Πανεπιστήμιο Αιγαίου
Τμήμα Γραφικών Τεχνών & Σχεδίασης Συστημάτων

Οργάνωση του μαθήματος

Τεχνολογίες & Μεθοδολογίες Προγραμματισμού

Ιωάννης Γαβιώτης, gaviotis@aegean.gr

Περιγραφή του μαθήματος

Το μάθημα αυτό έχει στόχο να διδάξει στους φοιτητές πώς να λύνουν προβλήματα χρησιμοποιώντας υπολογιστές και να τους εισάγει στον προγραμματισμό, ενθαρρύνοντας καλές τεχνικές ανάπτυξης. Προϋποθέτει βασικές γνώσεις οργάνωσης και λειτουργίας υπολογιστικών συστημάτων που αποκτήθηκαν στο μάθημα «Πληροφορική» του α΄ έτους.

Η σχεδίαση προγραμμάτων, δηλαδή ο τρόπος με τον οποίο ο προγραμματιστής δημιουργεί τη δομή του προγράμματος, διδάσκεται ανεξάρτητα από συγκεκριμένη γλώσσα προγραμματισμού[1]. Αναλύονται διάφορα προβλήματα και αναπτύσσεται η λύση τους, εκφρασμένη έτσι ώστε να μπορεί να εκτελεστεί από υπολογιστή. Διαγράμματα ροής και ψευδοκώδικας χρησιμοποιούνται για την αρχική έκφραση του αλγορίθμου με σαφήνεια και ελεγξιμότητα.

Όσο η λύση γίνεται πιο πολύπλοκη, καθίσταται αναγκαία η χρήση τεχνικών βαθμιαίας αποδόμησης κατά το πρότυπο του δομημένου προγραμματισμού και της αντικειμενοστρεφούς σχεδίασης και υλοποίησης. Εξετάζεται ο μεθοδικός μετασχηματισμός των αναλυτικών προδιαγραφών σε δομημένες λύσεις. Επίσης, περιγράφεται το μοντέλο του οδηγούμενου από συμβάντα υπολογισμού σε σχέση με γραφικά περιβάλλοντα.

Για να κατανοηθούν οι αρχές και οι τεχνικές που περιγράφονται στις διαλέξεις του μαθήματος και για να αποκτήσουν οι φοιτητές πρακτική εμπειρία, στα εργαστήρια έρχονται σε επαφή με συγκεκριμένες γλώσσες και περιβάλλοντα προγραμματισμού, όπως η Visual Basic και η Java. Εκτός από τη σύνταξη προγραμμάτων, στα εργαστήρια περιγράφεται η χρήση συντακτών κατευθυνόμενου συντακτικού (syntax-directed editors), εργαλείων εκσφαλμάτωσης (debugging tools) και προκατασκευασμένων χειριστηρίων (component-based programming).

Γνωστικό πεδίο

Στην ενότητα αυτή περιγράφεται το θεματικό αντικείμενο για το μάθημα ακολουθώντας την κατηγοριοποίηση του πιο πρόσφατου Computing Curriculum[2]. Κατά κύριο λόγο το μάθημα καλύπτει τα γνωστικά πεδία που έχουν κωδικοποιηθεί ως PF (Programming Fundamentals) και PL (Programming Languages). Πιο συγκεκριμένα, το πεδίο γνώσης PF καλύπτεται καθ' ολοκληρίαν (PF1-PF5), ενώ το PL καλύπτεται μερικώς (PL1-PL6). Σημειώνουμε ότι όλα τα πεδία που είναι ενταγμένα στο μάθημα αναγνωρίζονται από την αναφορά Steelman ως ύλη κορμού.

PF	**Programming Fundamentals**	**Εισαγωγή στον προγραμματισμό**	**38 ώρες[3]**
PF1	Fundamental programming constructs	Βασικές δομές προγραμματισμού	9
PF2	Algorithms and problem-solving	Αλγόριθμοι και επίλυση προβλημάτων	6
PF3	Fundamental data structures	Βασικές δομές δεδομένων	14
PF4	Recursion	Αναδρομή	5

[1] Η προσέγγιση αυτή ακολουθείται προς αποφυγή του "συνδρόμου της άνω τελείας" (semicolon syndrome), όπου ο φοιτητής παρασύρεται στις λεπτομέρειες του (αυστηρού) συντακτικού των γλωσσών προγραμματισμού, αντί να δίνει έμφαση στις μεθοδολογίες αντιμετώπισης των προβλημάτων και τις τεχνικές προγραμματισμού των λύσεών τους [12].

