

Ταξινόμηση (Sort) και Αναζήτηση (Search)

Εισαγωγή στον Προγραμματισμό

(CEID_NY131)

A Step-By-Step Approach
με τη χρήση παραδειγμάτων

Kleanthis Thramboulidis
Prof. of Software and System Engineering
University of Patras
<https://sites.google.com/site/thramboulidiskleanthis/>



Οργάνωση Διάλεξης – Ταξινόμηση στην i2p library

■ Ταξινόμηση με χρήση της i2p βιβλιοθήκης

- Πίνακα ακεραίων
- Πίνακα αλφαριθμητικών

■ Ταξινόμηση - Bubble Sort

- A step-by-step Approach για πίνακα ακεραίων
- Εναλλακτικές Υλοποιήσεις (+Αναδρομικότητα)
- Ταξινόμηση πίνακα Δεικτών σε ακεραίους
- Ταξινόμηση Λέξεων
 - Ταξινόμηση πίνακα αλφαριθμητικών
 - Ταξινόμηση πίνακα Δεικτών σε αλφαριθμητικά

■ Αναζήτηση

- Linear
- Binary
- Σε πίνακα ακεραίων
- Σε πίνακα αλφαριθμητικών

Ταξινόμηση με χρήση της i2p βιβλιοθήκης

1. Ταξινόμηση (αύξουσα και φθίνουσα) **Πίνακα ακεραίων**

i2p library

```
void sortIntArrayInc(int ar[],int numElements);  
void sortIntArrayDec(int ar[],int numElements);
```

2. Ταξινόμηση (αύξουσα και φθίνουσα) **Πίνακα αλφαριθμητικών**

```
void sortStringArrayInc(char base[],int numElements, int strWidth);  
void sortStringArrayDec(char base[],int numElements, int strWidth);
```

Οδηγίες αξιοποίησης της βιβλιοθήκης i2p και παραδείγματα κλήσης των συναρτήσεων ταξινόμησης θα βρείτε στο αρχείο

[i2pLibrary.pdf](#)

Ταξινόμηση Πίνακα ακεραίων

```
void sortIntArrayInc(int ar[],int numElements);  
void sortIntArrayDec(int ar[],int numElements);
```

```
C:\Code\courses\I2P2023-24_ x + v  
Array before sorting  
9 4 12 2 17 21 8 10 3  
Array status: Not sorted  
Press any key to continue . . .  
Array after incremental sort  
2 3 4 8 9 10 12 17 21  
Array status: Incrementally sorted  
Press any key to continue . . .  
Array after decremental sort  
21 17 12 10 9 8 4 3 2  
Array status: Decrementally sorted  
-----  
Process exited after 4.325 seconds with return value 0  
Press any key to continue . . .
```

Αναπτύξτε ένα πρόγραμμα που να έχει την παραπλεύρως έξοδο κάνοντας χρήση των δύο παραπάνω συναρτήσεων ταξινόμησης.

Η διαπίστωση του αν είναι ταξινομημένος ο πίνακας και πως, να γίνεται από την μηχανή με την εκτέλεση κατάλληλης συνάρτησης

Ταξινόμηση Πίνακα αλφαριθμητικών

```
void sortStringArrayInc(char base[],int numElements, int strWidth);  
void sortStringArrayDec(char base[],int numElements, int strWidth);
```

```
char ar[][10]={ "paris", "nikos", "alekos", "kostas",  
               "basos", "andreas", "petros"};
```

Χρησιμοποιήστε τις δύο παραπάνω συναρτήσεις για να αναπτύξετε ένα πρόγραμμα που να έχει έξοδο ανάλογη με αυτή του προγράμματος που αναπτύξατε για ταξινόμηση ακεραίων.

