Τμήμα Μηχανικών Πληροφορικής & Υπολογιστών

Μάθημα: Σχεδίαση & Ανάλυση Αλγορίθμων Ιούνιος 2022

Ερωτήματα

Ερώτημα 1 (4)**.**
Στην ακόλουθη σειρά εντολών η μεταβλητή n έχει ακέραια τιμή μεγαλύτερη του μηδενός.

$k \leftarrow 0;$
for $i \leftarrow 1; i \leq n; i++$ **do**
  for $j \leftarrow n; j \geq i; j--$ **do**
    $k++;$
  end_for
end_for

1. Να προσδιορίσετε την τιμή του k μετά το τέλος της εκτέλεσης του παραπάνω κώδικα.
2. Να προσδιορίσετε τον ακριβή αριθμό των Στοιχειωδών Υπολογιστικών Βημάτων που εκτελεί ο παραπάνω κώδικας.
3. Να εκπονήσετε αλγόριθμο τάξης $\Omega(n)$ ο οποίος να υπολογίζει την τιμή του k που υπολογίστηκε από τον δοθέντα κώδικα και να έχει καλύτερη ασυμπτωτική συμπεριφορά από αυτόν. Να αποδείξετε την πολυπλοκότητα του.
4. Υπάρχει ασυμπτωτικά πιο αποτελεσματικός αλγόριθμος από αυτόν που εκπονήσατε στο προηγούμενο υποερώτημα; Να αιτιολογήσετε την απάντηση σας.

Ερώτημα 2 (6)**.** Έστω A τετραγωνικός πίνακας ακεραίων $m \times n$ όπου τα στοιχεία κάθε γραμμής, από αριστερά προς τα δεξιά σχηματίζουν αύξουσα ακολουθία και τα στοιχεία κάθε στήλης από πάνω προς τα κάτω επίσης. Για παράδειγμα, ένας τέτοιος πίνακας για $m = n = 5$ δίνεται παρακάτω

$$A = \begin{bmatrix} 2 & 7 & 12 & 15 & 17 \\ 4 & 9 & 14 & 18 & 19 \\ 5 & 11 & 17 & 19 & 20 \\ 8 & 14 & 22 & 25 & 32 \\ 10 & 16 & 24 & 30 & 35 \end{bmatrix}$$

Σε έναν τέτοιο πίνακα μας ενδιαφέρει να βρούμε αν υπάρχει ένας δεδομένος ακέραιος a.

1. Να εκπονήσετε αναδρομικό αλγόριθμο πολυπλοκότητας $O(m+n)$ που να επιλύει το παραπάνω πρόβλημα. Πριν την περιγραφή του αλγόριθμου σε ψευδοκώδικα να παρουσιάσετε αναλυτικά την ιδέα που υλοποιεί ο αλγόριθμός σας.
   Υπόδειξη: Αρχικώς να συγκρίνετε το προς αναζήτηση στοιχείο με το «κάτω-αριστερά» στοιχείο του πίνακα.
2. Να εκτελέσετε τον αλγόριθμο που εκπονήσατε αναζητώντας τον αριθμό 15 : σε κάθε αναδρομική κλίση να απεικονίσετε τις τιμές των μεταβλητών και των παραμέτρων του αλγόριθμου σας.
3. Να γράψετε την αναδρομική σχέση που περιγράφει τον αριθμό των ΣΥΒ και να αποδείξετε την πολυπλοκότητα του αλγόριθμου.

Ενδεικτικές Απαντήσεις

Ερώτημα 1.

Απάντηση.

1. Η τιμή της μεταβλητής k μετά την εκτέλεση του κώδικα δίνεται από

$$\begin{aligned} k &= \sum_{i=1}^{n}\sum_{j=i}^{n} 1 = \sum_{i=1}^{n}(n-i+1) \\ &= n^2 - \frac{n(n+1)}{2} + n = n(n+1-\frac{n+1}{2}) \\ &= \frac{n(n+1)}{2}. \end{aligned}$$

Exists_a_in_A$(15, A, 5, 1)$	$A[5,1] = 10 < 15$	
Exists_a_in_A$(15, A, 5, 2)$	$A[5,2] = 16 > 15$	
Exists_a_in_A$(15, A, 4, 2)$	$A[4,2] = 14 < 15$	
Exists_a_in_A$(15, A, 4, 3)$	$A[4,3] = 22 > 15$	
Exists_a_in_A$(15, A, 3, 3)$	$A[3,3] = 17 > 15$	
Exists_a_in_A$(15, A, 2, 3)$	$A[2,3] = 14 < 15$	
Exists_a_in_A$(15, A, 2, 4)$	$A[2,4] = 18 > 15$	
Exists_a_in_A$(15, A, 1, 4)$	$A[1,4] = 15 = 15$	

