ΚΕΦΑΛΑΙΟ 1ο

Περί Προγραμματισμού και Γλωσσών Προγραμματισμού

Προγράμματα και Λειτουργικά Συστήματα

Οι ηλεκτρονικοί υπολογιστές είναι ηλεκτρονικές συσκευές (όπως είναι και ένα ραδιόφωνο) που επιτρέπουν την επέκταση της λειτουργικότητάς τους. Το ραδιόφωνο έχει συγκεκριμένες και προκαθορισμένες δυνατότητες. Μπορεί να επιλέγει ραδιοφωνική συχνότητα (και κατά συνέπεια ραδιοφωνικό σταθμό) και να ρυθμίζει την ένταση του ήχου. Για να το πετύχει αυτό, το ραδιόφωνο διαθέτει δύο, αντίστοιχα, χειριστήρια. Σε αντίθεση με το ραδιόφωνο, ο ηλεκτρονικός υπολογιστής επιτρέπει την προσθήκη δυνατοτήτων που εξυπηρετούν διαφορετικούς σκοπούς. Για παράδειγμα, μπορούμε να προσθέσουμε: ένα πρόγραμμα που να υπολογίζει τη μισθοδοσία του προσωπικού μιας επιχείρησης, ένα πρόγραμμα με το οποίο να δημιουργούμε αρχιτεκτονικά διαγράμματα (σχέδια για τη μορφή που θα έχουν κάποια κτίρια) και ένα πρόγραμμα για να γράφουμε και να μορφοποιούμε κείμενα. Ένα τέτοιο πρόγραμμα (επεξεργαστή κειμένου) χρησιμοποίησε και ο συγγραφέας του παρόντος βιβλίου για να φτάσει στο αποτέλεσμα που τώρα εσείς διαβάζετε.

Το βασικό συμπέρασμα από τα παραπάνω είναι ότι οι ηλεκτρονικοί υπολογιστές δεν κάνουν μόνο μια εργασία (ή περισσότερες) προκαθορισμένες ερ-

γασίες αλλά μπορούν να επεκτείνουν τη χρησιμότητά τους. Η επέκταση αυτή επιτυγχάνεται με την προσθήκη προγραμμάτων. Το ερώτημα που προκύπτει είναι τι είναι αυτά τα προγράμματα και πώς προστίθενται στον ηλεκτρονικό υπολογιστή μας. Πριν απαντήσουμε σε αυτό θα αναφερθούμε στα χειριστήρια και στις συσκευές που διαθέτουν οι ηλεκτρονικοί υπολογιστές και στον τρόπο που λειτουργούν.

Τα χειριστήρια των ηλεκτρονικών υπολογιστών διακρίνονται σε δύο είδη: συσκευές εισόδου και συσκευές εξόδου. Με τις συσκευές εισόδου μπορούμε να υποδείξουμε στον ηλεκτρονικό υπολογιστή τι πρέπει να κάνει και να παρέχουμε στον ηλεκτρονικό υπολογιστή τα δεδομένα που απαιτούνται. Με βάση τα μέχρι τώρα παραδείγματα μπορούμε να υποδείξουμε ότι πρέπει να εκδώσει τη μηνιαία μισθοδοσία (δηλαδή να ζητήσουμε να λειτουργήσει ή – όπως συνηθίζεται να λέγεται – να εκτελέσει το πρόγραμμα μισθοδοσίας). Αν απαιτούνται δεδομένα που δεν διαθέτει ο υπολογιστής τότε αυτά μπορούν να τροφοδοτηθούν στον υπολογιστή και πάλι από τις συσκευές εισόδου. Για παράδειγμα, μπορούμε να τροφοδοτήσουμε τις ώρες που εργάσθηκε υπερωριακά ο κάθε υπάλληλος στη διάρκεια του μήνα για τον οποίο εκδίδεται η μισθοδοσία. Τα μέχρι τώρα παραδείγματά μας είναι σχεδόν βέβαιο ότι παραπέμπουν στο πληκτρολόγιο το οποίο είναι μία συσκευή (κυρίως) για την είσοδο χαρακτήρων και ψηφίων. Μια άλλη τέτοια συσκευή είναι το ποντίκι (mouse). Για την έξοδο αποτελεσμάτων, μπορούμε να αναφέρουμε την οθόνη και τον εκτυπωτή. Υπάρχουν πολλές ακόμα συσκευές για το χειρισμό των ηλεκτρονικών υπολογιστών αλλά δεν θα επεκταθούμε. Αναφέρουμε εντελώς επιγραμματικά την αναγκαιότητα συσκευών διατήρησης πληροφορίας (συνηθίζεται να λέγονται συσκευές αποθήκευσης πληροφορίας). Τέτοιες συσκευές είναι οι μαγνητικοί δίσκοι (σκληροί δίσκοι), οι δισκέτες (εύκαμπτοι μαγνητικοί δίσκοι), τα CD και οι συσκευές flash disk.

