Εισαγωγή στην Αλγοριθμική, στον Αντικειμενοστρεφή Προγραμματισμό ....και στη Java

ΕΙΣΑΓΩΓΗ *(απόσπασμα από την διδακτορική διατριβή μου…)*

Κατά την δεκαετία του '60 η κατασκευή λογισμικού με άμεσο προγραμματισμό, χρησιμοποιώντας ως εργαλεία τις αντίστοιχες γλώσσες μηχανής, είχε ως σκοπό να υλοποιήσει εφαρμογές λογισμικού, απλές ή σύνθετες, με ένα μη δομημένο αλλά περισσότερο προσιτό τρόπο από ότι επέτρεπαν οι γλώσσες μηχανής. Στην δεκαετία του '70 η προσέγγιση στην ανάλυση και σχεδίαση των συστημάτων λογισμικού αλλά και οι γλώσσες προγραμματισμού απέκτησαν στοιχεία δομής τα οποία δεν διέθετε ο *άμεσος προγραμματισμός και οι γλώσσες μηχανής.* Η έννοια της **δομημένης προσέγγισης** *(structured paradigm)* χαρακτήρισε τον τρόπο σκέψης στη σύλληψη και σχεδίαση του λογισμικού με έντονα τα χαρακτηριστικά των εργαλείων ανάπτυξης (γλώσσες προγραμματισμού).

Όμως η **πολυπλοκότητα** *(complexity)* τόσο στην υπολογιστική διεργασία όσο και στην διαχείριση μεγάλου όγκου δεδομένων καθώς και η αδυναμία απεικόνισης των οντοτήτων του πραγματικού κόσμου σε συστατικά λογισμικού, κατέστησε ανεπαρκή την δομημένη προσέγγιση. Η θεώρηση ότι τα δεδομένα είναι ανεξάρτητα από τις λειτουργικές μονάδες που επιδρούν σε αυτά δεν ισχύει στον πραγματικό κόσμο και ιδιαίτερα στην **επιχειρησιακή λογική** *(business logic).* Η δομημένη προσέγγιση, ως μεθοδολογικό εργαλείο στην ανάπτυξη λογισμικού, προκαλεί επίσης μια σειρά από πρακτικά προβλήματα στον προσδιορισμό των απαιτήσεων, στην διαχείριση των μοντέλων παράστασης λογισμικού, στην συντήρηση λογισμικού αλλά και στην επαναχρησιμοποίηση των συστατικών λογισμικού.

Τα προβλήματα αυτά έφεραν την **κρίση λογισμικού** (σαν όρος εμφανίστηκε το 1968), και ώθησαν στην γέννηση της **αντικειμενοστρεφούς φιλοσοφίας ή παραδείγματος** *(object-oriented philosophy, paradigm)* που άλλαξε ριζικά τον τρόπο σκέψης κατά την ανάλυση και σχεδίαση των συστημάτων λογισμικού. Σαν νέο μεθοδολογικό εργαλείο

εισήγαγε αρκετές νέες έννοιες και συμβολισμούς όπως **κλάση** *(class),* **αντικείμενο** *(object),* ***αφαίρεση*** *(abstraction),* **κληρονομικότητα** *(inheritance)* ή **γενίκευση** *(generalization),* **συσχέτιση** *(association),* **συναρμολόγηση** *(aggregation),* **ενθυλάκωση ή κελυφοποίηση** *(encapsulation)* και **πολυμορφισμό** *(polymorphism).* Η πρώτη γλώσσα που διέθετε αυτά τα χαρακτηριστικά ήταν η Simula-67 (Νορβηγία το 1967) ενώ πολλές άλλες γλώσσες έγιναν διάσημες σε διάφορες χρονικές περιόδους, όπως η ADA, Smalltalk, C++, Java, κ.ά. Η αντικειμενοστρεφής τεχνολογία λογισμικού δεν αποτελεί σε καμία περίπτωση πανάκεια, είναι ωστόσο το καλύτερο από τα εργαλεία που διαθέτουμε για να αντιμετωπίσουμε τα προβλήματα της ανάπτυξης λογισμικού.

Η ΣΗΜΑΝΤΙΚΟΤΗΤΑ ΤΗΣ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΟΥΣ ΑΝΑΠΤΥΞΗΣ

Πολλά είναι τα χαρακτηριστικά της αντικειμενοστρεφούς τεχνολογίας που την καθιστούν ικανή να αντιμετωπίσει τις αδυναμίες των παραδοσιακών διεργασιών ανάπτυξης:

- Το πρώτο σημαντικό χαρακτηριστικό είναι η **συνέπεια στη γλώσσα που χρησιμοποιείται σε όλα τα στάδια της ανάπτυξης**. Η περιγραφή του προβλήματος και της λύσης του γίνεται πλέον με τους ίδιους όρους σε όλα τα στάδια της διεργασίας ανάπτυξης. Οι όροι ***κλάσεις, αντικείμενα, μέθοδοι, χαρακτηριστικά*** και ***συμπεριφορές*** χρησιμοποιούνται στην αναπαράσταση του συστήματος σε όλες τις φάσεις της ανάπτυξης, από την Αντικειμενοστρεφή Ανάλυση *(Object Oriented Analysis - OOA)* έως την Αντικειμενοστρεφή Σχεδίαση *(Object Oriented Design - OOD)* και τον Αντικειμενοστρεφή Προγραμματισμό *(Object Oriented Programming - OOP)*. Αυτό το χαρακτηριστικό καθιστά την αντικειμενοστρεφή τεχνολογία μια φιλοσοφία αναπαράστασης και λύσης προβλημάτων που επεκτείνεται σε όλους τους κύκλους ζωής όπως ο καταρράκτης, το σπειροειδές μοντέλο και το επαναληπτικό μοντέλο.
- Η **κελυφοποίηση** των δεδομένων *και* της **συμπεριφοράς** μέσα στις κλάσεις, που σημαίνει την απόκρυψη των λεπτομερειών υλοποίησης των αντικειμένων, είναι το δεύτερο σημαντικό χαρακτηριστικό. Ο στόχος αυτού του χαρακτηριστικού είναι διττός, αφ' ενός μεν η επίτευξη της **συναρμολογησιμότητας** *(modularity),* αφετέρου δε **η απόκρυψη των πληροφοριών** *(information hiding).* Τα αντικείμενα ως ανεξάρτητες δομές, που περιλαμβάνουν από κοινού δεδομένα και δράσεις που επιδρούν σε αυτά, μπορούν πλέον εύκολα να αντιστοιχηθούν με τις οντότητες του πραγματικού κόσμου, καθώς αυτές έχουν και κατάσταση και συμπεριφορά.

