Διάλεξη 01
Δομές Δεδομένων και Αλγόριθμοι - Εισαγωγή

Στην ενότητα αυτή θα μελετηθούν τα εξής επιμέρους θέματα:
Εισαγωγή στις έννοιες:

- Δεδομένα και Ενδόμνημη Αναπαράσταση τους
- Οργάνωση Δεδομένων και Δομές Δεδομένων
- Αλγόριθμοι και Πολυπλοκότητα

Το μάθημα

Αλγόριθμοι

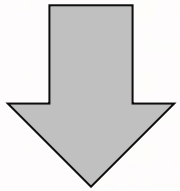
Μέθοδοι Επίλυσης Προβλημάτων

Δομές Δεδομένων

Μέθοδοι Αποθήκευσης Δεδομένων

Το μάθημα

Αλγόριθμοι + Δομές Δεδομένων



Προγραμματισμός και Επίλυση Προβλημάτων

Συμβόλαιο Μαθήματος

- **Επίπεδο:** Προπτυχιακό
- **Πίστωση:** 7.5 μονάδες ECTS
- **Μέθοδοι Διδασκαλίας**
  - Διαλέξεις (3 ώρες εβδομαδιαίως): Παράδοση Διδασ. Ύλης
  - Φροντιστήριο (1 ώρα εβδομαδιαίως): Ύλη / Θεωρητική Εξάσκηση
  - Εργαστήριο (2 ώρες εβδομαδιαίως): Πρακτική Εξάσκηση
- **Αξιολόγηση**
  - 45% Τελική Εξέταση
  - 30% Διαγνωστικά τεστ(3 συνολικά)
  - 5% Θεωρητικές Ασκήσεις
  - 20% Διαγνωστικά τεστ / Προγραμματιστικές Ασκήσεις (3 συνολικά)
  - 5% Quiz της ημέρας

Πληροφορίες Μαθήματος

- **Διδάσκων:** Γιώργος Πάλλης
  - **Γραφείο:** FST-01 B119
  - **Τηλέφωνο:** 22-892743
  - **E-mail:** gpallis@cs.ucy.ac.cy
  - **Ώρες Γραφείου:** Δευτέρα, 10:00-12:00 ή κατόπιν συνεννόησης
- **Εργαστήρια**
  - **Υπεύθυνος:** Χατζηπολλάς Γεώργιος
  - **Email:** hadjipollas.george@ucy.ac.cy

Προαπαιτούμενα

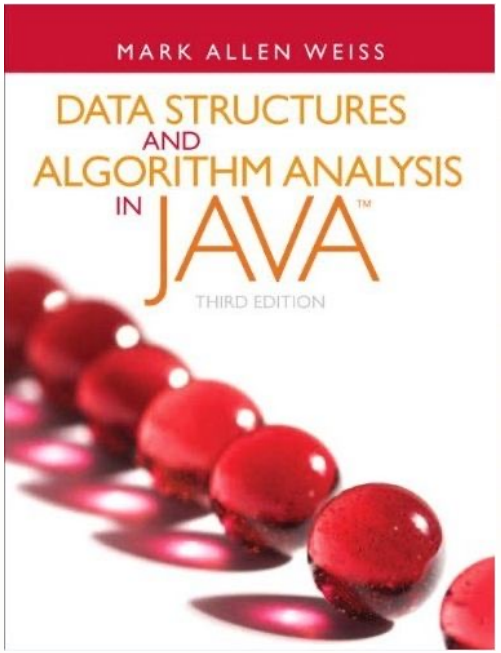
- **Προαπαιτούμενα:**
  - ΕΠΛ 111 (Διακριτές Δομές στην Πληροφορική και Υπολογισμό)
  - ΕΠΛ 133 (Αρχές Προγραμματισμού)

