

Δομές Δεδομένων & Ανάλυση Αλγορίθμων

3ο Εξάμηνο

Γραμμικές Δομές Δεδομένων

(Linear Data Structures)

- Πίνακες (Arrays)
- Class ArrayList

Δημοσθένης Σταμάτης

<http://www.iee.ihu.gr/~demos>

Τμήμα Μηχανικών Πληροφορικής & Ηλεκτρονικών Συστημάτων

Γραμμική Δομή Δεδομένων

Ονομάζεται η δομή δεδομένων το σύνολο των στοιχείων της οποίας είναι διατεταγμένο με τέτοιο τρόπο ώστε να ισχύουν τα εξής:

- (α) υπάρχει ένα στοιχείο το οποίο ονομάζεται αρχή και έχει ένα και μόνον ένα επόμενο στοιχείο
- (β) υπάρχει ένα στοιχείο το οποίο ονομάζεται τέλος και έχει ένα και μόνον ένα προηγούμενο
- (γ) κάθε άλλο στοιχείο έχει ένα και μόνον ένα προηγούμενο και ένα και μόνον ένα επόμενο



ΠΙΝΑΚΕΣ (ARRAYS)

- ☞ Οι πίνακες στη Java θεωρούνται αντικείμενα και επομένως οι μεταβλητές τύπου πίνακα συμπεριφέρονται σαν μεταβλητές τύπου κλάσης είναι δηλαδή αναφορές (references) σε αντικείμενα.
- ☞ Αν και δεν υπάρχει κλάση με την ονομασία array αυτή υλοποιείται εσωτερικά από τη Java για λόγους αποτελεσματικότητας.
- ☞ Το μέγεθος του πίνακα καθορίζεται με τη δημιουργία του και δεν μπορεί να αλλάξει κατά τη διάρκεια της εκτέλεσης του προγράμματος (στατικό μέγεθος):

Αν **A** πίνακας :

```
int i = A.length;
A.length = 100; // Προσοχή, λάθος!
```

Δήλωση Πίνακα

☞ `int [] pin;` ή ισοδύναμα `int pin [];`

Ανάθεση μνήμης για τα στοιχεία του Πίνακα

☞ `pin = new int[10];`

☞ `int j = 5;`
`int max = 10 * j;`
`int [] pin = new int[max];`

Δήλωση & Ανάθεση μαζί

☞ `int [] pin = new int[10];`

Αρχικοποίηση Πίνακα

☞ `int[] pin = {2,4,6,8,10,12,14,16,18,20};`

Αρχικοποίηση μεταβλητής Πίνακα

☞ `char[] data = null;`

Ενδείκτες Πίνακα

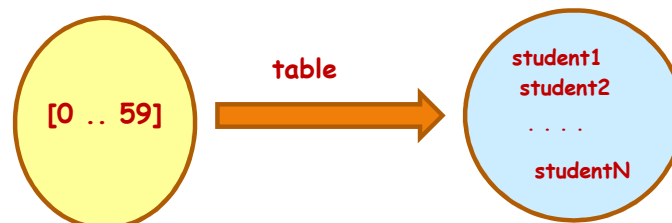
☞ `a = 3;`
`b = 5;`
`pin[a+b] += 2;`

Η Java πραγματοποιεί έλεγχο εύρους (*range checking*) στους ενδείκτες (*subscripts*) των πινάκων

ΠΙΝΑΚΕΣ (ARRAYS)

☞ Τα στοιχεία ενός πίνακα μπορεί να είναι οποιοδήποτε τύπου, είτε πρωταρχικού (π.χ. `int`, `char` κλπ) είτε τύπου κλάσης (π.χ. `student`). Στη δεύτερη περίπτωση αντιμετωπίζουμε κάθε στοιχείο του πίνακα σα μεταβλητή τύπου κλάσης.

`Student[] table = new Student[60]`



☞ Ο πίνακας είναι μία απεικόνιση από ένα πεδίο ορισμού (τιμές ενδείκτη) σε ένα πεδίο τιμών (τιμές των στοιχείων)

ΠΟΛΥΔΙΑΣΤΑΤΟΙ ΠΙΝΑΚΕΣ

👉 Εάν τα στοιχεία ενός πίνακα είναι με τη σειρά τους τύπου πίνακα, τότε αυτός ονομάζεται Πολυδιάστατος

Παραδείγματα δισδιάστατων πινάκων:

👉 `String [] [] pin2;`
`pin2 = new String [3] [4];`

👉 `int [] [] pin3= { {4,5}, {1,7} };`

👉 `int [] [] pin4;`
`pin4 = new int[2] [];`
`pin4[0] = new int[3]; //γραμμές διαφορετικού μήκους`
`pin4[1] = new int[5];`

ArrayList

👉 Γραμμική Δομή της οποίας το μέγεθος μεταβάλλεται.
Υλοποιείται από την κλάση `ArrayList`

```
java.lang.Object
java.util.AbstractCollection<E>
java.util.AbstractList<E>
java.util.ArrayList<E>
```

Δομητές (παραδείγματα):

👉 `ArrayList AL2 = new ArrayList();`
Αρχικό μέγεθος εξ ορισμού 10 θέσεις.
(Το μέγεθος αυξάνει όταν χρειάζονται επιπλέον στοιχεία)

👉 `ArrayList AL1 = new ArrayList(20);`
Αρχικό μέγεθος 20 θέσεις -
(Το μέγεθος αυξάνει όταν χρειάζεται)