- Οι σχέσεις μεταξύ των κλάσεων, όπως της συσχέτισης, κληρονομικότητας ή γενίκευσης και συναρμολόγησης, υλοποιούνται σε **επίπεδο αντικειμένων των αντιστοίχων κλάσεων** και αποτελούν εργαλεία μοντελοποίησης αλλά και βασικές πρακτικές, που εφαρμόζονται στην κατασκευή του λογισμικού.

Το πόσο σημαντική είναι η αλλαγή στον τρόπο ανάπτυξης που επιφέρει η αντικειμενοστρεφής τεχνολογία αποδεικνύεται από τον τρόπο αντιμετώπισης της πολυπλοκότητας. Η πολυπλοκότητα αφορά δύο από τα βασικά στοιχεία του λογισμικού: την υπολογιστική εργασία του λογισμικού και την διαχείριση μεγάλου όγκου δεδομένων. Για την αντιμετώπιση της πολυπλοκότητας η αντικειμενοστρεφής τεχνολογία χρησιμοποιεί την απόκρυψη των πληροφοριών και τη συναρμολογησιμότητα αλλά και διάφορα εργαλεία, όπως τα έτοιμα **συστατικά λογισμικού** *(components)* και την οπτική μοντελοποίηση για την παράσταση των μοντέλων λογισμικού με την χρήση της UML. Η μοντελοποίηση αν και αφορά πρωτίστως τη σχεδίαση, εντούτοις βοηθά στην μείωση της πολυπλοκότητας του συστήματος.

Σημαντικά και ποικιλόμορφα είναι τα πλεονεκτήματα που επιφέρει στην ανάπτυξη του λογισμικού η χρήση της αντικειμενοστρεφούς τεχνολογίας. Η υψηλή **ποιότητα του λογισμικού** είναι η θεμελιώδης αρχή της αντικειμενοστρεφούς τεχνολογίας που σημαίνει ότι το λογισμικό πρέπει να είναι **σωστό** *(correct),* **αποτελεσματικό** *(efficient),* **επαναχρησιμοποιήσιμο** *(reusable),* **επεκτάσιμο** *(extendible)* και **συντηρήσιμο** *(maintainable)*.

Η **επαναχρησιμοποίηση** *(reusability)* είναι η ικανότητα των προϊόντων λογισμικού να επαναχρησιμοποιούνται, είτε ολικά είτε τμηματικά, στην κατασκευή νέων εφαρμογών. Η αντικειμενοστρεφή τεχνολογία παρέχει επιπρόσθετους μηχανισμούς επαναχρησιμοποίησης δηλαδή τις κλάσεις, την κελυφοποίηση, την κληρονομικότητα και τον πολυμορφισμό. Υπάρχει όμως και ο ανθρώπινος παράγοντας, που παίζει σημαντικό ρόλο στην ανάπτυξη επαναχρησιμοποιούμενου λογισμικού. Δηλαδή το ερώτημα είναι πως οι κατασκευαστές θα χρησιμοποιήσουν τους προαναφερθέντες μηχανισμούς στην κατασκευή του λογισμικού. Ένα πρώτο βήμα είναι η καλύτερη κατανόηση των εννοιών αυτών, καθότι δεν υπάρχουν οδηγίες για την χρήση των αλλά και για ενιαίο τρόπο κατασκευής του λογισμικού. Στην πράξη η επαναχρησιμοποίηση λογισμικού απαιτεί πολύ προσπάθεια και πειθαρχία σε όλα τα στάδια της ανάπτυξης.

Η **επεκτασιμότητα** *(extendibility)* είναι η ευκολία με την οποία τα προϊόντα λογισμικού ανταποκρίνονται σε αλλαγές των προδιαγραφών και μπορούν να επεκταθούν. Η επέκταση σε νέες λειτουργικότητες γίνεται σχετικά εύκολα στα αντικειμενοστρεφή συστήματα με την

χρήση των μηχανισμών της κελυφοποίησης και κληρονομικότητας. Ένας σημαντικός τρόπος για την καλύτερη ανταπόκριση στις αλλαγές είναι η ανάπτυξη ανεξάρτητων κομματιών κώδικα και δομών με όσο το δυνατό λιγότερη επικοινωνία με το υπόλοιπο πρόγραμμα και ελαχιστοποίηση των διεπαφών. Όσο περισσότερες ανεξάρτητες δομές χρησιμοποιούνται στην ανάπτυξη ενός συστήματος, τόσο αυξάνεται η πιθανότητα ότι μια απλή αλλαγή θα επιδράσει μία ή έστω λίγες δομές και δεν θα προκαλέσει μία αλυσιδωτή αντίδραση σε ολόκληρο το σύστημα.

Η **συντηρησιμότητα** *(maintainability)* είναι η δυνατότητα αλλαγών στον κώδικα συμπεριλαμβανομένης και της διόρθωσης λαθών. Η μη επαρκής δόμηση του κώδικα ήταν το κύριο χαρακτηριστικό των παραδοσιακών διαδικασιών ανάπτυξης, με αποτέλεσμα να γίνεται δύσκολο το έργο των αλλαγών και διορθώσεων. Η αντικειμενοστρεφής τεχνολογία, με την βοήθεια του μηχανισμού της κελυφοποίησης, συντελεί στην συντηρησιμότητα του λογισμικού, χωρίς αυτό να σημαίνει ότι μπορεί να επιτευχθεί αυτόματα. Η κατασκευή λογισμικού εξαρτάται άμεσα από τον ανθρώπινο παράγοντα και ιδιαίτερα η κατασκευή λογισμικού που θα ανταποκρίνεται με ευκολία στις μεταβολές και διορθώσεις είναι μια επίπονη και διαρκής προσπάθεια την οποία πρέπει να καταβάλουν οι κατασκευαστές σε όλα τα στάδια της ανάπτυξης.

Η αντικειμενοστεφής προσέγγιση είναι μια καλά ορισμένη τεχνολογία που ακόμα δεν έχει **τυποποιηθεί** *(standardization).* Υπόσχεται την ανάπτυξη ποιοτικού λογισμικού, δηλαδή λογισμικού ικανού να αντιμετωπίσει τις ανάγκες, που δημιουργεί η σύγχρονη επιχειρησιακή λογική, όπως η εύκολη ανταπόκριση στις αλλαγές, η εύκολη επεκτασιμότητα, η μείωση του κόστους συντήρησης του κώδικα και η επαναχρησιμοποίηση κώδικα ή δομών. Στην βιβλιογραφία αναφέρονται και άλλα οφέλη όπως η ***μείωση του χρόνου ανάπτυξης, η αύξηση της παραγωγικότητας, η απλότητα του κώδικα, η αύξηση της αυτοπεποίθησης και φρονήματος των κατασκευαστών, η καλύτερη μοντελοποίηση και η καλύτερη κατανόηση του συστήματος*.** Για την εκμετάλλευση όλων αυτών των πλεονεκτημάτων απαιτείται η ριζική αλλαγή κατεύθυνσης στον τρόπο ανάλυσης – σχεδίασης - υλοποίησης, η κατανόηση του συνόλου των αρχών και η ορθή χρήση των εννοιών - μηχανισμών της αντικειμενοστρέφειας αλλά και η διαρκής εκπαίδευση τόσο των διαχειριστών όσο και των κατασκευαστών των έργων λογισμικού. Η αντικειμενοστρεφής τεχνολογία είναι ένα καλύτερο εργαλείο για την λύση των προβλημάτων, δεν είναι η ίδια η λύση των προβλημάτων. Πολλά από τα προβλήματα που υπήρχαν στην ανάπτυξη του λογισμικού εξακολουθούν να υπάρχουν και στην αντικειμενοστρεφή τεχνολογία, ενδεχομένως σε μικρότερο βαθμό. Ο διαφορετικός και πολλές φορές μη ενδεδειγμένος τρόπος χρήσης των