Στο παράδειγμα του προγράμματος μισθοδοσίας, τα βασικά στοιχεία των υπαλλήλων (όνομα, βασικός μισθός, κλπ) διατηρούνται σε κάποια μονάδα αποθήκευσης πληροφορίας. Κατά την εκτέλεση του προγράμματος μισθοδοσίας, ανακαλούνται από το σκληρό δίσκο τα βασικά στοιχεία του υπαλλήλου (όνομα και βασικός μισθός), συνδυάζονται με τα δεδομένα εισόδου (ώρες υπερωριακής απασχόλησης του υπαλλήλου) και με κατάλληλους υπολογισμούς (επεξεργασία) προκύπτει το αποτέλεσμα (ο μισθός που πρέπει να καταβληθεί) το οποίο εμφανίζεται (στην οθόνη) ή εκτυπώνεται (στον εκτυπωτή). Ένα δεύτερο συμπέρασμα προκύπτει από τα παραπάνω. Η λειτουργία των ηλεκτρονικών υπολογιστών βασίζεται στο τρίπτυχο: είσοδος – επεξεργασία – έξοδος. Αν συνδυάσουμε όλα τα προηγούμενα μπορούμε να

συνοψίσουμε σε: οι ηλεκτρονικοί υπολογιστές είναι επεκτεινόμενες ηλεκτρονικές συσκευές που κάθε νέα λειτουργία που αποκτούν καθοδηγείται από ένα πρόγραμμα το οποίο χρησιμοποιεί τις συσκευές εισόδου και εξόδου καθώς και αποθηκευτικές συσκευές προκειμένου να επιτελέσει μια προσαρμοσμένη εκδοχή του βασικού τρίπτυχου και να οδηγήσει στην έκδοση των αποτελεσμάτων.

Το λειτουργικό σύστημα είναι ένα ακόμα πρόγραμμα αλλά με ιδιαίτερο ρόλο. Σκοπός του λειτουργικού συστήματος είναι να συνδυάζει και να ελέγχει σε ένα ενιαίο σύνολο όλες τις συσκευές που αναφέραμε καθώς και να αποτελεί το μέσο με τη βοήθεια του οποίου μπορούμε να προσθέτουμε νέα προγράμματα (νέες λειτουργίες) στον ηλεκτρονικό υπολογιστή μας.

Τα προγράμματα κατασκευάζονται από ανθρώπους και αποτελούνται από ιδιαίτερα απλοϊκές εντολές που γίνονται σε δυαδικό επίπεδο (βλέπε παρακάτω γλώσσα μηχανής). Όμως επειδή οι απλοϊκές εντολές (π.χ. πρόσθεσε δύο δυαδικούς αριθμούς, ανακάλεσε ένα δυαδικό χαρακτήρα – byte – από την κύρια μνήμη) θα οδηγούσαν σε τεράστια έκταση προγραμμάτων (δύσκολα σε συντήρηση) επινοήθηκαν οι γλώσσες υψηλού επιπέδου που επιτρέπουν περισσότερο σύνθετες ενέργειες σε μία εντολή. Για παράδειγμα μπορούμε, με μια εντολή να ανακαλέσουμε (από την περιφερειακή μνήμη – το σκληρό δίσκο) όλα τα στοιχεία ενός υπαλλήλου, με μια άλλη εντολή να υπολογίσουμε το μισθό του υπαλλήλου (λαμβάνοντας υπόψη βασικό μισθό, ώρες υπερωριακής απασχόλησής του και ωριαία αμοιβή για τις ώρες υπερωρίας) και με μία τρίτη εντολή να εκτυπώσουμε το έντυπο μισθοδοσίας που αφορά τον υπάλληλο. Σε τέτοιες γλώσσες (γλώσσες τρίτης γενιάς) επικεντρώνουμε την προσοχή μας στην επόμενη ενότητα.