Η διαπίστωση του αν είναι ταξινομημένος ο πίνακας και πως, να γίνεται από την μηχανή με την εκτέλεση κατάλληλης συνάρτησης

Οργάνωση Διάλεξης - Ταξινόμηση - Bubble Sort

- Ταξινόμηση με χρήση της i2p βιβλιοθήκης
- **Ταξινόμηση - Bubble Sort**
 - A step-by-step Approach για πίνακα ακεραίων
 - Ενναλακτικές Υλοποιήσεις (+Αναδρομικότητα)
 - Ταξινόμηση πίνακα Δεικτών σε ακεραίους
 - Ταξινόμηση Λέξεων
 - Ταξινόμηση πίνακα αλφαριθμητικών
 - Ταξινόμηση πίνακα Δεικτών σε αλφαριθμητικά
- **Αναζήτηση (Search)**
 - Linear Search
 - Binary Search
 - Σε πίνακα ακεραίων
 - Σε πίνακα αλφαριθμητικών

Bubble Sort - Ορισμός

```
int ar[] = {3,5,4,7,2,8,1,0,9,6};
```

"Bubble sort (...) is a simple sorting algorithm that repeatedly steps through the input list element by element, comparing the current element with the one after it, swapping their values if needed. These passes through the list are repeated until no swaps had to be performed during a pass, meaning that the list has become fully sorted. The algorithm, which is a comparison sort, is named for the way the larger elements "bubble" up to the top of the list." *Wikipedia*

Το Πρόγραμμα με την bubbleSort()

Αναπτύξτε ένα πρόγραμμα που θα χρησιμοποιεί μια συνάρτηση που υλοποιεί τον αλγόριθμο Bubble Sort για να ταξινομήσει ένα πίνακα ακεραίων. Το πρόγραμμα να έχει την παρακάτω έξοδο στην οθόνη.

```
C:\Code\courses\I2P2023-24_ × + v
3      5      4      7      2      8      1      0      9      6
Array status: not sorted
Press any key to continue . . .
0      1      2      3      4      5      6      7      8      9
Array status: sorted

-----
Process exited after 2.028 seconds with return value 0
Press any key to continue . . .
```

Screenshot εκτέλεσης του
ζητούμενου προγράμματος.

bubbleSort(int ar[], int numElements)

```
16 int ar[] = {3,5,4,7,2,8,1,0,9,6};
17 int main(int argc, char *argv[]) {
18
19     int numElements = sizeof(ar)/sizeof(int);
20     displayArray(ar, numElements);
21     arrayStatus(ar, numElements);
22     system("pause");
23     bubbleSort(ar, numElements);
24     displayArray(ar, numElements);
25     arrayStatus(ar, numElements);
26     return 0;
27 }
```

Η bubbleSort() ταξινομεί ένα πίνακα ακεραίων ar με αριθμό στοιχείων numElements.

```
3      5      4      7      2      8      1      0      9      6
Array status: not sorted
Press any key to continue . . .
0      1      2      3      4      5      6      7      8      9
Array status: sorted
-----
Process exited after 2.028 seconds with return value 0
Press any key to continue . . .
```

Screenshot εκτέλεσης του παραπάνω προγράμματος.

Κλεάνθης Θραμπουλίδης

τάξινομηση - αναζήτηση

9

Bubble Sort – Βασικές διεργασίες

```
int ar[] = {3,5,4,7,2,8,1,0,9,6};
```

"Bubble sort (...) is a simple sorting algorithm that repeatedly steps through the input list element by element, comparing the current element with the one after it, swapping their values if needed. These passes through the list are repeated until no swaps had to be performed during a pass, meaning that the list has become fully sorted. The algorithm, which is a comparison sort, is named for the way the larger elements "bubble" up to the top of the list."

Wikipedia

- Δύο είναι οι βασικές διεργασίες που εκτελεί
 - Σύγκριση δύο στοιχείων (comparing the current element with the one after it)
 - Εναλλαγή θέσης αν η σύγκριση το απαιτεί (swapping their values if needed)
- Με την επαναλαμβανόμενη εκτέλεση των δύο αυτών διεργασιών επιτυγχάνει την ταξινόμηση των στοιχείων του πίνακα
 - repeatedly steps through the input list element by element
 - These passes through the list are repeated