εννοιών – μηχανισμών της αντικειμενοστρεφούς τεχνολογίας, λόγω και της έλλειψης οδηγιών για την σωστή χρήση των, επιφέρει διαφορετικά αποτελέσματα και πολλές διαφωνίες μεταξύ των εμπλεκομένων ερευνητών και κατασκευαστών.

Εισαγωγή στην Αλγοριθμική και τον Προγραμματισμό

Καθημερινά καλούμαστε να επιλύσουμε προβλήματα. Βέβαια, αυτό δεν είναι πάντα εύκολο για πολλούς και διαφορετικούς λόγους. Η αλγοριθμική είναι η μεθοδολογία προσέγγισης ενός προβλήματος. Δίνει έμφαση στον τρόπο επίλυσης του προβλήματος, ανεξάρτητα από το χώρο από τον οποίο προέρχεται το πρόβλημα. Για την επίλυση ενός προβλήματος θα πρέπει πρώτα να κατανοήσουμε το πρόβλημα, να κατανοήσουμε τα δεδομένα και τα ζητούμενα και έπειτα να σκεφτούμε τις ενέργειες που θα εκτελέσουμε για την επίλυσή του.

Αλγόριθμος είναι ένα σύνολο από λειτουργίες, οι οποίες περιγράφουν βήμα προς βήμα τις ενέργειες που πρέπει να εκτελεστούν για να ολοκληρωθεί μια εργασία ή να επιλυθεί ένα πρόβλημα με πεπερασμένο αριθμό ενεργειών. Ο αλγόριθμος πρέπει να επιλύει το πρόβλημα με καθορισμένα βήματα, να είναι απλός και κατανοητός και φυσικά να τελειώνει.

Ας αναλύσουμε τις ανωτέρω έννοιες και ορισμούς από μια διαφορετική οπτική. Από το σχολείο μάθαμε να λύνουμε ένα πρόβλημα, με την θετική σκέψη, προσπαθώντας να ξεκαθαρίσουμε:

1) ποια είναι τα δεδομένα του προβλήματος (τι μας δίνεται),
2) τι πρέπει να υπολογίσουμε (τι μας ζητάει το πρόβλημα) και
3) τι τύπους (παραστάσεις, σχέσεις – εκφράσεις) θα υλοποιήσουμε για να βρούμε το αποτέλεσμα.

Για παράδειγμα σκεφτείτε την παρακάτω εκφώνηση*:" Δίνεται η βάση ενός τριγώνου β=2 και το ύψος του υ=4. Ζητείται το Εμβαδόν του τριγώνου."* Αμέσως σκεφτόμαστε, ακολουθώντας τα τρία ανωτέρω βήματα: 1) μας δίνεται η βάση και το ύψος του τριγώνου, 2) μας ζητά να υπολογίσουμε το εμβαδόν του και 3) σκεφτόμαστε τον τύπο υπολογισμού που είναι:

Εμβαδόν = Βάση x Ύψος / 2. Άρα στον τύπο, αντικαθιστώντας τη Βάση με 2, το Ύψος με 4 και διαιρώντας δια 2, βρίσκουμε το Εμβαδόν = 4. Στο παράδειγμα αυτό ο αλγόριθμος ήταν πολύ εύκολος και τα βήματα (ή ενέργειες) που κάναμε για να το λύσουμε ήταν λίγα και απλά. Τις πιο πολλές φορές τα βήματα που πρέπει να κάνουμε είναι πολλά και δύσκολα και απαιτούν πέρα της **θετικής σκέψης** και την **λογική**. Για παράδειγμα, σκεφτείτε τον υπολογισμό του συνολικού ποσού πληρωμής μιας αγοράς, πριν τον υπολογισμό του ποσού ΦΠΑ για αυτή την αγορά. Κάτι τέτοιο δεν μπορεί να γίνει, γιατί πρέπει πρώτα να υπολογίσουμε το ποσό ΦΠΑ και μετά το συνολικό ποσό πληρωμής (αρχική τιμή + ποσό

ΦΠΑ). Άρα, ας τροποποιήσουμε λίγο τον ορισμό του αλγορίθμου, κάνοντας τον πιο απλό και πιο κατανοητό:

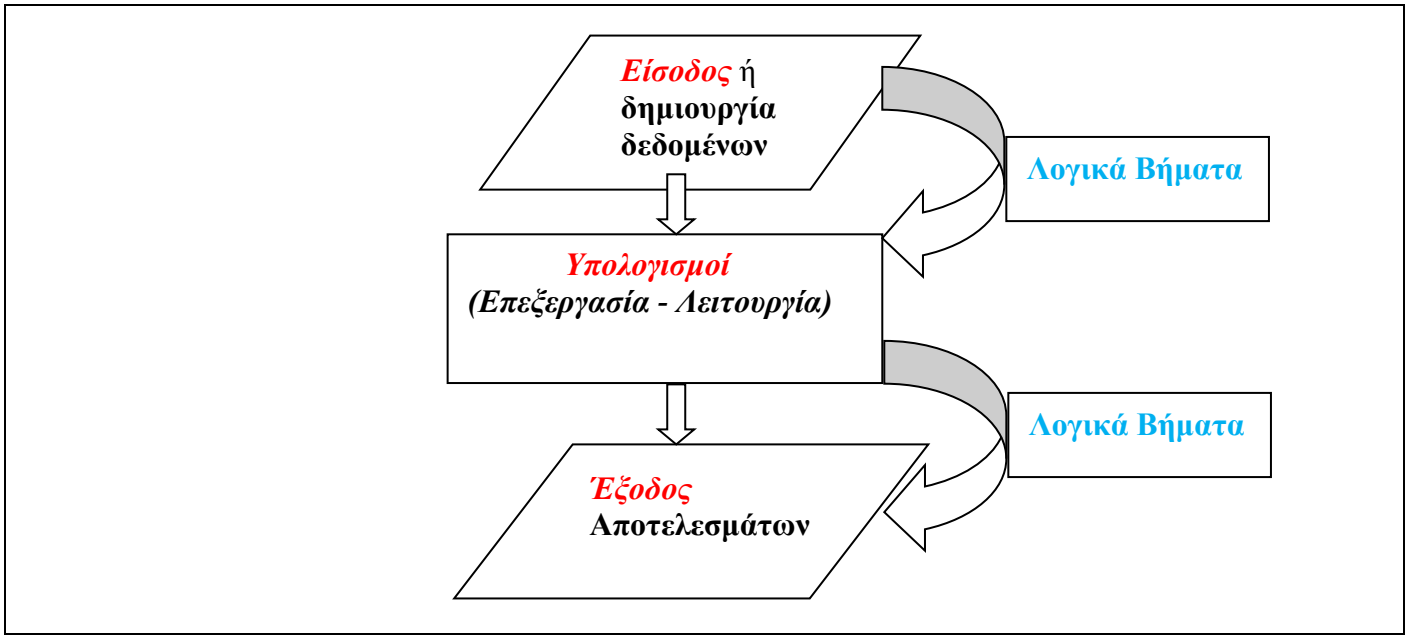
Αλγόριθμος είναι ένα σύνολο από λογικά βήματα και αντίστοιχες λειτουργίες που πρέπει να εκτελεστούν για να ολοκληρωθεί μια εργασία ή να επιλυθεί ένα πρόβλημα.

