```
ΠΡΟΓΡΑΜΜΑ BubbleSort_and_BinarySearch
! Να γραφεί πρόγραμμα σε ΓΛΩΣΣΑ το οποίο διαβάζει έναν πίνακα (μονοδιάστατο)
! Ν πραγματικών αριθμών (το Ν να δηλωθεί ως σταθερά) από το πληκτρολόγιο.
! Στη συνέχεια ταξινομεί τον πίνακα σε αύξουσα σειρά (δηλαδή από το μικρότερο
! προς το μεγαλύτερο) με χρήση του αλγόριθμου BubbleSort.
! Έπειτα δείχνει τον (ταξινομημένο πλέον) πίνακα στην οθόνη.
! Στην συνέχεια, με χρήση του αλγόριθμου δυαδικής αναζήτησης (binary search)
! αναζητά εάν υπάρχει στον πίνακα ένα στοιχείο key το οποίο δίνει ο χρήστης
! από το πληκτρολόγιο και εάν το βρει (στην πρώτη του εμφάνιση) εμφανίζει
! σχετικό μήνυμα επιτυχίας που περιλαμβάνει και τη θέση του πίνακα στην οποία
! βρέθηκε, αλλιώς εμφανίζει σχετικό μήνυμα αποτυχίας.
ΣΤΑΘΕΡΕΣ
  ! Ν : το πλήθος των στοιχείων του πίνακα
  N = 5
ΜΕΤΑΒΛΗΤΕΣ
  ! table : πίνακας (μονοδιάστατος) με Ν πραγματικούς αριθμούς
  ! temp: μεταβλητή για την "προσωρινή" αποθήκευση κατά την
  !       αντιμετάθεση στοιχείων
  ! i και j : μετρητές των αντίστοιχων ΓΙΑ
  ! left, right: τα όρια μέσα στα οποία γίνεται η αναζήτηση
  ! position: η θέση στην οποία βρέθηκε το στοιχείο, αλλιώς μηδέν(0)
  ! isFound : λογική μεταβλητή που δείχνει εάν βρέθηκε το key στον table
  ! m : η μέση του διαστήματος (left, right) ως: (left+right) div 2
  ΠΡΑΓΜΑΤΙΚΕΣ: table[N], temp, key
  ΑΚΕΡΑΙΕΣ: i, j, position, left, right, m
  ΛΟΓΙΚΕΣ: isFound
ΑΡΧΗ
  ! Διαβάζουμε από το πληκτρολόγιο Ν πραγματικούς αριθμούς
  ! με τους οποίους αποθηκεύουμε στον πίνακα table
  ΓΙΑ i ΑΠΟ 1 ΜΕΧΡΙ N
    ΓΡΑΨΕ 'Παρακαλώ πληκτρολογήστε το ', i, 'ο στοιχείο του πίνακα table και πατήστε enter:'
    ΔΙΑΒΑΣΕ table[i]
  ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
  ! Ταξινόμηση BubbleSort
  ! Η εξωτερική ΓΙΑ μετά από κάθε βήμα φέρνει και ένα στοιχείο
  ! "προς τα πάνω" (δηλαδή πρώτα το μικρότερο από όλα κ.ο.κ.) στην
  ! οριστική του θέση.
  ! Η εσωτερική ΓΙΑ σε κάθε επανάληψη ελέγχει ζευγάρια στοιχείων (ξεκινώντας από
  ! το τέλος του πίνακα) και εάν είναι σε λάθος διάταξη τα αντιμεταθέτει.
  ΓΙΑ i ΑΠΟ 2 ΜΕΧΡΙ N
    ΓΙΑ j ΑΠΟ N ΜΕΧΡΙ i ΜΕ ΒΗΜΑ -1
      ΑΝ table[j-1] > table[j] ΤΟΤΕ
        ! Οι γραμμές κώδικα 49, 50, και 51
        ! αντιμεταθέτουν τα στοιχεία table[j-1] και table[j]:
        ! πρώτα αποθηκεύουμε την τιμή του table[j-1] στην μεταβλητή temp,
        ! μετά βάζουμε την τιμή του table[j] στην μεταβλητή table[j-1],
        ! και τέλος βάζουμε την ΠΑΛΙΑ τιμή του table[j-1] (δηλαδή το temp)
        ! στην μεταβλητή table[j]
        temp <-- table[j-1]
        table[j-1] <-- table[j]
        table[j] <-- temp
      ΤΕΛΟΣ_ΑΝ
    ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
  ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
  ! Εμφάνιση ταξινομημένου πίνακα:
  ΓΡΑΨΕ 'Ο ταξινομημένος πίνακας είναι: (αύξουσα ταξινόμηση)'
  ΓΙΑ i ΑΠΟ 1 ΜΕΧΡΙ N
    ΓΡΑΨΕ 'Το ', i, 'ο στοιχείο του πίνακα table είναι:', table[i]
  ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
  ! Αλγόριθμος αναζήτησης binary search
  ! (εφαρμόζεται σε ήδη ταξινομημένο πίνακα!)