Κλεάνθης Θραμπουλίδης

Ταξινόμηση - Αναζήτηση

10

Bubble Sort – A step-by-step Approach

"Bubble sort (...) is a simple sorting algorithm that repeatedly steps through the input list element by element, comparing the current element with the one after it, swapping their values if needed. These passes through the list are repeated until no swaps had to be performed during a pass, meaning that the list has become fully sorted. The algorithm, which is a comparison sort, is named for the way the larger elements "bubble" up to the top of the list." Wikipedia

```
int ar[] = {3,5,4,7,2,8,1,0,9,6};
```

- Στόχος μας να αναπτύξουμε την bubbleSort βήμα-προς-βήμα.
 - **Step 1 – Υλοποιεί το κίτρινο μέρος της περιγραφής**
Κάνει moveBubbleUp για τον πίνακα ar με στοιχεία numElements
 - **Step 2 – Υλοποιεί την επανάληψη του Step 1 για όλους τους υπο-πίνακες του ar μέχρι τον υπο-πίνακα με 2 στοιχεία (πράσινο μέρος περιγραφής).**
Στην ουσία επαναλαμβάνει την δουλειά του βήματος 1, δηλαδή το moveBubbleUp, για κάθε υπο-πίνακα του ar μέχρι αυτόν με 2 στοιχεία.
 - **Step 3 – Βελτίωση του βήματος 2 για να αποφύγουμε τις άσκοπες επαναλήψεις**

1st Step – moveBubbleUp for ar (1/2)

"Bubble sort (...) is a simple sorting algorithm that repeatedly steps through the input list element by element, comparing the current element with the one after it, swapping their values if needed. (...) " Wikipedia

- Ονομάζουμε **moveBubbleUp** τη διεργασία που περιγράφεται παραπάνω με κίτρινο
- Αυτή είναι η πρώτη δουλειά που πρέπει να κάνει η bubbleSort
- Με βάση αυτό η bubbleSort διαμορφώνεται όπως παρακάτω

```
28 void bubbleSort(int ar[], int numElements){  
29     moveBubbleUp(ar, numElements);  
30 }  
31  
32 void moveBubbleUp(int ar[], int numElements){
```

Δώστε το σώμα της moveBubbleUp

1st Step – moveBubbleUp for ar (2/2)

Ο πίνακας

```
int ar[] = {3,5,4,7,2,8,1,0,9,6};
```

Screenshot εκτέλεσης 1ης έκδοσης

C:\Code\courses\I2P2023-24_ X + v

3 5 4 7 2 8 1 0 9 6

Array status: not sorted

Press any key to continue . . .

Iteration No2->3 4 5 7 2 8 1 0 9 6

Iteration No4->3 4 5 2 7 8 1 0 9 6

Iteration No6->3 4 5 2 7 1 8 0 9 6

Iteration No7->3 4 5 2 7 1 0 8 9 6

Iteration No9->3 4 5 2 7 1 0 8 6 9

End of array pass

Press any key to continue . . .

3 4 5 2 7 1 0 8 6 9

Array status: not sorted

Process exited after 15.26 seconds with return value 0

Press any key to continue . . .

To iteration είναι η επανάληψη μιας εκτέλεσης της σύγκρισης δυο στοιχείων και εναλλαγής θέσης τους αν απαιτείται.

Δώστε τον πηγαίο κώδικα για το βήμα αυτό (V1)

Next Step for bubbleSort ?

Ο πίνακας πριν την εκτέλεση του Step 1

C:\Code\courses\I2P2023-24_ X + v

3 5 4 7 2 8 1 0 9 6

Array status: not sorted

Ο πίνακας μετά την εκτέλεση του Step 1 (moveBubbleUp)

3 4 5 2 7 1 0 8 6 9

Array status: not sorted

Next step

3 4 2 5 1 0 7 6 8

Next step

3 2 4 1 0 5 6 7

Πόσα steps;