Στο παράδειγμα του εμβαδού του τριγώνου τα λογικά βήματα που κάναμε ήταν πολύ απλά και λίγα:

1) ξεχωρίσαμε τα δεδομένα – οι τιμές στη βάση και το ύψος (***είσοδος ή δημιουργία δεδομένων***),
2) εφαρμόσαμε τον τύπο του Εμβαδού του τριγώνου (E=B x Y / 2) και υπολογίσαμε το εμβαδό του, αντικαθιστώντας στον τύπο τις τιμές (***υπολογισμός***) και
3) γράψαμε το αποτέλεσμα (***εμφάνιση του αποτελέσματος***).

Με **bold** και *italics* είναι οι αντίστοιχες λειτουργίες – ή βήματα – στον προγραμματισμό. Δηλαδή, για να λύσουμε ένα πρόβλημα θα πρέπει: να βρούμε τα δεδομένα που μας δίνονται (αλλιώς θα πρέπει να τα δημιουργήσουμε), να βρούμε τις σχέσεις που χρειάζονται και να τις υπολογίσουμε και τέλος να εμφανίσουμε τα αποτελέσματα. Τι μεσολαβεί μεταξύ των τριών αυτών σταδίων; **Τα λογικά βήματα που πρέπει να κάνουμε**. ***Τα τρία στάδια απεικονίζουν τη χρήση του υπολογιστή (από αυτές τις συσκευές αποτελείται) και τα λογικά βήματα (αλγοριθμική) που θα μας πάνε από το ένα στάδιο στο άλλο για την επίλυση ενός προβλήματος στον προγραμματισμό.***

Τρόπος επίλυσης προβλημάτων στον Προγραμματισμό :



Περιγραφή ή αναπαράσταση Αλγορίθμων - Προγραμματισμός

Η περιγραφή ή αναπαράσταση ενός αλγορίθμου γίνεται με:

1) την φυσική γλώσσα,
2) τα διαγράμματα ροής,
3) τον ψευδοκώδικα, και
4) τις γλώσσες προγραμματισμού

Το **διάγραμμα ροής** *(flowchart)* αναπαριστά έναν αλγόριθμο, δείχνοντας τα βήματα ως κουτάκια (γραφικά) διαφόρων ειδών που συνδέονται μεταξύ τους με βέλη (όπως αυτό στην προηγούμενη σελίδα). Αυτή η διαγραμματική παρουσίαση μπορεί να δώσει λύση βήμα προς βήμα σε ένα πρόβλημα. Ο **ψευδοκώδικας** μοιάζει με μια τυπική γλώσσα προγραμματισμού αλλά περιλαμβάνει μεγαλύτερη ελευθερία ως προς τους συντακτικούς κανόνες.

Οι **γλώσσες προγραμματισμού** επιτρέπουν την δημιουργία προγραμμάτων για την υλοποίηση αλγορίθμων από τα υπολογιστικά συστήματα. Με τις γλώσσες προγραμματισμού υλοποιούνται τα στοιχεία των αλγορίθμων, όπως βήματα (εντολές), σημεία απόφασης ή διακλάδωσης (αν..τότε..), επαναλήψεις (για όσο…επανάλαβε..), κλπ. Οι εντολές είναι του τύπου διάβασε μια τιμή π.χ. από το πληκτρολόγιο, υπολόγισε π.χ. το εμβαδό του τριγώνου στον τύπο, εμφάνισε το αποτέλεσμα, κλπ. Οι γλώσσες προγραμματισμού διακρίνονται σε χαμηλού και υψηλού επιπέδου και έχουν τα χαρακτηριστικά κάθε φυσικής γλώσσας δηλαδή αλφάβητο, λεξιλόγιο και συντακτικό. Κάθε πρόγραμμα γράφεται σε ένα κατανοητό σε μας τρόπο και μετά μεταγλωττίζεται ώστε να εκτελεστεί από τον υπολογιστή. Εμβάθυνση στις έννοιες αυτές θα κάνουμε παρακάτω.

Ο προγραμματισμός βοηθά στην επίλυση συνθέτων αλγορίθμων. Αν σκεφτούμε το παράδειγμα της εύρεσης του εμβαδού ενός τριγώνου, τότε η συγγραφή ενός προγράμματος για την επίλυση του προβλήματος σε μια οποιαδήποτε γλώσσα προγραμματισμού θα ήταν πιο κουραστική και σχεδόν ανώφελη σε σχέση με τον υπολογισμό του με μολύβι και χαρτί. Όμως σκεφτείτε κάποιον δύσκολο αλγόριθμο που μάλιστα θα υλοποιείται πολλές φορές με διαφορετικά δεδομένα. Για παράδειγμα τον υπολογισμό της μισθοδοσίας, την εκτέλεση τραπεζικών συναλλαγών, κλπ. Άρα ο προγραμματισμός, υλοποιούμενος μέσα από μια γλώσσα προγραμματισμού, είναι αυτός που χρησιμοποιείται για την αναπαράσταση ή περιγραφή συνθέτων και δύσκολων αλγορίθμων. Ο προγραμματισμός δεν αποκλείει την χρήση αλλά και την βοήθεια που παρέχουν τα διαγράμματα ροής και ο ψευδοκώδικας.
Πολλές είναι οι γλώσσες προγραμματισμού. Εμείς θα ασχοληθούμε με την εκμάθηση του προγραμματισμού μέσα από την αντικειμενοστρεφή γλώσσα προγραμματισμού JAVA.