- **Το ΕΠΛ231 είναι προαπαιτούμενο για τα υποχρεωτικά μαθήματα**
  - ΕΠΛ236 (Αλγόριθμοι και Πολυπλοκότητα)
  - ΕΠΛ341 (Τεχνητή Νοημοσύνη)
  - ΕΠΛ342 (Βάσεις Δεδομένων)
- όπως και για διάφορα μαθήματα περιορισμένης επιλογής
  - ΕΠΛ421 (Προγραμματισμός Συστημάτων)
  - ΕΠΛ429 (Μεταγλωττιστές)
  - ΕΠΛ431 (Παράλληλοι Αλγόριθμοι)
  - ΕΠΛ422 (Κατανεμημένοι Αλγόριθμοι)
  - ΕΠΛ433 (Προγραμματισμός και Ικανοποίηση Περιορισμών)
  - ΕΠΛ442 (Μηχανική Μάθηση)
  - ΕΠΛ445 (Επεξεργασία Εικόνας)
  - ΕΠΛ442 (Υπολογιστική Όραση)
  - ΕΠΛ448 (Εξόρυξη Δεδομένων στον Παγκόσμιο Ιστό)
  - ΕΠΛ481 (Τεχνολογία Λογισμικού για Λογισμικό ως Υπηρεσία)

Βιβλιογραφία

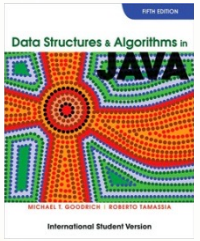
Βασική Βιβλιογραφία

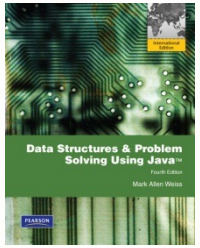
- Mark Allen Weiss, *Data Structures and Algorithm Analysis in Java*. 3rd edition, Addison Wesley, 2011.



Βοηθητική Βιβλιογραφία

- Σημειώσεις Μαθήματος και Συνοδευτικό Υλικό
- Data Structures & Algorithms in JAVA: Michael T. Goodrich, Roberto Tamassia, ISBN-10: 0470398809, ISBN-13: 978-0470398807, 2010.
- Data Structures and Problem Solving Using Java, Mark Allen Weiss, ISBN-10: 0321546229, ISBN-13: 9780321546227, Prentice Hall, 2009.
- Algorithm Design: Jon Kleinberg, Eva Tardos, ISBN-10: 813170310X, ISBN-13: 9788131703106, Pearson Education, 2006.
- Algorithms: Robert Sedgewick, Kevin Daniel Wayne, ISBN-10: 032157351X, ISBN-13: 978-0321573513, Addison-Wesley Professional, 2011.





ΕΠΛ231
Δομές Δεδομένων και Αλγόριθμοι

Οικοσελίδα Ανακοινώσεις Πρόγραμμα Εργαστήρια Ασκήσεις

http://www.cs.ucy.ac.cy/courses/EPL231/

Καλώς ήλθατε στην ιστοσελίδα του μαθήματος ΕΠΛ231 - Δομές Δεδομένων και Αλγόριθμοι

Το μάθημα μελετά μεθόδους οργάνωσης των πληροφοριών, αλγόριθμους μετασχηματισμού τους και την ανάλυση της πολυπλοκότητας των αλγορίθμων. Για την υλοποίηση των προγραμμάτων θα χρησιμοποιηθεί η γλώσσα προγραμματισμού JAVA.

Συμβόλαιο Μαθήματος

Συμβόλαιο Μαθήματος download

Διδάσκοντες Μαθήματος

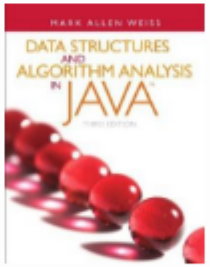
Διδάσκων Καθηγητής:	Γιώργος Πάλλης
Τηλέφωνο:	+357 22 89 27 43
ΦΑΞ:	+357 22 89 27 01
Γραφείο:	ΘΕΕ01 - Β119
Ηλ. Ταχυδρομείο:	gpallis-AT-cs.ucy.ac.cy
Ώρες Γραφείου:	Τρίτη, 10:00-12:00, ή κατόπιν συνεννόησης
Διαλέξεις:	Τρίτη και Παρασκευή, 12:00 – 13:30, ΧΩΔ-01 103

Τελευταίες Ασκήσεις

Τελευταίες Ανακοινώσεις

Σημαντικές Πληροφορίες

Το Blackboard θα χρησιμοποιείται στην υποβολή των εργασιών αλλά και για πρόσβαση στο εκπαιδευτικό υλικό των εργαστηριακών μαθημάτων.