Πίνακας 1: Διαδοχικές αναδρομικές κλήσεις του Αλγόριθμου 1

2. Ο αριθμός των επαναλήψεων ου εσωτερικού βρόχου είναι $n-i+1$. Σε κάθε επανάληψη κάνει δύο ΣΥΒ σε σχέση με την μεταβλητή k και τρία σε σχέση με τη μεταβλητή j. Επίσης εκτελείται ένα ΣΥΒ για την αρχικοποίηση της μεταβλητής j και μία σύγκριση κατά την οποία η μεταβλητή j έχει τιμή μικρότερη από αυτή της μεταβλητής i. Ο συνολικός αριθμός των ΣΥΒ που γίνονται στον εσωτερικό βρόχο είναι
$$A(i) = 5(n-i+1) + 2.$$
Άρα ο συνολικός αριθμός των ΣΥΒ που εκτελεί ο αλγόριθμος είναι
$$\begin{aligned} T(n) &= \sum_{i=1}^{n} A(i) + 3n + 2 + 1 \\ &= 5\sum_{i=1}^{n}(n-i+1) + \sum_{i=1}^{n} 2 + 3(n-1) + 3 = 5\frac{n(n+1)}{2} + 2n + 3n + 3 \\ &= 5\frac{n^2}{2} + 15\frac{n}{2} + 3. \end{aligned}$$

3. Ο ζητούμενος αλγόριθμος παρουσιάζεται στη συνέχεια
 $k \leftarrow 0;$
 for $i \leftarrow 1; i \leq n; i++$ **do**
 $k \leftarrow k + i;$
 end_for
 Ο αριθμός των ΣΥΒ που εκτελεί ο αλγόριθμος αυτός είναι
$$T(n) = 3n + 2 + 2n + 1 = 5n + 3.$$
Επομένως ο αλγόριθμος αυτός είναι τάξης $\Theta(n)$ και είναι χαμηλότερης τάξης από τον δοθέντα.

4. Η τιμή του k μπορεί να υπολογιστεί σε $\Theta(1)$ βήματα από την εντολή
$$k \leftarrow (n \cdot (n+1))/2.$$

■

Ερώτημα 2. *Απάντηση.*

1. Για να εκμεταλλευτούμε τη διάταξη που υπάρχει στον πίνακα θα χρησιμοποιήσουμε δύο δείκτες i, j· ο δείκτης i διατρέχει τις γραμμές και ο δείκτης j τις στήλες. Ξεκινώντας από το στοιχείο $A[i=m, j=1]$ (κάτω αριστερά στοιχείο) αν το στοιχείο $A[i,j] < a$ τότε αυξάνουμε κατά ένα το δείκτη j ενώ αν $A[i,j] > a$ τότε μειώνουμε τον δείκτη i κατά ένα.

 Ο Αλγόριθμος 1 καλείται αρχικά με παραμέτρους $i = m, j = 1$.

2. Στον Πίνακα 1 παρουσιάζονται οι αναδρομικές κλήσεις του Αλγόριθμου 1 όταν αναζητείται το στοιχείο 15.

3. Παρατηρούμε ότι σε κάθε κλήση ο αριθμός των ΣΥΒ φράζεται από τα επάνω από μία σταθερά έστω c. Επομένως ο αριθμός των ΣΥΒ φράζεται από τα επάνω από τη συνάρτηση
$$T(i,j) = \begin{cases} \max\{T(i-1,j), T(i,j+1)\} + c, & m \geq i > 1, 1 \leq j < n, \\ c, & i = 1 \text{ ή } j = n. \end{cases} \tag{1}$$
Είναι εύκολο να επιλυθεί η παραπάνω αναδρομική σχέση. Θεωρώντας ότι το προς-αναζήτηση στοιχείο είναι το $A[1,n]$, έχουμε τον μεγαλύτερο αριθμό αναδρομικών κλήσεων. Στην περίπτωση αυτή έχουμε μία κλίση για κάθε γραμμή και κάθε στήλη του πίνακα και επομένως ο αριθμός των ΣΥΒ είναι τάξης $O(m+n)$.

■

Αλγόριθμος 1 Εύρεση στοιχείου a σε πίνακα A.

Απαιτείται: Πίνακας ακεραίων A με m γραμμές και n στήλες όπου τα στοιχεία της κάθε γραμμής και στήλης είναι ταξινομημένα σε αύξουσα σειρά, ακέραιος a, ακέραιος i, ακέραιος j.

Επιστρέφεται: Επιστροφή τιμής `True` αν υπάρχει ο ακέραιος a, `False` διαφορετικά.

```
function Exists_a_in_A(int a, int A[], int i, int j)
    m ← nRows(A);
    n ← nCols(A);
    if A[i, j] = a then
        return True;
    end if
    if A[i, j] < a then
        if j < n then
            return Exists_a_in_A(a, A, i, j + 1);
        else
            return False;
        end if
    end if
    if A[i, j] > a then
        if i < 1 then
            return Exists_a_in_A(a, A, i − 1, j);
        else
            return False;
        end if
    end if
end function
```