Εισαγωγή στη Java

Η γλώσσα προγραμματισμού κυκλοφόρησε σε αρχική μορφή το 1991. Η πρώτη έκδοση JDK 1.0 (Java Development Kit) κυκλοφόρησε το 1996. Σήμερα κυκλοφορεί η έκδοση Java 7.0 και με αυτή την έκδοση θα δουλέψουμε στα εργαστήρια. Αποτελεί μία πλατφόρμα προγραμματισμού ανεξάρτητη από τα διαφορετικά λειτουργικά συστήματα και για τον λόγο αυτό κυριάρχησε στις εφαρμογές διαδικτύου, αλλά και σε άλλες εφαρμογές πληροφορικής. Επιπλέον είναι απλή, αντικειμενοστρεφής και σχετικά γρήγορη (ως προς την εκτέλεση των εφαρμογών).

Τα πλεονεκτήματα της Java είναι η ασφάλεια των εφαρμογών της αλλά και η ανεξαρτησία της από τα διαφορετικά λειτουργικά συστήματα. Αυτά τα δύο μεγάλα πλεονεκτήματα της οφείλονται στον Java bytecode, ο οποίος είναι ο ενδιάμεσος κώδικας που παράγεται κατά την **μεταγλώττιση** του προγράμματος. Ο κώδικας αυτός δεν είναι απευθείας εκτελέσιμος αλλά **εκτελείται** από την εικονική μηχανή Java, τον **διερμηνέα** της γλώσσας (Java Virtual Machine - JVM). Με την δομή αυτή επιτυγχάνεται η ασφαλής διακίνηση των εφαρμογών μέσα στο διαδίκτυο αλλά και η εκτέλεση των σε διαφορετικά λειτουργικά συστήματα, αρκεί να υπάρχει ο διερμηνέας της Java στα συστήματα αυτά.

Το εξάμηνο αυτό θα μάθουμε να χρησιμοποιούμε την Java στην επίλυση προβλημάτων προγραμματισμού, ξεκινώντας από τα πλέον απλά δηλαδή, την εκτέλεση αριθμητικών πράξεων και θα επεκταθούμε σε σύνθετες έννοιες, όπως των κλάσεων, αντικειμένων, μεθόδων και όλων εκείνων των εννοιών που χρησιμοποιούνται για την επίλυση προγραμματιστικών προβλημάτων.

Το πρόγραμμα στη Java είναι αρχείο κειμένου *(πηγαίος κώδικας – source code)*, που γράφεται σε απλούς επεξεργαστές κειμένου όπως Notepad, TextPad ή σε ειδικά περιβάλλοντα όπως JCreator, NetBeans, Eclipse, κλπ. Το πρόγραμμα πρώτα μεταγλωττίζεται με την κλήση του **μεταγλωττιστή *(compiler)*** *(παραγωγή ενδιάμεσου κώδικα - bytecode)* και στη συνέχεια εκτελείται με την κλήση του **διερμηνέα *(interpreter)*** *(JVM)* στη γραμμή εντολών ή μέσα από τα περιβάλλοντα (λεπτομέρειες παρακάτω). Η σύνταξη των προγραμμάτων πρέπει να γίνεται με ιδιαίτερη προσοχή.

Οι εργασίες εγκατάστασης της Java

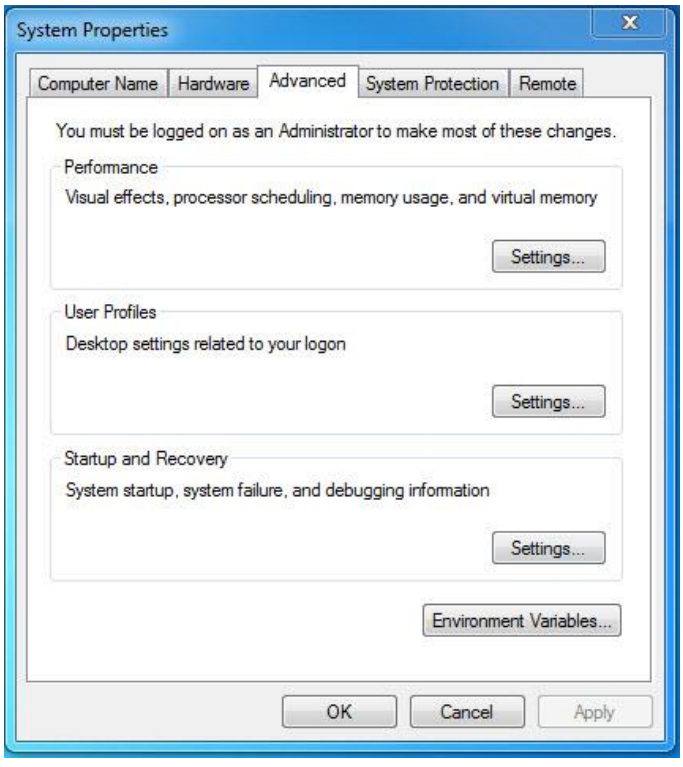
Το Java Development Kit *(JDK)* είναι μια συλλογή προγραμμάτων που μπορούμε να 'κατεβάσουμε' χωρίς κόστος από την ιστοσελίδα της εταιρίας Oracle. Πηγαίνουμε στην ιστοσελίδα της εταιρίας: www.oracle.com/technetwork/java/javase/downloads/index.html

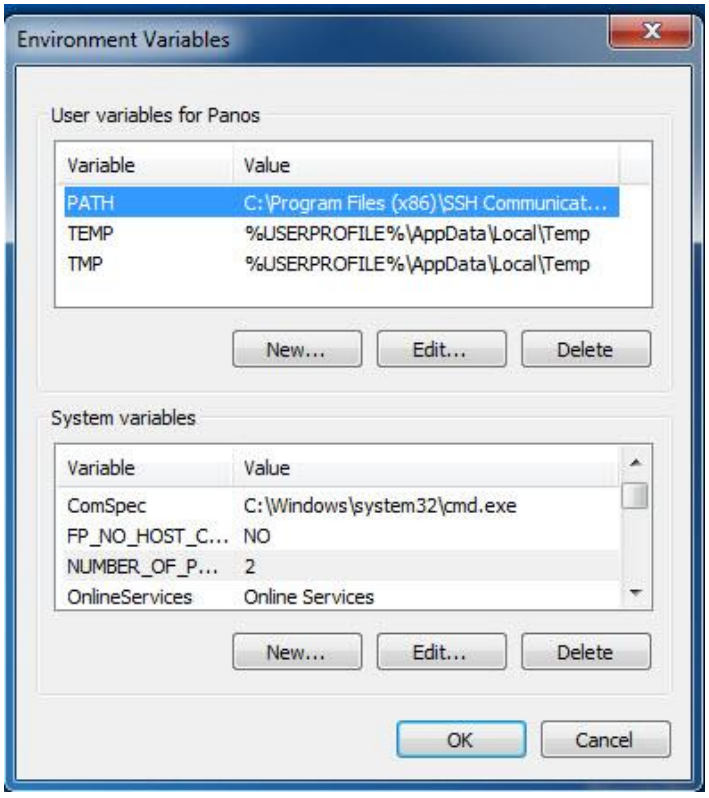
από όπου 'κατεβάζουμε' την τελευταία έκδοση της (π.χ. Java Platform (JDK) 7u7 για windows). Με την εγκατάσταση δημιουργείται ο φάκελος **C:\Program Files\Java\jdk1.7.0_02** όπου τοποθετούνται τα αρχεία της Java στους επί μέρους φακέλους bin, lib, κλπ. Οι φάκελοι bin και lib είναι πολύ σημαντικοί (περιέχουν τον μεταγλωττιστή, διερμηνέα, και τις βιβλιοθήκες των έτοιμων κλάσεων). Θα πρέπει το σύστημα να ξέρει τη διαδρομή *(path)* ώστε να βρίσκει τους δύο αυτούς φακέλους και να εκτελεί τα αντίστοιχα προγράμματα.