👉 `ArrayList<String> Students = new ArrayList<String>(20);`
Λίστα για 20 φοιτητές (αναπαριστώνται με `Strings`)
(Το μέγεθος αυξάνει όταν χρειάζεται)

Class ArrayList

Βασικές Μέθοδοι (1/2)

public void add(E Item)

Προσθέτει το **Item** στο τέλος της λίστας

public void add(int index, E item,)

Παρεμβάλει το **item** σαν νέο στοιχείο της λίστας στη θέση **index**

public E get(int index)

Επιστρέφει το στοιχείο της λίστας που βρίσκεται στη θέση **index**

public E set(Object item, int index)

Τοποθετεί στο υπ' αριθμό **index** στοιχείο της λίστας το **item**. Επιστρέφει το παλιό στοιχείο που αντικαταστάθηκε από το **item**

public boolean isEmpty()

Ελέγχει εάν η λίστα δεν έχει καθόλου στοιχεία

ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ

Γ' ΕΞΑΜΗΝΟ

Class ArrayList

Βασικές Μέθοδοι (2/2)

public boolean contains(Object item)

Ελέγχει εάν το **item** αποτελεί στοιχείο της λίστας. (**Object vs E!**)

public int indexOf(Object item)

Επιστρέφει τη θέση του **item** στη λίστα. Αν δεν υπάρχει επιστρέφει -1

public boolean remove(Object item)

Βρίσκει και διαγράφει το στοιχείο **item** της λίστας εφόσον υπάρχει

public int size()

Επιστρέφει το τρέχον μέγεθος της λίστας (αριθμό των στοιχείων της)

public void trimToSize()

"Ψαλιδίζει" τη χωρητικότητα της λίστας ώστε να γίνει ίση με το τρέχον μέγεθός της

ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ

Γ' ΕΞΑΜΗΝΟ

Η κλάση ArrayList στο Application Programming Interface (API) της Java

docs.oracle.com/en/java/javase/13/docs/api/java.base/java/util/ArrayList.html#set(int,E)

OVERVIEW MODULE PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

SUMMARY NESTED FIELD CONSTR METHOD DETAIL FIELD CONSTR METHOD

Module java.base
Package java.util
Class ArrayList<E>

java.lang.Object
 java.util.AbstractCollection<E>
 java.util.AbstractList<E>
 java.util.ArrayList<E>

Type Parameters:
 E - the type of elements in this list

All Implemented Interfaces:
 Serializable, Cloneable, Iterable<E>, Collection<E>, List<E>, RandomAccess

Direct Known Subclasses:
 ArrayList, RoleList, RoleUnresolvedList

```
public class ArrayList<E>
    extends AbstractList<E>
    implements List<E>, RandomAccess, Cloneable, Serializable
```

Resizable-array implementation of the List interface. Implements all optional list operations, and permits all elements, including null. In all methods to manipulate the size of the array that is used internally to store the list. (This class is roughly equivalent to Vector, except that its size, isEmpty, get, set, iterator, and listIterator operations run in constant time. The add operation runs in amortized constant time operations run in linear time (roughly speaking). The constant factor is low compared to that for the LinkedList implementation.

Each ArrayList instance has a capacity. The capacity is the size of the array used to store the elements in the list. It is always at least as large as capacity grows automatically. The details of the growth policy are not specified beyond the fact that adding an element has constant amortized time.

An application can increase the capacity of an ArrayList instance before adding a large number of elements using the ensureCapacity operation. This operation can be used to hint at an expected number of future elements of the list, so that if the capacity is too small, the list can be immediately reallocated to a larger array. For example, adding an element to the end of the ArrayList with a capacity of 10 will cause the array to be reallocated to a new array of capacity 16. This is useful to avoid the time and space cost of creating a new ArrayList object for every addition.

Note that this implementation is not synchronized. If multiple threads access an ArrayList instance concurrently, and at least one of them externally, then the following conditions must be met: (A structural modification is any operation that adds or deletes one or more elements, or explicitly resizes the backing array; merely

Η κλάση ArrayList στο Application Programming Interface (API) της Java

docs.oracle.com/en/java/javase/13/docs/api/java.base/java/util/ArrayList.html#set(int,E)

OVERVIEW MODULE PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

SUMMARY NESTED FIELD CONSTR METHOD DETAIL FIELD CONSTR METHOD SEARCH Search

Constructor Summary

Constructors

Constructor	Description
ArrayList()	Constructs an empty list with an initial capacity of ten.
ArrayList(int initialCapacity)	Constructs an empty list with the specified initial capacity.
ArrayList(Collection<? extends E> c)	Constructs a list containing the elements of the specified collection, in the order they are returned by the collection's Iterator.

Method Summary

All Methods	Instance Methods	Concrete Methods	
Modifier and Type	Method		Description
void	add(int index, E element)		Inserts the specified element at the specified position in this list.
boolean	add(E e)		Appends the specified element to the end of this list.
boolean	addAll(int index, Collection<? extends E> c)		Inserts all of the elements in the specified collection into this list, starting at the specified position.
boolean	addAll(Collection<? extends E> c)		Appends all of the elements in the specified collection to the end of this list, in the order that they are returned by the collection's Iterator.
void	clear()		Removes all of the elements from this list.
Object	clone()		Returns a shallow copy of this ArrayList instance.
boolean	contains(Object o)		Returns true if this list contains the specified element.