[2] The Joint Task Force on Computing Curricula (IEEE & ACM), *Computing Curricula 2001 – Computer Science*, Steelman report, final draft version, Dec. 15, 2001

[3] Όπως σημειώνεται στην αναφορά [σελ. 72], οι ώρες που διατίθενται για κάθε πεδίο αποτελούν τον ελάχιστο χρόνο διδασκαλίας που είναι απαραίτητος για να καλυφθεί η διδακτική ύλη. Επίσης, είναι ενδεικτικό μέγεθος που αντιστοιχεί μόνον στις ώρες διδασκαλίας (lecture). Υπολογίζεται ότι κάθε διδακτική ώρα απαιτεί άλλες 3 ώρες σπουδής. Έτσι μια ενότητα που εμφανίζεται να απαιτεί 5 ώρες, τυπικά συνεπάγεται 20 ώρες συνολικά (5 ώρες στην τάξη και άλλες 15 εκτός).

PL	**Programming Languages**	**Γλώσσες προγραμματισμού**	**21 ώρες**
PL1	Overview of programming languages	Επισκόπηση των γλωσσών προγραμματισμού	2
PL2	Virtual machines	Ιδεατές μηχανές	1
PL3	Introduction to language translation	Εισαγωγή στην μετάφραση γλωσσών	2
PL4	Declarations and types	Δηλώσεις και τύποι	3
PL5	Abstraction mechanisms	Μηχανισμοί αφαίρεσης	3
PL6	Object-oriented programming	Αντικειμενοστρεφής προγραμματισμός	10

Θέματα

Στην ενότητα αυτή δίνονται τα θέματα που καλύπτονται στο μάθημα ανά γνωστικό πεδίο.

PF1	Βασικό συντακτικό και σημειολογία γλωσσών υψηλού επιπέδου Μεταβλητές, τύποι, παραστάσεις και ανάθεση Απλή είσοδος / έξοδος Δομές ελέγχου επιλογής και επανάληψης Συναρτήσεις και πέρασμα παραμέτρων Δομημένη αποδόμηση
PF2	Στρατηγικές επίλυσης προβλημάτων Ο ρόλος των αλγορίθμων στη διαδικασία επίλυσης προβλημάτων Στρατηγικές υλοποίησης για αλγόριθμους Στρατηγικές εκσφαλμάτωσης Η έννοια και οι ιδιότητες του αλγορίθμου
PF3	Βασικοί τύποι Πίνακες Εγγραφές Συμβολοσειρές και επεξεργασία συμβολοσειρών Αναπαράσταση δεδομένων στη μνήμη Εκχώρηση μνήμης στατικά, με στοίβα (stack) και με σωρό (heap)
PF4	Η έννοια της αναδρομής Αναδρομικές μαθηματικές συναρτήσεις Απλές αναδρομικές διαδικασίες Στρατηγικές «διαίρει και βασίλευε»
PF5	Μέθοδοι χειρισμού συμβάντων Μετάδοση συμβάντων

PL1	Ιστορία των γλωσσών προγραμματισμού Συνοπτική επισκόπηση των προτύπων προγραμματισμού - Διαδικαστικές γλώσσες - Αντικειμενοστρεφείς γλώσσες - Λειτουργικές γλώσσες - Δηλωτικές, μη-αλγοριθμικές γλώσσες Τα αποτελέσματα της κλίμακας στην προγραμματιστική μεθοδολογία
PL2	Η έννοια της ιδεατής μηχανής Ιεραρχία ιδεατών μηχανών Ενδιάμεσες γλώσσες Ζητήματα ασφαλείας κώδικα που εκτελείται απομακρυσμένα
PL3	Σύγκριση μεταγλωττιστών και διερμηνευτών
PL4	Η έννοια των τύπων ως σύνολο τιμών μαζί με σύνολο λειτουργιών Έλεγχος τύπων Συλλογή απορριμμάτων
PL5	Διαδικασίες, συναρτήσεις ως μηχανισμοί αφαίρεσης Μηχανισμοί παραμέτρων (αναφορά, τιμή) Αρθρώματα στις γλώσσες προγραμματισμού
PL6	Αντικειμενοστρεφής σχεδίαση Ενθυλάκωση και απόκρυψη πληροφορίας Διαχωρισμός συμπεριφοράς και υλοποίησης Κλάσεις και υποκλάσεις Κληρονομικότητα (υπερφόρτωση) Πολυμορφισμός