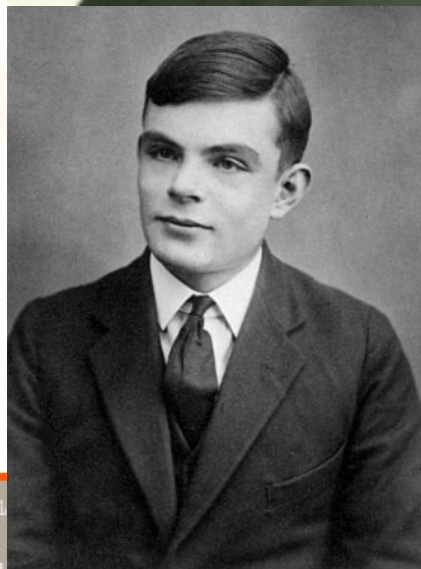
Data Structures and Algorithm Analysis in Java (3rd Edition)
Mark A. Weiss

Απορίες

- Διαλέξεις
- Εργαστήριο
- Φροντιστήριο
- Ώρες Γραφείου
- Ανακοινώσεις/Απορίες στο **Piazza**

Γιατί είναι σημαντικό το μάθημα;

- Οι αλγόριθμοι έχουν παλιές ρίζες, νέες ευκαιρίες.
  - Η μελέτη των αλγορίθμων ξεκινά τουλάχιστον από τον Ευκλείδη.
  - Διατυπώθηκαν από τους Church και Turing τη δεκαετία του 1930.
  - Μερικοί σημαντικοί αλγόριθμοι ανακαλύφθηκαν από προπτυχιακούς φοιτητές στα πλαίσια παρόμοιων μαθημάτων!



Γιατί είναι σημαντικό το μάθημα;

- Η επίδρασή των αλγορίθμων είναι ευρεία και εκτεταμένη
  - **Internet**. Web search, packet routing, distributed file sharing, ...
  - **Biology**. Human genome project, protein folding, ...
  - **Computers**. Circuit layout, file system, compilers, ...
  - **Computer graphics**. Movies, video games, virtual reality, ...
  - **Security**. Cell phones, e-commerce, voting machines, ...
  - **Multimedia**. MP3, JPG, DivX, HDTV, face recognition, ...
  - **Social networks**. Recommendations, news feeds, advertisements, ...
  - **Physics**. N-body simulation, particle collision simulation, ...

Γιατί είναι σημαντικό το μάθημα;

Video Games

Δεδομένα

- **Τα δεδομένα βρίσκονται στο χαμηλότερο επίπεδο αφαιρετικότητας**
- **Από τα δεδομένα προκύπτει η πληροφορία και η γνώση.**

- Τα δεδομένα που χρησιμοποιούμε **δεν έχουν πάντοτε τον ίδιο βαθμό πολυπλοκότητας**, π.χ., "Στοιχεία Φοιτητή" vs. "Ηλικία Ατόμου"

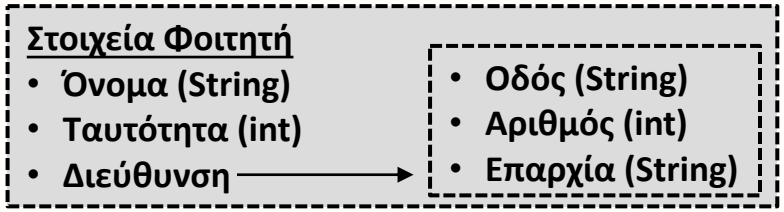
- Μερικά μπορούν να αναλυθούν σε απλούστερα συστατικά, ενώ άλλα δεν μπορούν. Δεδομένα που δεν μπορούν να αναλυθούν λέγονται ***πρωτογενή δεδομένα*** **– primitive data types (π.χ. int, char)**