Γλώσσες προγραμματισμού και εργαλεία ανάπτυξης προγραμμάτων

Προκειμένου να γίνει κατανοητή η αναγκαιότητα μιας γλώσσας προγραμματισμού καθώς και να ενταχθεί αυτή στο σύνολο των λειτουργιών (και αντίστοιχων εργαλείων) που απαιτούνται για την ανάπτυξη λογισμικού στην πλατφόρμα (υπό την αιγίδα) ενός λειτουργικού συστήματος (όπως είναι το DOS, το Unix, το Linux, κλπ), θα εξετάσουμε ορισμένες σχετικές βασικές έννοιες. Οι έννοιες που θα εξετάσουμε είναι: γλώσσα μηχανής, συμβολική γλώσσα (assembly language), γλώσσα προγραμματισμού υψηλού επιπέδου, πηγαίο πρόγραμμα, object (ή αντικειμενικό) αρχείο, εκτελέσιμο πρόγραμμα,

συμβολομεταφραστής (assembler), macro-assembler, μεταγλωττιστής (compiler), βιβλιοθήκη (library) και συνδέτης (linker).

Γλώσσα μηχανής (machine language)

Το σύνολο οδηγιών που αναγνωρίζει και μπορεί να εκτελέσει μία μηχανή (ένας επεξεργαστής) ονομάζεται γλώσσα μηχανής. Κάθε οδηγία συνίσταται από τον κώδικα εντολής (ή λειτουργίας ή πράξης, instruction code) και τους όρους ή τελεστέους (operands) της πράξης. Τόσο ο κώδικας λειτουργίας όσο και οι όροι της πράξης είναι δυαδικοί αριθμοί.

Συμβολική γλώσσα (assembly language)

Συμβολική γλώσσα είναι μια γλώσσα που χρησιμοποιεί κατάλληλα μνημονικά σύμβολα αντί για δυαδικούς αριθμούς που καταλαβαίνει ο επεξεργαστής του Η/Υ. Ένας στοιχειώδης συμβολομεταφραστής (ή συναρμολογητής – assembler) αναλαμβάνει να μετατρέψει (μεταφράσει) τα μνημονικά σύμβολα στους δυαδικούς αριθμούς που καταλαβαίνει ο επεξεργαστής του Η/Υ, δηλαδή μετατρέπει τη συμβολική γλώσσα μηχανής σε γλώσσα μηχανής.

Υπάρχουν διάφορα επίπεδα ωρίμανσης των συμβολικών γλωσσών προγραμματισμού. Μπορούμε να συναντήσουμε συμβολικές γλώσσες που χρησιμοποιούν μνημονικά ονόματα του κώδικα εντολής και οκταδική ή δεκαεξαδική αναπαράσταση των θέσεων μνήμης για τα δεδομένα ή τα αποτελέσματα (τελεστέους), συμβολικές γλώσσες που χρησιμοποιούν μνημονικά ονόματα του κώδικα εντολής και δεκαδική αναπαράσταση των τελεστέων αλλά ακόμα και συμβολικές γλώσσες που χρησιμοποιούν μνημονικά ονόματα τόσο για τον κώδικα εντολής, όσο και για τους τελεστέους.

Ανεξάρτητα από τα επίπεδα ωρίμανσης, οι εντολές σε μια συμβολική γλώσσα δεν απαλλάσουν τον προγραμματιστή από τον προγραμματισμό σε χαμηλό επίπεδο καθώς δεν παρέχουν «έτοιμες λύσεις» για παραστάσεις και χειρισμό ακεραίων και πραγματικών αριθμών, παραστάσεις και χειρισμό λογικών τιμών. Προφανώς, δεν παρέχουν «έτοιμες λύσεις» για παραστάσεις και χειρισμό πιο σύνθετων τύπων (ακολουθίες χαρακτήρων ή συμβολοσειρές, πίνακες, δομές που ορίζει ο χρήστης).

Είναι προφανές από τα παραπάνω ότι τα προγράμματα σε συμβολική γλώσσα μηχανής ήταν απόλυτα συνυφασμένα με τη γλώσσα μηχανής και δεν μπορούσαν να μεταφερθούν σε άλλους Η/Υ, παρά μόνο εάν οι Η/Υ

στους οποίους μεταφέρονταν είχαν την ίδια γλώσσα μηχανής (πρακτικά είχαν τον ίδιο επεξεργαστή).