2nd Step – MoveBubbleUp for all sub-arrays

Screenshot εκτέλεσης 2ης έκδοσης (1st part)

```
C:\Code\courses\I2P2023-24_ x + v
3      5      4      7      2      8      1      0      9      6
Array status: not sorted
Press any key to continue . . .
Iteration No2->3      4      5      7      2      8      1      0      9      6
Iteration No4->3      4      5      2      7      8      1      0      9      6
Iteration No6->3      4      5      2      7      1      8      0      9      6
Iteration No7->3      4      5      2      7      1      0      8      9      6
Iteration No9->3      4      5      2      7      1      0      8      6      9
End of array pass for 10 elements
Press any key to continue . . .
Iteration No3->3      4      2      5      7      1      0      8      6
Iteration No5->3      4      2      5      1      7      0      8      6
Iteration No6->3      4      2      5      1      0      7      8      6
Iteration No8->3      4      2      5      1      0      7      6      8
End of array pass for 9 elements
Press any key to continue . . .
Iteration No2->3      2      4      5      1      0      7      6
Iteration No4->3      2      4      1      5      0      7      6
Iteration No5->3      2      4      1      0      5      7      6
Iteration No7->3      2      4      1      0      5      6      7
End of array pass for 8 elements
```

Αναβαθμίστε τον πηγαίο κώδικα του βήματος 1 (V1) για να υλοποιήσει την εκτέλεση της λειτουργικότητας `μονεBubbleUp` για όλους τους υπο-πίνακες του `ar` (V2).

2nd Step – MoveBubbleUp for all sub-arrays

Screenshot εκτέλεσης 2ης έκδοσης (last part)

```
Iteration No1->1      2      0      3      4
Iteration No2->1      0      2      3      4
End of array pass for 5 elements
Press any key to continue . . .
Iteration No1->0      0      1      2      3
End of array pass for 4 elements
Press any key to continue . . .
End of array pass for 3 elements
Press any key to continue . . .
End of array pass for 2 elements
Press any key to continue . . .
0      1      2      3      4      5      6      7      8      9
Array status: sorted

-----
Process exited after 17.62 seconds with return value 0
Press any key to continue . . .
```

Παρατηρήστε τις άσκοπες επαναλήψεις για 3 και 2 στοιχεία

3rd Step – Avoid meaningless iterations

Screenshot εκτέλεσης 3^{ης} έκδοσης (last part)

```
Press any key to continue . . .
Iteration No2->2      1      3      0      4      5
Iteration No3->2      1      0      3      4      5

End of array pass for 6 elements

Press any key to continue . . .
Iteration No1->1      2      0      3      4
Iteration No2->1      0      2      3      4

End of array pass for 5 elements

Press any key to continue . . .
Iteration No1->0      1      2      3

End of array pass for 4 elements

Press any key to continue . . .

End of array pass for 3 elements

Press any key to continue . . .
0      1      2      3      4      5      6      7      8      9
Array status: sorted

-----
Process exited after 12.04 seconds with return value 0
Press any key to continue . . . |
```

Αναβαθμίστε τον πηγαίο κώδικα του βήματος 2 (V2) για να αποφύγετε την εκτέλεση της λειτουργικότητας `moveBubbleUp` για τους ταξινομημένους υπο-πίνακες του `ar` (V3).

Εναλλακτικές Υλοποιήσεις

1. Ένας βρόχος και συνάρτηση `moveBubbleUp`
2. Δύο βρόχοι (χωρίς τη συνάρτηση `moveBubbleUp`)
3. Αναδρομικότητα (recursion)
Η `bubbleSort` καλεί τον εαυτό της.
4. Ταξινόμηση πίνακα δεικτών σε ακραίους ?

```
int *ar[100];
```

5. Ταξινόμηση Αλφαριθμητικών ?

```
14 char ar[][10] = {"cde", "abc", "cef", "abb", "aab",
15                  "klm", "kab", "gbc", "gab", "aaa"};

char *ar[] = {"cde", "abc", "cef", "abb", "aab",
              "klm", "kab", "gbc", "gab", "aaa"};
```

Ταξινόμηση Πίνακα Δεικτών σε int

```
11 int arr[] = {3,5,4,7,2,8,1,0,9,6}; //
12 //int arr[] = {1,0,4,7,2,8,5,3,9,6}; //4
13
14 int main(int argc, char *argv[]) {
15     int numOfElements;
16
17     numOfElements=sizeof(arr)/sizeof(int);
18
19     int *ar[numOfElements];
20     for(int i;i<numOfElements;i++)
21         ar[i]=&arr[i];
```