- Τα δεδομένα ενός προβλήματος συνήθως δεν είναι μια άμορφη συλλογή στοιχείων. Τις περισσότερες φορές μπορούν να εκφραστούν με γνωστές μαθηματικές δομές, π.χ.
  - σαν ένα απλό **σύνολο** (διακριτών στοιχείων): {2, 3, 5, 7, 11, 13}
  - σαν ένα **διάνυσμα** (διατεταγμένων στοιχείων): (2, 3, 5, 7, 11, 13)
  - σαν ένας **πίνακας** : $\begin{pmatrix} 2 & 3 & 5 \\ 7 & 11 & 13 \end{pmatrix}$

Δομές Δεδομένων

- **Οι δομές δεδομένων είναι συλλογές πρωτόγονων δεδομένων, που συνδυάζονται για να σχηματίσουν πολυπλοκότερα δεδομένα.**

- Χρησιμοποιώντας τους αρχέγονους τύπους δεδομένων, μπορούμε να συνθέσουμε πιο πολύπλοκες δομές



- Θα χρησιμοποιήσουμε τον όρο ***πίνακας*** για να περιγράψουμε μια συλλογή στοιχείων τα οποία θα ονομάσουμε ***κόμβους*** ή ***καταχωρήματα***.
- Ένας κόμβος μπορεί να είναι ***απλός***, δηλαδή να αποτελείται από ένα μόνο πρωτόγονο δεδομένο, ή να είναι ***σύνθετος***, οπότε αποτελείται από δύο ή περισσότερα ***πεδία***. Τα πεδία ενός κόμβου μπορεί να αντιπροσωπεύουν πρωτόγονα δεδομένα διαφόρων τύπων.

Παράδειγμα Δομής Φοιτητή στην JAVA

```
class Address{
   String street;
   int number;
}
class Student{
   String name;
   int id;
   Address saddress;
}
```

Δεδομένα (διεύθυνση και μήκος)

- Ένα πεδίο ή ένα καταχώρημα χαρακτηρίζεται από τη ***διεύθυνση*** του και το ***μήκος*** του. Η διεύθυνση ενός πεδίου ή ενός καταχωρήματος είναι η διεύθυνση της πρώτης του κυψελίδας μνήμης και το μήκος του είναι ο αριθμός κυψελίδων από τις οποίες αποτελείται.

Address ← 4 bytes →*	Value
3669504	1
3669508	2
3669512	3
3669516	A
3669520	5

int a[] μήκος=3

Παραδείγματα στην C

```
int x = 5;
printf("x=%d\tAddress of x=%d\n",
        x, &x);
int c = 'A';
printf("c=%c\tAddress of c=%d\n",
        c, &c);
int a[]={1,2,3};
for(int i=0; i<3; i++){
   printf("a[%d]=%d\tAddress of
   a[%d]=%d\n", i,a[i],i,&a[i]);
}
```

- Η διεύθυνση του πρώτου πεδίου και ολόκληρου του πίνακα είναι **3669504**. Η διεύθυνση του δεύτερου καταχωρήματος **3669508** κ.ο.κ.

* Ισχύει για x86 αρχιτεκτονικές με σύνολο διευθύνσεων $2^{32} \approx 4x10^9$.
Για x64 αρχιτεκτονικές είναι 8 bytes με σύνολο διευθύνσεων $2^{64} = 18x10^{18}$!

Ενδόμνημη Παράσταση Δομών Δεδομένων

- Για να αναφερθούμε σε ένα καταχώρημα ή σε ένα πεδίο, χρησιμοποιούμε συνήθως συμβολικά ονόματα (όπως σε γλώσσες προγραμματισμού η ονομασία μεταβλητών αντιστοιχεί σε μία λέξη της μνήμης, π.χ., array).
- Για να πραγματοποιηθεί μια δομή δεδομένων, απαιτείται η **συσχέτιση** των **συμβολικών ονομάτων** με αντίστοιχα τμήματα της μνήμης του υπολογιστή. Αυτό μπορεί να γίνει με δύο τρόπους:

1. Διαδοχική (Στατική) Χορήγηση Μνήμης

Address	Value
3143928	1
3143932	2
3143936	3
3143940	4
3143944	5
3143948	6

Παράδειγμα στην C

```
int b[2][3]={{1,2,3},
             {4,5,6}};
for(int i=0; i<2; i++){
  for(int j=0; j<3; j++){
    printf("Address of
            b[%d][%d]=%d\n",
            i,j,&b[i][j]);
   }
}
```