Στόχοι

Οι φοιτητές που θα ολοκληρώσουν αυτό το μάθημα:

- θα αναπτύξουν δεξιότητα στην επίλυση προβλημάτων με υπολογιστές,
- θα γνωρίσουν το συντακτικό και τη σημασιολογία των τυπικών προγραμματιστικών δομών, όπως η ανάθεση, η επιλογή, η επανάληψη,
- θα γνωρίζουν να σχεδιάζουν και υλοποιούν ευανάγνωστα και αποδοτικά προγράμματα,
- θα ελέγχουν την ορθότητα των προγραμμάτων και την καταλληλότητά τους σε σχέση με το πρόβλημα και το χρήστη,
- θα γνωρίσουν τους βασικούς τύπους δεδομένων και τη δυνατότητα ορισμού νέων με χρήση δηλωτικών εντολών,
- θα κατανοούν τις βασικές αρχές του δομημένου προγραμματισμού, όπως η σταδιακή εκλέπτυνση (step-wise refinement), ο ορισμός και η κλήση διαδικασιών,
- θα κατανοούν τις βασικές αρχές του αντικειμενοστραφή προγραμματισμού και σχεδίασης, όπως οι κλάσεις, η απόκρυψη πληροφορίας και η κληρονομικότητα,
- θα έχουν κατανοήσει τη χρήση απλών δομών δεδομένων, όπως πίνακες, διανύσματα, στοίβες, σωροί, ουρές,
- θα αποκτήσουν αίσθηση της πολυπλοκότητας των λειτουργιών ενός αλγορίθμου, και τέλος
- εκτός από επιτακτικές (imperative) γλώσσες προγραμματισμού, θα γνωρίσουν τον συναρτησιακό (functional) και τον λογικό προγραμματισμό.

Δεν είναι στόχος του μαθήματος απλώς η εκμάθηση της σύνταξης μιας ή δύο γλωσσών προγραμματισμού. Συγκεκριμένες γλώσσες χρησιμοποιούνται για να κατανοήσουν οι φοιτητές τις αρχές, τεχνικές και μεθόδους που περιγράφονται στο μάθημα, ωστόσο θα πρέπει να έχουν τη δυνατότητα να τα εφαρμόσουν αυτά σε διάφορα περιβάλλοντα ανάπτυξης και γλώσσες.

Εκπαιδευτική προσέγγιση

Παρατίθεται προσαρμοσμένο απόσπασμα από το βιβλίο [9]:

[...] Εισαγωγικά μαθήματα στην επιστήμη των υπολογιστών μπορεί να δίνουν έμφαση είτε στις έννοιες και τη θεωρία, ή στην προγραμματιστική πρακτική, ή στην αρχιτεκτονική των υπολογιστών, ή στην τεχνολογία λογισμικού, κοκ. Οι φοιτητές που σπουδάζουν επιστήμη των υπολογιστών ή πληροφορική αργά ή γρήγορα θα εκτεθούν σε όλα αυτά τα θέματα και θα αποκτήσουν μια ισορροπημένη άποψη για όλα αυτά. Ωστόσο, τώρα μαθήματα στους υπολογιστές διδάσκονται σε φοιτητές από άλλους κλάδους, όπως μαθηματικούς, φυσικούς, μηχανικούς, ενώ η τάση είναι να διδάσκονται ακόμη και σε φοιτητές ανθρωπιστικών επιστημών. Αυτοί οι φοιτητές συνήθως έχουν ένα ή δύο μαθήματα σε υπολογιστές. Έτσι είναι πιθανό να αποκτήσουν μια μη ισορροπημένη άποψη ως αποτέλεσμα της περιορισμένης τους εμπειρίας. Μπορεί να μάθουν να προγραμματίζουν έναν υπολογιστή, αλλά να μην γνωρίσουν πώς λειτουργεί. Ή μπορεί να κατανοήσουν την εσωτερική αρχιτεκτονική των ολοκληρωμένων κυκλωμάτων, αλλά να μην συνδέσουν την γνώση τους με γλώσσες προγραμματισμού υψηλού επιπέδου. Η απουσία μιας «συνολικής εποπτείας» αφήνει στους φοιτητές την εντύπωση ότι πολλά ανεξήγητα χαρακτηριστικά της επιστήμης των υπολογιστών είναι «μαγικά». [...] Εκπαιδευτικός μας στόχος είναι να βοηθήσουμε φοιτητές που δεν ειδικεύονται στην επιστήμη των υπολογιστών, να κατανοήσουν τους υπολογιστές συνολικά.