Ορισμός της διαδρομής *(path)* - Windows 7

- Ανοίγουμε τον πίνακα ελέγχου - *Control Panel,* επιλέγουμε **System and Security**
- Επιλέγουμε **System ->** από το μενού αριστερά **Advanced System Settings** -> και τέλος **Enviromental variables**
- Στο παράθυρο **User Variables list,** επιλέγουμε **New ->** προσθέτουμε τη λέξη **Path**, και στο δεύτερο παράθυρο πληκτρολογούμε τη διαδρομή (π.χ.):

  C:\Program Files\Java\jdk1.7.0_02\bin
- Πατάμε **OK** για να τελειώσουμε την εργασία.

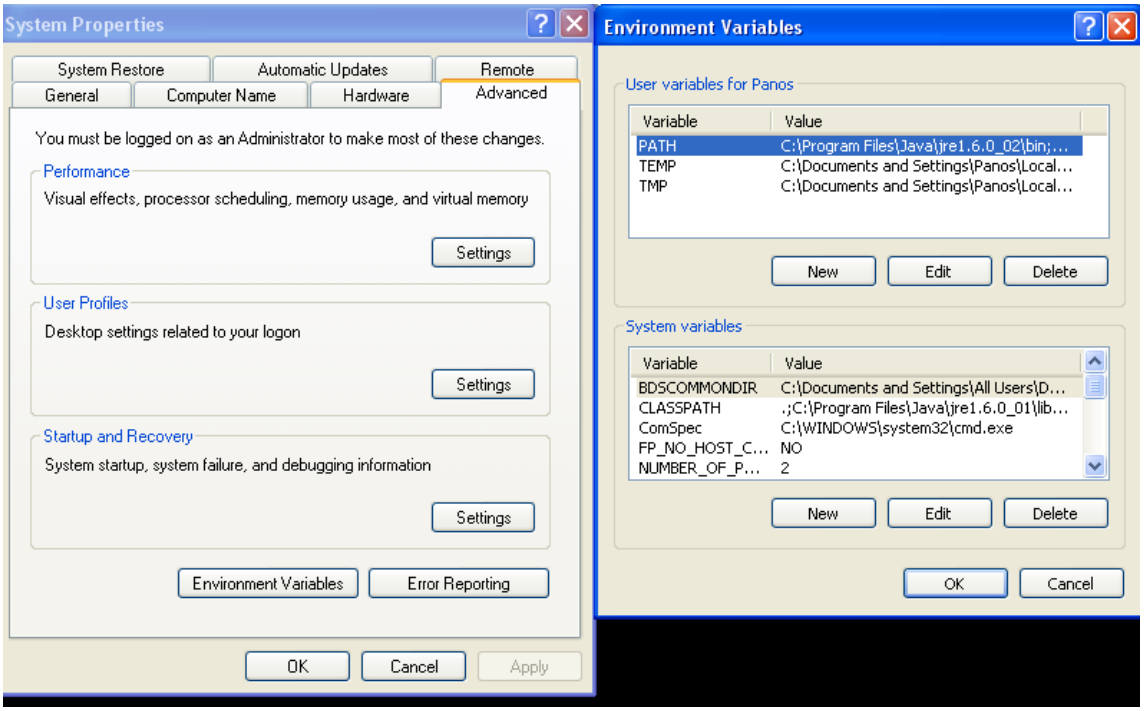




Ορισμός της διαδρομής *(path)* - Windows XP

- Ανοίγουμε τον πίνακα ελέγχου - *Control Panel,* επιλέγουμε **System**

- Επιλέγουμε την σελίδα για **προχωρημένους - *advanced*** και το κουμπί **Μεταβλητές περιβάλλοντος - *Environment Variables***
- Στο παράθυρο **System Variables list,** επιλέγουμε **Path,** και μετά το κουμπί **Edit.**



- Σε ένα μικρότερο παράθυρο που εμφανίζεται πληκτρολογούμε τη διαδρομή (π.χ.): **C:\Program Files\Java\jdk1.7.0_02\bin;**
- Πατάμε **OK** για να τελειώσουμε την εργασία

Ορισμός της διαδρομής *(path)* (σε παράθυρο - Dos)

Ορίζουμε τη διαδρομή με την εντολή:

path C:\Program Files\Java\jdk1.7.0_02\bin

Το πρώτο μας Πρόγραμμα

Το παρακάτω απλό πρόγραμμα θα εμφανίσει στην οθόνη το μήνυμα : Hello Java.

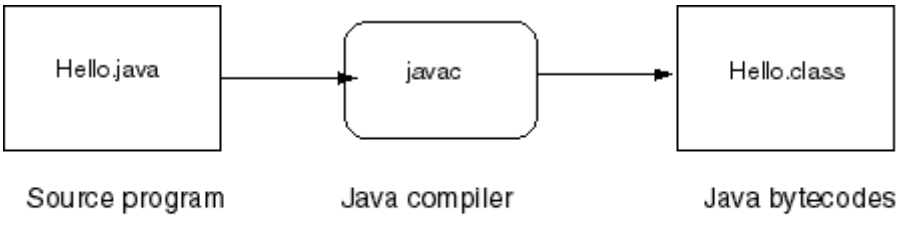
```
/*
 Αυτό είναι ένα απλό πρόγραμμα στη Java
 Το όνομα του αρχείου είναι Hello.java
*/

class Hello
{
 // Το πρόγραμμα αρχίζει με μία κλήση στη main()

  public static void main(String args[])
  {
    System.out.println("Hello Java");
  }
}
```