Τυπώνει

```
Address of b[0][0]= 3143928
Address of b[0][1]= 3143932
Address of b[0][2]= 3143936
Address of b[1][0]= 3143940
Address of b[1][1]= 3143944
Address of b[1][2]= 3143944
```

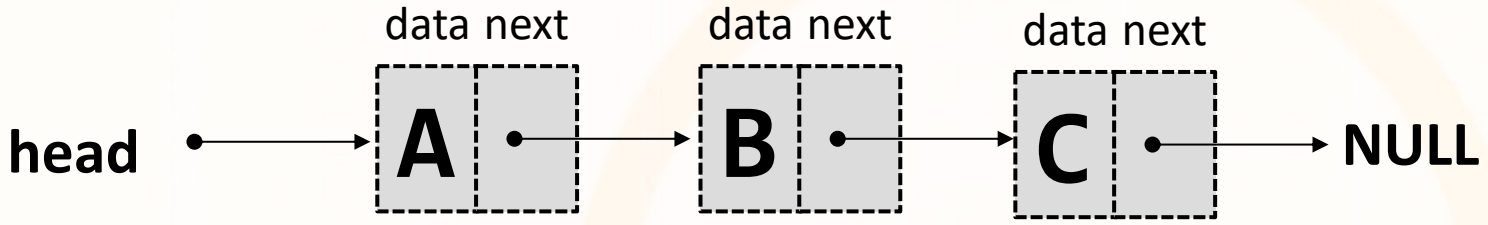
Ενδόμνημη Παράσταση Δομών Δεδομένων

2. Συνδετική Χορήγηση Μνήμης

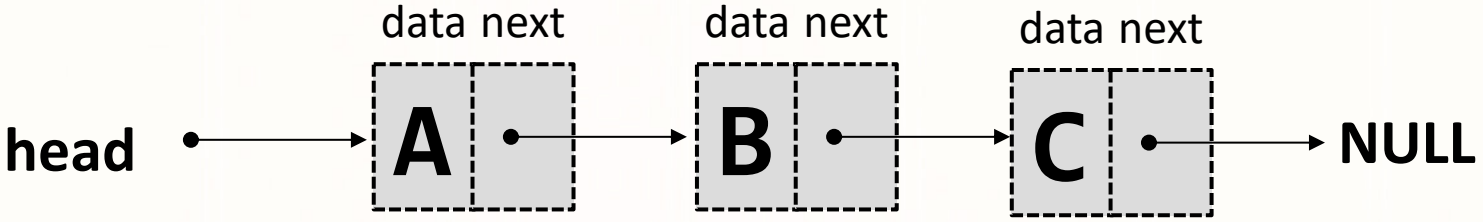
Ένα πεδίο μπορεί να παριστά τη διεύθυνση ενός άλλου πεδίου. Έτσι κατά τη **συνδετική χορήγηση** μνήμης κάθε κόμβος περιέχει πληροφορία σχετικά με το που **βρίσκεται κάποιος άλλος κόμβος της δομής** (π.χ., ο επόμενος, ο προηγούμενος).

Για την αποθήκευση αυτής της πληροφορίας κάθε κόμβος χρειάζεται ένα **ειδικό πεδίο** (**pointer**). Το πεδίο αυτό είναι χαρακτηριστικό της συνδετικής χορήγησης μνήμης.

Παράδειγμα: Απλά Συνδεδεμένη λίστα (linked-list)



Παράδειγμα: Απλά Συνδεδεμένη λίστα



Παράδειγμα στην JAVA

```
class Node {
  char data;
  Node next;
}
Node n1, n2, n3;
n1.data='A';
n2.data='B';
n3.data='C';

n1.next=n2;
n2.next=n3;
n3.next=null;
```

Node	Address	data	next
n2	3143928	B	3143948
	3143932		
n1	3143936	A	3143928
	3143940		
	3143944		
n3	3143948	C	0

Οι κόμβοι βρίσκονται σε τυχαίες θέσεις μνήμης.