Η προσέγγιση που ακολουθούμε ξεκίνησε με περιγραφές των βασικών συστατικών και έπειτα περιγράφεται η λειτουργικότητα που μπορούν να προσφέρουν αυτά τα συστατικά. Βαθμιαία περιγράφονται όλο και πιο πολύπλοκες λειτουργίες. Οι φοιτητές μαθαίνουν για αλγορίθμους, τη δομή της μηχανής και τη δομή των προγραμμάτων κι έτσι κατανοούν την εκτέλεση του πρώτου προγράμματος που γράφουν. Όταν πληκτρολογούν κάτι και διαβάζουν ένα μήνυμα καταλαβαίνουν τι συνέβη στη μηχανή και πώς παράχθηκε αυτό το αποτέλεσμα. Θέλουμε οι φοιτητές να αντιληφθούν ότι οι υπολογιστές δεν έχουν τίποτα το μαγικό ή ανεξήγητο [...]

Παρότι κατά κύριο λόγο ακολουθείται η προσέγγιση που περιγράφθηκε πρωτύτερα, σε ορισμένα σημεία ακολουθείται η ακριβώς αντίθετη μεθοδολογία, δηλαδή μια έννοια παρουσιάζεται μέσω ενός προγράμματος και της εκτέλεσής του, χωρίς να έχει προηγηθεί θεωρητική παρουσίαση και ανάλυση. Η μέθοδος αυτή κυρίως χρησιμοποιείται όταν οι φοιτητές έχουν κατανοήσει τα βασικά στοιχεία του προγραμματισμού και έχουν εξοικειωθεί με τις βασικές γλωσσικές δομές. Σε αυτή την περίπτωση, ο φορμαλισμός της ίδιας της γλώσσας χρησιμοποιείται ως εργαλείο για την πληρέστερη κατανόηση των νέων εννοιών που εισάγονται.

Διδάσκοντες

Το μάθημα διδάσκει ο Γιάννης Γαβιώτης (gaviotis@aegean.gr). Για τα εργαστήρια είναι υπεύθυνοι οι Δημήτρης Λέκκας (dlek@aegean.gr), Παναγιώτης Κουτσάμπασης (kgp@aegean.gr) και Βαγγέλης Βλαχογιάννης (evlach@aegean.gr).

Οι διαλέξεις του μαθήματος γίνονται στο Αμφιθέατρο και τα εργαστήρια στις δύο αίθουσες υπολογιστών παράλληλα και για τις δύο ομάδες φοιτητών.

Εργαστήριο

Στο εργαστήριο οι φοιτητές χωρίζονται σε δύο ομάδες από τους υπεύθυνους, οι οποίοι καθοδηγούν σε ασκήσεις που εφαρμόζουν τις αρχές που διδάχθηκαν στις διαλέξεις. Εάν ο φοιτητής δεν μπορεί να κάνει αυτό που ζητήθηκε, πρώτα ρωτάει τους γείτονες του και έπειτα ζητάει βοήθεια από τον υπεύθυνο. Οι ερωτήσεις / παρατηρήσεις των φοιτητών είναι σημαντικές γιατί μπορεί να αντιστοιχούν σε απορίες κι άλλων συναδέλφων τους. Επίσης, υποδηλώνουν παρακολούθηση και ενεργό συμμετοχή. Όταν κάποιος τελειώνει μια άσκηση, βλέπει εάν οι γείτονές του χρειάζονται βοήθεια, ή αν έχουν ακολουθήσει διαφορετική προσέγγιση. Η συνεργασία των φοιτητών στα εργαστήρια πρέπει να ενθαρρύνεται στο βαθμό που δεν δυσκολεύει το έργο του υπεύθυνου. Στις παραδοτέες εργασίες που αξιολογούνται, η συν εργασία επιτρέπεται στο βαθμό που πχ ανταλλάσσονται ιδέες ή τεχνικές, αλλά απαγορεύεται αυστηρά η αντιγραφή. Σε περίπτωση αντιγραφής τιμωρούνται όλοι οι εμπλεκόμενοι.

