάληψη του **while** loop, τότε ισχύει και μετά από αυτήν την επανάληψη. Έστω ότι $p = (i-1)!$ και $i \leq n+1$ πριν από κάποια επανάληψη. Θα δείξουμε ότι για τις νέες τιμές p' και i' των μεταβλητών p και i αντίστοιχα, ισχύει $p' = (i'-1)!$ και $i' \leq n+1$.

  Στο τέλος της επανάληψης αυτής οι νέες τιμές των μεταβλητών p και i είναι:

$$\begin{aligned} p' &= p \cdot i \\ i' &= i+1 \end{aligned}$$

  Επειδή η εκτέλεση εισήλθε στο **while** loop, έπεται ότι η συνθήκη της γραμμής 4 ισχύει. Άρα:

$$i \leq n \Rightarrow i' \leq n+1$$

  Επίσης, για την τιμή p' έχουμε:

$$p' = p \cdot i = (i-1)! \cdot i = i! = \big(i'-1\big)!$$

  Προκύπτει λοιπόν ότι η Εξ. 1 ισχύει και για τις τιμές των μεταβλητών p, i μετά την επανάληψη.

- Η εκτέλεση του **while** loop ολοκληρώνεται, γιατί όπως είδαμε η τιμή της μεταβλητής i αυξάνεται σε κάθε επανάληψη. Συνεπώς κάποια στιγμή θα ξεπεράσει την τιμή n. Όταν ολοκληρωθεί η εκτέλεση του **while** loop, ισχύουν ταυτόχρονα η αναλλοίωτη του βρόχου και η άρνηση της συνθήκης της γραμμής 4. Άρα έχουμε:

$$p = (i-1)! \tag{2}$$

$$i \leq n+1 \tag{3}$$

$$i > n \tag{4}$$

  Από τις Εξ. 3 και 4 έπεται ότι $i = n+1$. Άρα η Εξ. 2 γράφεται $p = n!$. Συνεπώς η τιμή που επιστρέφει η συνάρτηση είναι όντως το παραγοντικό της εισόδου.

Ορθότητα αλγορίθμου REC_AVERAGE

Θα εξετάσουμε τον ακόλουθο, μάλλον ανορθόδοξο, αλγόριθμο που υπολογίζει το μέσο όρο των τιμών ενός πίνακα ακεραίων με μέγεθος 2^k, $k \in \mathbb{N}$.

Αρχικά παρατηρούμε ότι αν $k > 0$, το **for** loop στις γραμμές 6-8 υπολογίζει έναν νέο πίνακα B με μέγεθος 2^{k-1}, ο οποίος έχει το ίδιο άθροισμα στοιχείων

```
1: function REC_AVERAGE(A)
Input: A: πίνακας μεγέθους 2^k, k ∈ ℕ
2:     var B: πίνακας ακεραίων μεγέθους 2^{k-1}
3:     if length [A] = 1 then
4:         return A(1)
5:     else
6:         for i ← 2 to 2^k step 2 do ▷ η μεταβλητή i αυξάνεται κατά 2 στο τέλος κάθε επανάληψης
7:             B(i/2) ← A(i − 1) + A(i)
8:         end for
9:         return 1/2 · REC_AVERAGE(B)
10:    end if
11: end function
```

με τον A. Προσπαθήστε ως άσκηση να αποδείξετε τον παραπάνω ισχυρισμό χρησιμοποιώντας την ακόλουθη αναλλοίωτη για το βρόχο αυτό[1]:

$$i \text{ άρτιος} \wedge (\forall j : 1 \leq j \leq \frac{i}{2} - 1)\,[B(j) = A(2j-1) + A(2j)]$$

Για να δείξουμε ότι η συνάρτηση REC_AVERAGE είναι ορθή, θα δείξουμε ότι δουλεύει σωστά για κάθε είσοδο μεγέθους 2^k, όπου $k \in \mathbb{N}$. Η απόδειξη θα γίνει με επαγωγή στο k.

- Για $k = 0$, έχουμε ότι το μέγεθος του πίνακα εισόδου A είναι $2^0 = 1$. Σε αυτήν την περίπτωση ικανοποιείται η συνθήκη της γραμμής 3 και η συνάρτηση επιστρέφει το μοναδικό στοιχείο του πίνακα, $A(1)$. Αυτός είναι και ο μέσος όρος των στοιχείων του πίνακα.

- Υποθέτουμε ότι για πίνακα εισόδου με μέγεθος 2^k, $k \leq n$, η συνάρτηση REC_AVERAGE δουλεύει σωστά. Δηλαδή, για πίνακα A με τέτοιο μήκος, ισχύει $\text{REC\_AVERAGE}(A) = \frac{1}{2^k} \cdot \sum_{i=1}^{2^k} A(i)$.

- Έστω ότι η REC_AVERAGE καλείται με είσοδο έναν πίνακα A μεγέθους 2^{n+1}. Τότε ο πίνακας B που υπολογίζεται στο **for** loop έχει μήκος 2^n και ισχύει:

$$\sum_{i=1}^{2^n} B(i) = \sum_{i=1}^{2^{n+1}} A(i)$$

Επιπλέον, λόγω της επαγωγικής υπόθεσης, η τιμή που επιστρέφει η συ-

[1] Θα πρέπει να εξασφαλίσετε ότι η αναλλοίωτη ισχύει κάθε φορά αφότου έχει γίνει η ανάθεση της νέας τιμής της μεταβλητής βρόχου (i), αλλά πριν ελεγχθεί η συνθήκη επανάληψης ($i \leq 2^k$).

νάρτηση ισούται με:

$$\frac{1}{2} \cdot \text{REC\_AVERAGE}(B) = \frac{1}{2} \cdot \frac{1}{2^n} \cdot \sum_{i=1}^{2^n} B(i) = \frac{1}{2^{n+1}} \cdot \sum_{i=1}^{2^{n+1}} A(i)$$

Η τελευταία έκφραση είναι ακριβώς ο μέσος όρος των τιμών του πίνακα A. Άρα η REC_AVERAGE δουλεύει σωστά και για είσοδο μεγέθους 2^{n+1}.