Στην JAVA δεν μπορούμε να διαχειριστούμε τη μνήμη*. Τη διαχειρίζεται το JVM

Αλγόριθμοι

Τι είναι ένας αλγόριθμος;

Abu Jafar Mohammed ibn Musa Al-Khowarizmi (790-840)

Αλγόριθμος είναι μια πεπερασμένη ακολουθία εντολών, αυστηρά καθορισμένων και εκτελέσιμων σε πεπερασμένο (μετρήσιμο) χρόνο, οι οποίες αν ακολουθηθούν επιτυγχάνεται κάποιο επιθυμητό αποτέλεσμα.

Απαραίτητα κριτήρια:

- Υπάρχει είσοδος και έξοδος
- Καθορισμός εντολών (όχι ασάφειες).
- Περατότητα (να διεκπεραιώνει τον στόχο).

Επίδοση
δεν είναι απαραίτητη
αλλά άκρως επιθυμητή

Εξίσωση Wirth

Αλγόριθμοι + Δεδομένα = Προγράμματα

Δηλαδή, δοθέντος ενός **προβλήματος**, ένας αλγόριθμος παρέχει τις **οδηγίες** σύμφωνα με τις οποίες τα ***δεδομένα*** του προβλήματος ***μετασχηματίζονται*** και ***συνδυάζονται*** για να προκύψει η **λύση του προβλήματος**.

Αλγόριθμοι (συν.)

" For me, great algorithms are the poetry of computation. Just like verse, they can be terse, allusive, dense, and even mysterious. But once unlocked, they cast a brilliant new light on some aspect of computing. " — *Francis Sullivan*

" An algorithm must be seen to be believed. " — *Donald Knuth*

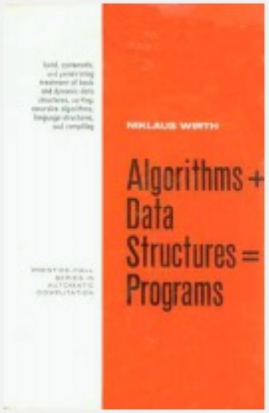
New York Times: The Yoda of Silicon Valley

Αλγόριθμοι (συν.)

" I will, in fact, claim that the difference between a bad programmer and a good one is whether he considers his code or his data structures more important. Bad programmers worry about the code. Good programmers worry about data structures and their relationships. "

— Linus Torvalds (creator of Linux)

" Algorithms + Data Structures = Programs. " *— Niklaus Wirth*



Παράδειγμα Αλγοριθμικού Προβλήματος

Το πρόβλημα επιλογής (selection problem)

- **Πρόβλημα: Έστω ότι έχουμε n αριθμούς και θέλουμε να προσδιορίσουμε τον k-οστό πιο μεγάλο.**

  π.χ. Έστω οι αριθμοί {5, 72, 3, 4, 1, 9, 65} (72,65,9,5,4,3,1)

  Ο 2^{os} πιο μεγάλος είναι ο 65, ο 6^{os} πιο μεγάλος είναι ο 3, κοκ.

- Αυτό το πρόβλημα είναι γνωστό ως το **πρόβλημα επιλογής (the selection problem).** Υπάρχουν διάφοροι 'εύκολοι' τρόποι λύσης:
  1. **Sort-Based:** 'Διαβάζουμε' τους n αριθμούς σε μια λίστα, a. Ταξινομούμε τη λίστα από το μεγαλύτερο στο μικρότερο με βάση κάποιο αλγόριθμο ταξινόμησης. Επιστρέφουμε το (k-1)οστό στοιχείο της λίστας, δηλαδή το a[k-1].
  2. **Buffer-based:** 'Διαβάζουμε' τους k πρώτους αριθμούς σε μια λίστα a. Ταξινομούμε τη λίστα από το μεγαλύτερο στο μικρότερο δηλ για k=3 έχουμε a={72,5,3}. Μετά, επεξεργαζόμαστε τους υπόλοιπους n-k αριθμούς ως εξής: αν ένα στοιχείο είναι πιο μικρό από το a[k-1] το αγνοούμε, διαφορετικά το τοποθετούμε στη σωστή θέση της λίστας. Όταν η διαδικασία αυτή τελειώσει, επιστρέφουμε το k-οστό στοιχείο της λίστας, δηλαδή το a[k-1].