Κατά τη διάρκεια του εργαστηρίου:

- Δίδονται προς λύση ή διερεύνηση προβλήματα που αντιστοιχούν στην ύλη του μαθήματος.
- Οι φοιτητές συγγραφούν και ελέγχουν ως προς τη σύνταξη και τη λογική προγράμματα.
- Δίδεται βοήθεια για τις εργασίες που πρόκειται να παραδοθούν.

Εργαστήρια - Ασκήσεις – Εργασίες – Αξιολόγηση

Γλώσσες προγραμματισμού στα εργαστήρια: Visual Basic, Java

Προαπαιτούμενο λογισμικό εγκατεστημένο στους υπολογιστές: Microsoft Visual Basic έκδ. 6.0, Microsoft Visio 2000 (για σχεδίαση flowcharts), Sun's Java 2 Runtime και SDK έκδ. 1.3.1. Επιπλέον, συνιστάται η χρήση ενοποιημένου περιβάλλοντος ανάπτυξης (IDE), όπως το Sun Forte for Java2 έκδ. 3.0 (http://www.sun.com/forte/ffj/ce/) , ή Borland Jbuilder 6 (http://www.borland.com/).

Τρόπος βαθμολόγησης:

- Συνδυασμός ασκήσεων που δίνονται σε τακτά χρονικά διαστήματα (κάθε 1-2 εβδομάδες) και εργασία παραδοτέα στο τέλος του εξαμήνου 50%
- Τελική εξέταση με γραπτό διαγώνισμα 50%

Η παρουσία και η συμμετοχή στο μάθημα και στα εργαστήρια μπορεί να επηρεάσει ±5% του τελικού βαθμού.

Βιβλία – Παραπομπές

Στους φοιτητές για το α΄ εξάμηνο διανέμεται το βιβλίο [17] και για το β' εξάμηνο το [20]. Ωστόσο, οι φοιτητές ενθαρρύνονται να ανατρέχουν κατά περίπτωση και στις άλλες αναφορές που παρατίθενται.

1. M. Bohl, M. Rynn, Tools for structured design: an introduction to programming logic, 5th ed., Prentice-Hall, 2001, 0-13-020037-9
2. Maureen Sprankle, *Problem solving and programming concepts,* 5th ed., Prentice Hall, 2001, 0-13-022967-9
3. Ellis Horowitz, *Βασικές αρχές γλωσσών προγραμματισμού,* 2η έκδοση, Κλειδάριθμος, 1993, 960-209-190-6
4. Ravi Sethi, *Programming languages,* 2nd ed., Addison Wesley, 1996, 0-201-59065-4
5. Roger Graham, *The synergy of hardware and software,* Prentice Hall, 1998, 0-13-145617-2
6. ΥΠΕΠΘ, Ανάπτυξη εφαρμογών σε προγραμματιστικό περιβάλλον, 1999, 960-7251-23-7
7. Douglas Bell, Ian Morey, John Pugh, *The essence of program design,* Prentice Hall, 1997, 0-13-367806-7
8. Terry Winograd, *Bringing design to software,* ACM Press, 2000, 0-201-85491-0
9. Stefano Ceri, Dino Mandrioli, Licia Sbattella, *The art and craft of computing,* Addison Wesley, 1998, 0-201-87698-1
10. Robert W. Sebesta, *Concepts of Programming Languages,* 5th ed., Addison Wesley, 2001, 0-201-75295-6
11. Ravi Sethi, *Programming Languages: Concepts and Constructs,* 2nd ed., Addison Wesley, 1996, 0-201-59065-4
12. David Harel, *Algorithmics: the spirit of computing,* 2nd ed., Addison Wesley, 1992, 0-201-50401-4
13. J. Glenn Brookshear, *Computer Science, an overview,* 6th ed., Addison Wesley, 2000, 0-201-35747-X
14. Ι. Ανδρεάδης, Μια σύντομη αναδρομή στην τεχνολογία των ηλεκτρονικών υπολογιστών, *Δελτίο ΠΣΔΜ-Η,* σελ. 48-52, Ιούλιος - Αύγουστος 2001
15. Deitel & Deitel, *Java How to program,* 3rd ed., Prentice Hall, 1999, 0-13-012507-5
16. Για συγκεκριμένες γλώσσες προγραμματισμού, προτείνονται τα βιβλία:
17. Paul Sheriff, *O Paul Sheriff διδάσκει VB6,* εκδόσεις Que, 960-520-224-7