Μεταγλώττιση και εκτέλεση στο Dos

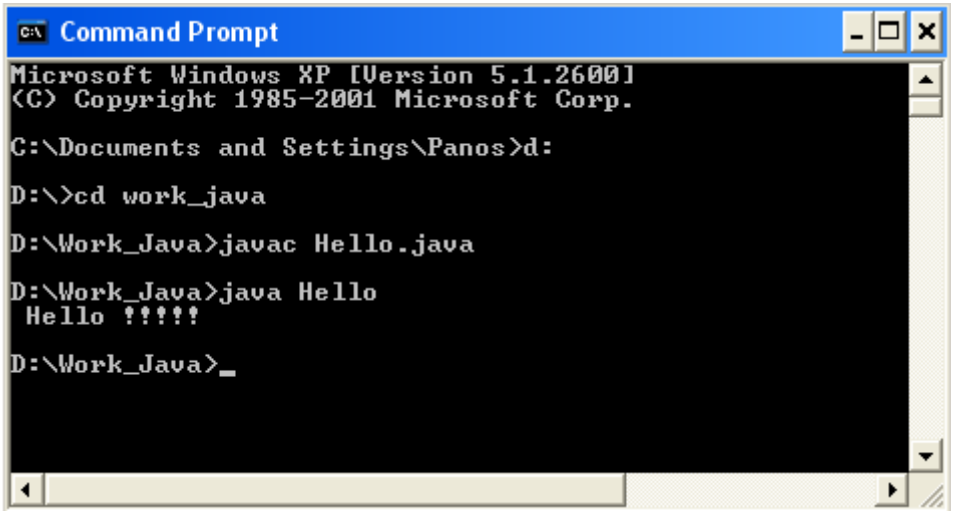
1) Γράφουμε το πρόγραμμα σε ένα κειμενογράφο, π.χ. Notepad
2) Αποθηκεύουμε το πηγαίο πρόγραμμα με το όνομα **Hello.java**
3) Το πηγαίο πρόγραμμα πρέπει πρώτα να μεταγλωττισθεί, ώστε να ελεγχθεί συντακτικά, και η εργασία αυτή γίνεται με τον μεταγλωττιστή της java (compiler) - **javac**. Για να εκτελεσθεί ο javac πρέπει προηγουμένως να έχει ορισθεί η πλήρης διαδρομή *(full path name)* (όπως παραπάνω).
4) Αν το όνομα του πηγαίου προγράμματος είναι Hello.java, τότε η εντολή για την μεταγλώττιση του προγράμματος θα είναι: **C:\>javac Hello.java**
5) Ο μεταγλωττιστής παράγει το αρχείο: **Hello.class.** Το αρχείο αυτό δεν είναι απ' ευθείας εκτελέσιμο *(executable),* αλλά ένας ενδιάμεσος κώδικας – ο bytecode – που εκτελείται από τον διερμηνέα της java *(Interpreter).*



Java Program Translation

6. Για να εκτελέσουμε το πρόγραμμα καλούμε τον διερμηνέα της java –(Interpreter) τον java με την παρακάτω εντολή: **C:\>java Hello.** Ο διερμηνέας της java (Java Virtual

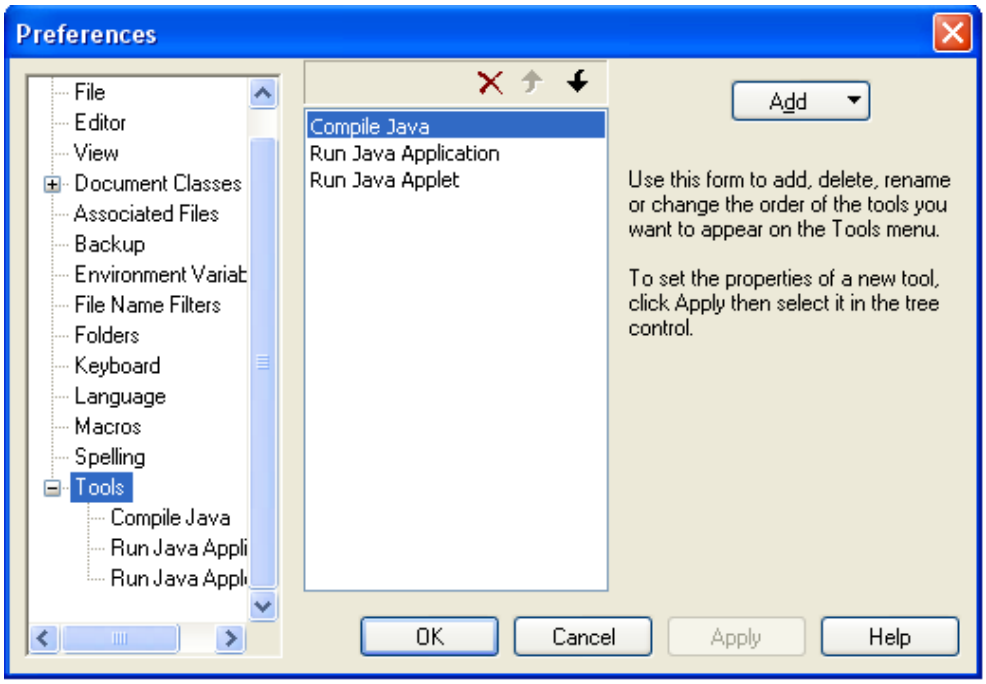
Machine - JVM) εκτελεί τον bytecode – κώδικα. Ο κώδικας αυτός μπορεί να μεταφερθεί σε οποιοδήποτε λειτουργικό σύστημα αρκεί να περιλαμβάνει τον αντίστοιχο JVM.



Η δυνατότητα εκτέλεσης των bytecode – προγραμμάτων της java σε διαφορετικά λειτουργικά συστήματα την έχει κάνει διάσημη. Τα προγράμματα διακινούνται και εκτελούνται μέσα στο διαδίκτυο. Η εκτέλεση των προγραμμάτων γίνεται από τους πλοηγούς του Internet (Web Browsers). Τα προγράμματα αυτά ονομάζονται applets.

Μεταγλώττιση και εκτέλεση στο TextPad

1) 'Κατεβάζουμε ' και εγκαθιστούμε τον TextPad: http://www.textpad.com
2) Επιλέγουμε **Configure** και μετά **Preferences**
3) Επιλέγουμε Tools και πατάμε το κουμπί **Add** και μετά να επιλέξουμε **Java SDK Commands.**
4) Αν τα ορίσαμε σωστά θα φαίνεται μια τέτοια περίπου εικόνα:



5. Πατάμε το κουμπί **OK** και γράφουμε στον κειμενογράφο το πρώτο μας πρόγραμμα. Η ανωτέρω ρύθμιση γίνεται μία φορά την πρώτη που θα δουλέψουμε με τον TextPad.
6. Αποθηκεύουμε το πρόγραμμα με το όνομα **Hello.java**
7. Μεταγλωττίζουμε (πατώντας Ctrl + 1) και εκτελούμε το πρόγραμμα (πατώντας Ctrl + 2). Εννοείται ότι θα δημιουργηθεί και το αρχείο **Hello.class.** Τέτοιες εργασίες θα εκτελούμε συνέχεια στο εργαστήριο.