Το πρόβλημα επιλογής

- Ποιος από τους δύο αλγόριθμους είναι ο καλύτερος;
- Οι αλγόριθμοι είναι ικανοποιητικοί;
- Υλοποίηση των αλγορίθμων και εφαρμογή τους σε υπολογιστή με n=1,000,000 και k=500,000 χρειάζεται πάρα πολύ χρόνο για να τερματίσει.
- Εντούτοις υπάρχει αλγόριθμος που επιτυγχάνει το ίδιο αποτέλεσμα σε δευτερόλεπτα.

Στόχος Μαθήματος

- Κατά την εκτέλεση ενός αλγόριθμου **η δομή που έχουν τα δεδομένα παίζει πολύ μεγάλο ρόλο**.
- Επίσης το να γράφουμε **ένα σωστό πρόγραμμα δεν είναι αρκετό**. Ιδιαίτερα, όταν το σύνολο αρχικών δεδομένων είναι μεγάλου μεγέθους, ο ***χρόνος εκτέλεσης*** ενός προγράμματος είναι κύριας σημασίας.
- Στο μάθημα αυτό θα μάθουμε να i) **υπολογίζουμε το χρόνο εκτέλεσης αλγόριθμων** και να ii) **συγκρίνουμε την αποδοτικότητα διαφορετικών αλγόριθμων**, **προτού τους υλοποιήσουμε.** Θα μελετήσουμε επίσης μεθόδους βελτίωσης της ταχύτητας προγραμμάτων.

Κεντρικός στόχος του μαθήματος είναι η μελέτη δομών δεδομένων, αναπαράστασής τους στη μνήμη ενός υπολογιστή, και αλγόριθμων που τις δημιουργούν και τις επεξεργάζονται.

Περιεχόμενα

- Πολυπλοκότητα αλγόριθμων και ανάλυση μέσης και χείριστης περίπτωσης.
- Τύποι δεδομένων και αφηρημένοι τύποι δεδομένων.
- Τύποι λίστας, σωρού και ουράς. Παράσταση και αποδοτική υλοποίηση τέτοιων δομών.
- Μη γραμμικές δομές δεδομένων. Δένδρα. Δένδρα διερεύνησης. Ισοζυγισμένα δένδρα.
- Bit-Διανύσματα. Τεχνικές κατακερματισμού (hashing).
- Ουρές προτεραιότητας.
- Αλγόριθμοι ταξινόμησης και ανάλυση της αποδοτικότητάς τους.
- Γράφοι και αλγόριθμοι επεξεργασίας τους.

Μαθησιακά αποτελέσματα

Με το πέρας του μαθήματος αναμένεται να είστε σε θέση:

- Να ***αναλύετε*** και να ***συγκρίνετε*** την αποδοτικότητα αλγορίθμων βάσει των τάξεων Ο, Ω και Θ.
- Να ***χρησιμοποιείτε***, να ***υλοποιείτε*** και να ***επεκτείνετε*** τις δομές δεδομένων που θα μελετηθούν στο μάθημα.
- Να ***εφαρμόζετε*** τους αλγόριθμους που θα μελετηθούν στο μάθημα σε τυχαία δεδομένα.
- Να ***ορίζετε*** τους αφηρημένους τύπους δεδομένων που απαιτούνται για την οργάνωση των δεδομένων προβλημάτων.
- Να ***σχεδιάζετε*** και να ***υλοποιείτε*** αλγόριθμους που ελαχιστοποιούν τον χρόνο εκτέλεσής τους όπως και τον χώρο που χρησιμοποιούν.
- Να ***επιλέγετε*** ή και να ***δημιουργείτε*** τις κατάλληλες δομές δεδομένων και τους κατάλληλους αλγόριθμους για υλοποίηση αφηρημένων τύπων δεδομένων.
- Να ***επιλύετε*** και να ***υλοποιείτε*** αποδοτικές λύσεις σε σύνθετα υπολογιστικά προβλήματα χρησιμοποιώντας τη γλώσσα Java.

Coursera

- https://www.coursera.org/course/algs4partI