  ΓΡΑΨΕ 'Παρακαλώ δώστε τον αριθμό που θέλετε να ψάξω εάν υπάρχει στον πίνακα:'
  ΔΙΑΒΑΣΕ key
  ! αρχικά η αναζήτηση γίνεται σε ολόκληρο τον πίνακα
  left <-- 1
  right <-- N
  ! αρχικοποίηση του position με μια τιμή εκτός των 1,2,...,N
  position <-- 0
  ! ακόμη δεν έχουμε βρει το key στον table άρα:
  isFound <-- ΨΕΥΔΗΣ
  ! Όσο υπάρχει διάστημα αναζήτησης με τουλάχιστον ένα στοιχείο
  ! και δεν έχει βρεθεί το στοιχείο που αναζητούμε
  ΟΣΟ ( Left <= Right ) ΚΑΙ ( isFound = ΨΕΥΔΗΣ ) ΕΠΑΝΑΛΑΒΕ
    ! βρίσκω τη μεσαία θέση του διαστήματος αναζήτησης
    m <-- ( left + right ) div 2
    ! Αν το στοιχείο που ψάχνω βρίσκεται στη μέση του
    ! διαστήματος αναζήτησης
    ΑΝ table[m] = key ΤΟΤΕ
```

```
79.         ! αποθηκεύω τη θέση που το βρήκα
80.         position <-- m
81.         ! ενημερώνω ότι βρήκα αυτό που έψαχνα
82.         isFound <-- ΑΛΗΘΗΣ
83.       ΑΛΛΙΩΣ
84.         ! αν το στοιχείο στη μέση του διαστήματος
85.         ! αναζήτησης είναι μικρότερο αυτού που ψάχνω
86.         ΑΝ table[m] < key  ΤΟΤΕ
87.           ! συνεχίζω την αναζήτηση στο δεξί μισό
88.           left <-- m+1
89.         ΑΛΛΙΩΣ
90.           ! συνεχίζω την αναζήτηση στο αριστερό μισό
91.           right <-- m-1
92.         ΤΕΛΟΣ_ΑΝ
93.       ΤΕΛΟΣ_ΑΝ
94.     ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
95.     ΑΝ isFound = ΑΛΗΘΗΣ  ΤΟΤΕ
96.       ΓΡΑΨΕ 'Το στοιχείο ', key, ' βρέθηκε στη θέση ', position, ' του πίνακα table.'
97.       ΓΡΑΨΕ 'Σημείωση: το στοιχείο μπορεί να υπήρχε και σε άλλες θέσεις στον πίνακα!'
98.     ΑΛΛΙΩΣ
99.       ΓΡΑΨΕ 'Το στοιχείο ', key, ' δεν βρέθηκε στον πίνακα table.'
100.    ΤΕΛΟΣ_ΑΝ
101.    ! η χρήση της μεταβλητής isFound δεν είναι απαραίτητη:
102.    ! μπορεί στη θέση της να χρησιμοποιηθεί η μεταβλητή position,
103.    ! όπου το αντίστοιχο της τιμής ΨΕΥΔΗΣ για την μεταβλητή isFound
104.    ! είναι η τιμή μηδέν(0) της μεταβλητής position,
105.    ! ενώ το αντίστοιχο της τιμής ΑΛΗΘΗΣ για την μεταβλητή isFound
106.    ! είναι οποιαδήποτε τιμή διάφορη του μηδέν(0) της μεταβλητής position,
107.    ! (η οποία προφανώς θα είναι κάποια των 1,2,...,N)
108.  ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```