Δώστε τον πηγαίο κώδικα
τροποποιώντας τον κώδικα της
bubbleSort για πίνακα ακεραίων

Ταξινόμηση Πίνακα Αλφαριθμητικών

```
char ar[][10] = {"cde", "abc", "cef", "abb", "aab",  
                "klm", "kab", "gbc", "gab", "aaa"};
```

```
C:\Code\courses\I2P2023-24_ X + v
cde  abc  cef  abb  aab  klm  kab  gbc  gab  aaa
Array status: not sorted
Press any key to continue . . .
Iteration No1->abc  cde  cef  abb  aab  klm  kab  gbc  gab  aaa
Iteration No3->abc  cde  abb  cef  aab  klm  kab  gbc  gab  aaa
Iteration No4->abc  cde  abb  aab  cef  klm  kab  gbc  gab  aaa
Iteration No6->abc  cde  abb  aab  cef  kab  klm  gbc  gab  aaa
Iteration No7->abc  cde  abb  aab  cef  kab  gbc  klm  gab  aaa
Iteration No8->abc  cde  abb  aab  cef  kab  gbc  gab  klm  aaa
Iteration No9->abc  cde  abb  aab  cef  kab  gbc  gab  aaa  klm
End of array pass for 10 Elements. Array is not sorted

Press any key to continue . . .
Iteration No2->abc  abb  cde  aab  cef  kab  gbc  gab  aaa
Iteration No3->abc  abb  aab  cde  cef  kab  gbc  gab  aaa
Iteration No6->abc  abb  aab  cde  cef  gbc  kab  gab  aaa
Iteration No7->abc  abb  aab  cde  cef  gbc  gab  kab  aaa
Iteration No8->abc  abb  aab  cde  cef  gbc  gab  aaa  kab
End of array pass for 9 Elements. Array is not sorted

Press any key to continue . . .
```

Δώστε τον πηγαίο κώδικα

Ταξινόμηση Πίνακα * σε Αλφαριθμητικά

```
char *ar[] = {"cde", "abc", "cef", "abb", "aab",  
              "klm", "kab", "gbc", "gab", "aaa"};
```

```
Array status: not sorted  
Press any key to continue . . .  
Iteration No1->abc    cde    cef    abb    aab    klm    kab    gbc    gab    aaa  
Iteration No3->abc    cde    abb    cef    aab    klm    kab    gbc    gab    aaa  
Iteration No4->abc    cde    abb    aab    cef    klm    kab    gbc    gab    aaa  
Iteration No6->abc    cde    abb    aab    cef    kab    klm    gbc    gab    aaa  
Iteration No7->abc    cde    abb    aab    cef    kab    gbc    klm    gab    aaa  
Iteration No8->abc    cde    abb    aab    cef    kab    gbc    gab    klm    aaa  
Iteration No9->abc    cde    abb    aab    cef    kab    gbc    gab    aaa    klm  
End of array pass for 10 Elements. Array is not sorted  
  
Press any key to continue . . .  
Iteration No2->abc    abb    cde    aab    cef    kab    gbc    gab    aaa  
Iteration No3->abc    abb    aab    cde    cef    kab    gbc    gab    aaa  
Iteration No6->abc    abb    aab    cde    cef    gbc    kab    gab    aaa  
Iteration No7->abc    abb    aab    cde    cef    gbc    gab    kab    aaa  
Iteration No8->abc    abb    aab    cde    cef    gbc    gab    aaa    kab  
End of array pass for 9 Elements. Array is not sorted  
  
Press any key to continue . . .  
Iteration No1->abb    abc    aab    cde    cef    gbc    gab    aaa  
Iteration No2->abb    aab    abc    cde    cef    gbc    gab    aaa  
Iteration No6->abb    aab    abc    cde    cef    gab    gbc    aaa  
Iteration No7->abb    aab    abc    cde    cef    gab    aaa    gbc  
End of array pass for 8 Elements. Array is not sorted
```