18. P. H. Winston, S. Narasimhan, *On to Java 1.2,* 2nd ed., Addison Wesley, 1998, 0-201-38598-8
19. Bruce Eckel, *Thinking in Java,* 2nd ed., HTML & PDF version downloadable from http://www.bruceEckel.com ή http://www.planetpdf.com
20. Γιώργος Λιακέας, Εισαγωγή στη Java, Κλειδάριθμος, 960-209-431-1

Χρήσιμες διευθύνσεις ιστοσελίδων στο Διαδίκτυο:

- http://olympiads.win.tue.nl/ioi - Ολυμπιάδες πληροφορικής με αλγοριθμικά προβλήματα
- http://icpc.baylor.edu/past/default.htm - Μαθητικοί διαγωνισμοί πληροφορικής που διοργανώνονται από την Association for Computing Machinery
- http://www.pi-schools.gr/greek/epps/but3-informatics.htm - Βιβλία πληροφορικής που έχουν εκδοθεί από το Παιδαγωγικό Ινστιτούτο

Σχετικές λίστες συζητήσεων (newsgroups) προσπελάσιμες μέσω του http://www.dejanews.com :

- comp.lang.basic – για τη γλώσσα Basic και τις διαλέκτους της
- comp.lang.java – για τη Java. Έχει 17 υποκατηγορίες λιστών

Γενικότερα η ομάδα newsgroups κάτω από τα comp.lang (~54 γλώσσες)

Περιγραφή μαθήματος για οδηγό σπουδών

Τεχνολογίες & Μεθοδολογίες Προγραμματισμού Ι

Το μάθημα αυτό έχει στόχο να διδάξει στους φοιτητές πώς να λύνουν προβλήματα χρησιμοποιώντας υπολογιστές και να τους εισάγει στον προγραμματισμό, ενθαρρύνοντας καλές τεχνικές ανάπτυξης. Προϋποθέτει βασικές γνώσεις οργάνωσης και λειτουργίας υπολογιστικών συστημάτων που αποκτήθηκαν στο μάθημα «Πληροφορική Ι & ΙΙ» του α΄ έτους.

Η σχεδίαση προγραμμάτων, δηλαδή ο τρόπος με τον οποίο ο προγραμματιστής δημιουργεί τη δομή του προγράμματος, διδάσκεται ανεξάρτητα από συγκεκριμένη γλώσσα προγραμματισμού. Αναλύονται διάφορα προβλήματα και αναπτύσσεται η λύση τους, εκφρασμένη έτσι ώστε να μπορεί να εκτελεστεί από υπολογιστή. Διαγράμματα ροής και ψευδοκώδικας χρησιμοποιούνται για την αρχική έκφραση του αλγορίθμου με σαφήνεια και ελεγξιμότητα. Οι φοιτητές επίσης μαθαίνουν τη δομή της μηχανής και τη δομή των προγραμμάτων κι έτσι κατανοούν την εκτέλεση των προγραμμάτων που γράφουν.

Οι φοιτητές που θα ολοκληρώσουν αυτό το μάθημα:

- θα αναπτύξουν δεξιότητα στην επίλυση προβλημάτων με υπολογιστές,
- θα γνωρίσουν το συντακτικό και τη σημασιολογία των τυπικών προγραμματιστικών δομών, όπως η ανάθεση, η επιλογή, η επανάληψη,
- θα κατανοήσουν τους βασικούς τύπους δεδομένων, όπως αριθμοί, συμβολοσειρές, και το χειρισμό τους,
- θα γνωρίσουν τη δυνατότητα ορισμού νέων τύπων δεδομένων, όπως πίνακες, με χρήση δηλωτικών εντολών, και το χειρισμό τους, και τέλος
- θα γνωρίζουν να σχεδιάζουν και υλοποιούν ευανάγνωστα και αποδοτικά προγράμματα.

Για να κατανοηθούν οι αρχές και οι τεχνικές που περιγράφονται στις διαλέξεις του μαθήματος και για να αποκτήσουν οι φοιτητές πρακτική εμπειρία, στα εργαστήρια χρησιμοποιείται η γλώσσα και το περιβάλλον ανάπτυξης της Visual Basic.