Η γενική δομή του προγράμματος

```
/*
 <σχόλια>
*/
import  <java βιβλιοθήκες - Packages>;
// <σχόλια>

public class <όνομα – κλάσης>  {
<Δηλώσεις Μεταβλητών>
<Δηλώσεις Μεθόδων>

public static void main(String[] args)  {
        <εντολές προγράμματος>
}
 }
```

- Κάθε πρόγραμμα είναι μια μονάδα μεταγλώττισης
- Το πρόγραμμα της Java είναι ένα αρχείο – κειμένου που περιέχει μία ή περισσότερες Κλάσεις
- Κάθε Κλάση περιέχει τις δικές της εντολές προγράμματος
- Στη χρήση των ονομάτων πρέπει να λαμβάνεται υπόψη η διαφορά των κεφαλαίων και μικρών γραμμάτων.

Μία δεύτερη ματιά στο πρόγραμμα

- Σχόλια προγράμματος πολλαπλών γραμμών. Αρχίζουν με το σύμβολο /* και τελειώνουν με το σύμβολο */. Τα σχόλια παραβλέπονται από τον μεταγλωττιστή. Π.χ.

```
/*
 Αυτό είναι ένα απλό πρόγραμμα στη Java
 Το όνομα του αρχείου είναι Hello.java
*/
```

- Ορισμός της κλάσης του προγράμματος με την προσβασιμότητα και το όνομα της. Η κλάση αρχίζει με το άγκιστρο **{** και κλείνει επίσης με το άγκιστρο **}**. Π.χ. **class** Hello {
- Σχόλιο προγράμματος απλής γραμμής. Αρχίζει με τα σύμβολα **//** και τελειώνει στο τέλος της γραμμής. Π.χ. // Το πρόγραμμα αρχίζει με μία κλήση στη main()
- Εντολή εκκίνησης της εκτέλεσης του προγράμματος. Κάθε πρόγραμμα στη java καλεί την μέθοδο main(), **public static void main(String args[])**
- **public,** καθορίζει ότι το πρόγραμμα θα έχει πρόσβαση στην κλάση – μέλος (καθολική ορατότητα). Το αντίθετο συμβαίνει με την **private,** η οποία απαγορεύει την πρόσβαση στον κώδικα της κλάσης – μέλους (τοπική ορατότητα).
- **static,** επιτρέπει στην main() να κληθεί προς εκτέλεση πριν δημιουργηθεί κάποιο αντικείμενο στην κλάση – μέλος
- **void,** δηλώνει στον μεταγλωττιστή ότι η main() δεν επιστρέφει κάποια τιμή
- **String args[]** ή **String[] args,** ορίζει ένα πίνακα (array) παραμέτρων με όνομα args (μπορεί να έχει και άλλο όνομα) με τιμές της κλάσης String. Η args θα πάρει τιμές κατά την εκτέλεση του προγράμματος (όχι βέβαια στο παράδειγμά μας).
- **main(),** είναι η μέθοδος που καλείται όταν μία εφαρμογή java ξεκινά. Το σώμα της περιέχεται μέσα σε άγκιστρα { }. Η main() είναι το σημείο εκκίνησης του διερμηνέα. Μία εφαρμογή μπορεί να έχει δεκάδες κλάσεις αλλά μία από αυτές πρέπει να περιέχει την main().
- Η τελευταία εντολή εμφανίζει στην οθόνη το μήνυμα χρησιμοποιώντας την μέθοδο **println()** και τις **System.out** που ορίζουν, η μεν πρώτη μία κλάση που παρέχει

πρόσβαση στα μέσα του Συστήματος, ενώ η δεύτερη την δέσμη εξόδου στην οθόνη. System.out.println("Hello Java");

- Η εντολή τελειώνει με το σύμβολο semicolon (**;**). Όλες οι εντολές της java τελειώνουν με το σύμβολο αυτό.
- Το πρώτο άγκιστρο κλείνει την main(), ενώ το δεύτερο την κλάση Hello.
- Όταν το πρόγραμμα είναι μικρό περιέχει συνήθως μία κλάση. Στην περίπτωση αυτή θα πρέπει το όνομα του πηγαίου κώδικα να είναι απόλυτα ίδιο με το όνομα της κλάσης.
- Προσέξτε για μία ακόμη φορά τον τρόπο σύνταξης των στοιχείων του προγράμματος καθώς η java τα λαμβάνει σοβαρά υπόψη και επιστρέφει λάθος αν δεν ακολουθήσουμε τους κανόνες σύνταξης. Αν κάνουμε κάποιο συντακτικό λάθος θα το διορθώσουμε χρησιμοποιώντας ξανά τον κειμενογράφο.

Ο κύκλος Γράψε - διόρθωσε, Μεταγλώττισε, Εκτέλεσε..

1. Γράψε το πρόγραμμα στον κειμενογράφο ή στο περιβάλλον
2. Αποθήκευσε το πρόγραμμα με την εντολή Save As
3. Μεταγλώττισε το πρόγραμμα με την εντολή javac ή μέσα στο περιβάλλον
4. Αν υπάρχουν συντακτικά λάθη πηγαίνουμε στο βήμα 1 για διορθώσεις.
5. Εκτέλεσε το πρόγραμμα με την εντολή java ή μέσα στο περιβάλλον
6. Αν υπάρχουν λάθη πηγαίνουμε στο βήμα 1, ενώ αν εκτελείται σωστά τότε τελειώνουμε

Παραλλαγή του προγράμματος Hello

Στην παραλλαγή αυτή (που θα εκτελεσθεί στο Dos - console) θα περάσουμε με παράμετρο το δεύτερο συνθετικό όνομα της εμφάνισης του μηνύματος. Προσέξτε την σύνταξη της args. Η πρώτη παράμετρος κατέχει την θέση 0 στη λίστα των παραμέτρων.

```
/*
 Αυτό είναι ένα απλό πρόγραμμα στη Java
 Το όνομα του αρχείου είναι Hello.java
*/

class Hello {

 // Το πρόγραμμα αρχίζει με μία κλήση στη main()
```

```
public static void main(String args[])
  {
    System.out.println("Hello " + args[0]);
  }
}
```

Αν εκτελέσουμε το πρόγραμμα θα πάρουμε το αποτέλεσμα:

```
C:\>javac Hello.java
C:\>java Hello Java   (πέρασμα της λέξης Java σαν πρώτη παράμερο)
Hello Java
```