Δώστε τον πηγαίο κώδικα

Οργάνωση Διάλεξης - Αναζήτηση

- Ταξινόμηση με χρήση της i2p βιβλιοθήκης
- Ταξινόμηση - Bubble Sort
 - A step-by-step Approach για πίνακα ακεραίων
 - Εναλλακτικές Υλοποιήσεις (+Αναδρομικότητα)
 - Ταξινόμηση πίνακα Δεικτών σε ακεραίους
 - Ταξινόμηση Λέξεων
 - Ταξινόμηση πίνακα αλφαριθμητικών
 - Ταξινόμηση πίνακα Δεικτών σε αλφαριθμητικά
- **Αναζήτηση (Search)**
 - Linear Search
 - Binary Search
 - Σε πίνακα ακεραίων
 - Σε πίνακα αλφαριθμητικών

Αναζήτηση (Linear-Binary) 4Int

```
5 int linearSearch(int ar[],int numElements, int key);
6 int binarySearch(int arr[], int low, int high, int num);
7
8 int ar[] = {0,1,2,3,4,5,6,7,8,9};
9
10 int main(int argc, char *argv[]) {
11     int index=0;
12     int numElements =sizeof(ar)/sizeof(int);
```

■ Linear Search

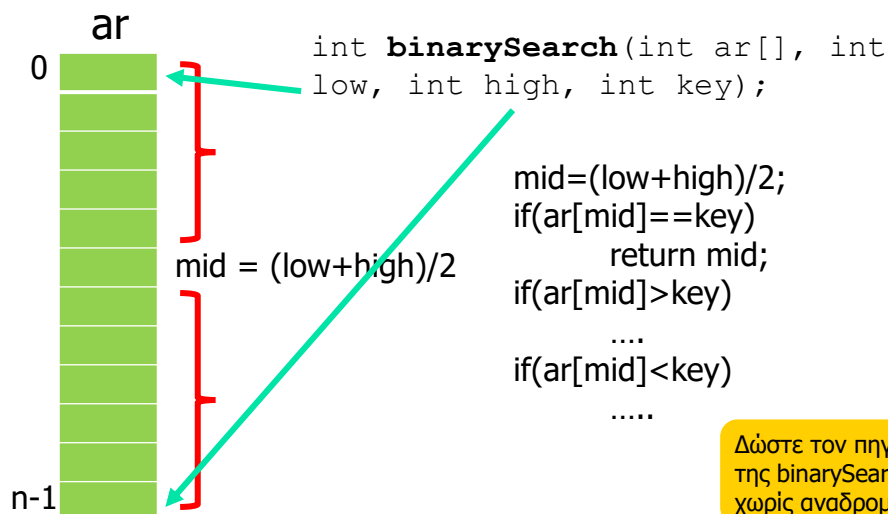
- Δώστε τον ορισμό της linearSearch()
- Δώστε παράδειγμα χρήσης της

■ Binary Search

- Δώστε τον ορισμό της binarySearch()
- Δώστε παράδειγμα χρήσης της

Δώστε τον πηγαίο κώδικα

Binary Search



Αναζήτηση (Linear-Binary) 4String

```
6  int linearSearch(char *ar[],int numElements, char *key);
7  int binarySearch(char *ar[], int low, int high, char *key);
8
9  char *ar[] = {"cde","abc","cef","abb","aab",
10             "klm","kab","gbc","gab","aaa"};
11
12  int main(int argc, char *argv[]) {
13      int index=0;
14      int numElements =sizeof(ar)/sizeof(char *);
15      char key[] = "gbc";
```

■ Linear Search

- Δώστε τον ορισμό της linearSearch()
- Δώστε παράδειγμα χρήσης της

■ Binary Search

- Δώστε τον ορισμό της binarySearch()
- Δώστε παράδειγμα χρήσης της

Δώστε τον πηγαίο κώδικα

How I feel when my code works



Απαραίτητη Προϋπόθεση

- A) Κατανόηση του προς ανάθεση έργου (είσοδοι, έξοδοι, ...)
- B) γνώση του αλγορίθμου,
- Γ) δυνατότητα περιγραφής του στην ομιλούμενη γλώσσα
- Δ) δυνατότητα μετατροπής της λεκτικής περιγραφής σε source code
- E) παραγωγή εκτελέσιμου κώδικα