Τεχνολογίες & Μεθοδολογίες Προγραμματισμού ΙΙ

Όσο η λύση γίνεται πιο πολύπλοκη, καθίσταται αναγκαία η χρήση τεχνικών βαθμιαίας αποδόμησης κατά το πρότυπο του δομημένου προγραμματισμού και της αντικειμενοστραφούς σχεδίασης και υλοποίησης. Εξετάζεται ο μεθοδικός μετασχηματισμός των αναλυτικών προδιαγραφών σε δομημένες λύσεις. Επίσης, περιγράφεται το μοντέλο του οδηγούμενου από συμβάντα υπολογισμού σε σχέση με γραφικά περιβάλλοντα.

Εκτός από τη σύνταξη προγραμμάτων, στα εργαστήρια περιγράφεται η χρήση συντακτών κατευθυνόμενου συντακτικού (syntax-directed editors), εργαλείων εκσφαλμάτωσης (debugging tools) και προκατασκευασμένων χειριστηρίων (component-based programming).

Οι φοιτητές που θα ολοκληρώσουν αυτό το μάθημα:

- θα κατανοούν τις βασικές αρχές του δομημένου προγραμματισμού, όπως η σταδιακή εκλέπτυνση (step-wise refinement), ο ορισμός και η κλήση διαδικασιών,
- θα κατανοούν τις βασικές αρχές του αντικειμενοστραφή προγραμματισμού και σχεδίασης, όπως οι κλάσεις, η απόκρυψη πληροφορίας και η κληρονομικότητα,
- θα ελέγχουν την ορθότητα των προγραμμάτων και την καταλληλότητά τους σε σχέση με το πρόβλημα και το χρήστη,

- θα έχουν κατανοήσει τη χρήση απλών δομών δεδομένων, όπως στοίβες, σωροί, ουρές,
- θα αποκτήσουν αίσθηση της πολυπλοκότητας των λειτουργιών ενός αλγορίθμου, και τέλος
- εκτός από επιτακτικές (imperative) γλώσσες προγραμματισμού, θα γνωρίσουν τον συναρτησιακό (functional) και τον λογικό προγραμματισμό.

Για να κατανοήσουν οι φοιτητές τις αρχές, τεχνικές και μεθόδους που περιγράφονται στο μάθημα, χρησιμοποιείται η γλώσσα Java.

Course description

Programming Technology and Methodology I

This course aims at teaching students to solve problems using computers and at introducing them to programming, promoting good programming practices. It presumes that students have knowledge about the organization and operation of computer-based information systems.

Software design, ie. The way a programmer creates the structure of a program, is taught independently of a specific programming language. Algorithmic solutions to various problems are presented and analyzed and their execution on computer systems is described. The solutions are illustrated using data flow diagrams and pseudo-code stressing clarity and testability. The students get acquainted with the structure of the virtual machine that executes the programs. They are able to follow the execution of their programs and estimate their space and time requirements.

After completing this course, students will:

- develop skills in problem solving using computers,
- become acquainted the syntax and semantics of typical programming structures, such as assignment, selection and repetition,
- understand the use and representation of the basic data types, such as numbers and strings, and the set of operations which are applicable on them
- learn to define new data types, such as arrays, records and enumerations, using declaration statements; to use them effectively by understanding their internal representation, and finally
- learn to design and develop readable and efficient computer programs.

In order to exemplify the concepts and techniques that are presented at the lectures, students will attend labs where they will practice using the MS Visual Basic integrated development environment.

Τεχνολογίες & Μεθοδολογίες Προγραμματισμού II

As the problems become more complex, we need to develop their algorithmic solutions using functional decomposition techniques, such as structured and object-oriented design and development. We present the methodology of transforming the system requirements to software architectures using structured approaches. We present the event-driven model of programming and argue its suitability for the development of software with graphical user interface.

At the labs, students develop larger programs, making full use of syntax-directed editors, debugging tools. They are also introduced to the concept of component-based programming.

After completing the course, students will:

- appreciate the merits of structured programming, and will be able to apply the methodology of step-wise refinement during software design,
- understand object-oriented programming, and related concepts, such as classes, objects, information hiding and inheritance,
- be able to check thoroughly the correctness of their programs and their suitability in relation with the task to be accomplished and the prospective user,
- understand the use of data structures, such as stacks, queues and heaps,
- acquire a perception of algorithmic (time and space) complexity, and finally
- be introduced to functional and logic programming, as compared to the traditional imperative programming.

The Java programming language will be used for practice, giving the opportunity to introduce students to programming for the Internet.