Γλώσσα υψηλού επιπέδου

Οι γλώσσες υψηλού επιπέδου σχεδιάσθηκαν για να απαλλάξουν τα προγράμματα από τον περιορισμό εξάρτησής τους από τον επεξεργαστή των Η/Υ και επιπλέον για να αυξήσουν την παραγωγικότητα των προγραμματιστών. Για το σκοπό αυτό, οι γλώσσες υψηλού επιπέδου παρέχουν «έτοιμες λύσεις» για παραστάσεις και χειρισμό ακεραίων και πραγματικών αριθμών, παραστάσεις και χειρισμό λογικών τιμών. Παρέχουν «εύκολες λύσεις» για παραστάσεις και χειρισμό συμβολοσειρών, πινάκων και δομές που ορίζει ο χρήστης. Παρέχουν επίσης δομές ελέγχου ροής προγράμματος (υπό συνθήκη εκτέλεση, επαναληπτική εκτέλεση κλπ) και εντολές εισόδου-εξόδου. Ορισμένες από τις γλώσσες υψηλού επιπέδου προσφέρουν μεγαλύτερη (έναντι άλλων γλωσσών) πρόσβαση και αξιοποίηση των λειτουργιών που επιτελεί το σύστημα Η/Υ (permit access to system-level functions). Την ιδιότητα αυτή διαθέτει και η γλώσσα προγραμματισμού C που την παραθέτουμε ως κλασικό παράδειγμα διαδικαστικής γλώσσας υψηλού επιπέδου. Σε επόμενο κεφάλαιο μιλάμε για τον αντικειμενοστρεφή προγραμματισμό. Στον τελευταίο εντάσσονται οι γλώσσες C++, Smalltalk και Java. Η Java είναι η γλώσσα την οποία αναπτύσσουμε σε αυτό το βιβλίο. Παρόλα αυτά στο παρόν κεφάλαιο εστιάζουμε στις διαδικαστικές γλώσσες υψηλού επιπέδου και τα παραδείγματά μας θα είναι σε γλώσσα C.

Πηγαίο πρόγραμμα

Ένα σύνολο από εντολές που ακολουθούν τους συντακτικούς κανόνες μίας γλώσσας προγραμματισμού (είτε υψηλού επιπέδου, είτε σε συμβολική γλώσσα μηχανής), συνταγμένο (συνήθως) με ένα συντάκτη κειμένου (όπως ο vi που παρέχει το λειτουργικό σύστημα Unix ή όπως το notepad που παρέχει το λειτουργικό σύστημα Windows) και αποθηκευμένο σε ένα αρχείο του λειτουργικού συστήματος, ονομάζεται πηγαίο πρόγραμμα ή απλά πηγαίο.

Object αρχείο

Ένα σύνολο από εντολές σε γλώσσα μηχανής, αποθηκευμένο σε ένα αρχείο του λειτουργικού συστήματος ονομάζεται object αρχείο. Ένα object αρχείο, παρότι έχει εντολές σε γλώσσα μηχανής την οποία «καταλαβαίνει» ο επεξεργαστής του Η/Υ, δεν μπορεί να εκτελεσθεί από τον Η/Υ γιατί (συνήθως)

περιέχει εξωτερικές αναφορές (επικλήσεις) λειτουργιών (συνήθως αναφορές στις «έτοιμες λύσεις» που παρέχουν οι γλώσσες υψηλού επιπέδου) των οποίων οι εντολές σε γλώσσα μηχανής είναι αποθηκευμένες σε άλλα αρχεία του λειτουργικού συστήματος (βιβλιοθήκες ή άλλα object αρχεία).

Εκτελέσιμο πρόγραμμα

Ένα σύνολο από εντολές σε γλώσσα μηχανής, αποθηκευμένο σε ένα αρχείο του λειτουργικού συστήματος, το οποίο είναι «έτοιμο προς εκτέλεση» από τον επεξεργαστή του Η/Υ ονομάζεται εκτελέσιμο αρχείο. Η έκφραση «έτοιμο προς εκτέλεση» υποδηλώνει ότι στο εκτελέσιμο αρχείο έχουν συγκεντρωθεί και διασυνδεθεί κατάλληλα οι αντικειμενικοί κώδικες (γλώσσα μηχανής), τόσο του αρχικού προγράμματος, όσο και των εξωτερικών αναφορών (βιβλιοθήκες, κλπ).

Συμβολομεταφραστής (assembler)

Ένα εργαλείο (εκτελέσιμο πρόγραμμα) που μεταφράζει ένα πηγαίο από συμβολική γλώσσα σε γλώσσα μηχανής, ονομάζεται συμβολομεταφραστής (assembler).

Macro-assembler

Πολλοί συμβολομεταφραστές (assemblers) υποστηρίζουν μακροεντολές (macros), δηλαδή σύμβολα που ορίζονται από τον προγραμματιστή και αντιστοιχούν σε (ονοματίζουν και προσδιορίζουν) μια σειρά από γραμμές εντολών στη συμβολική γλώσσα μηχανής. Αυτοί οι συμβολομεταφραστές ονομάζονται και μάκρο-συμβολομεταφραστές (macro-assemblers). Οι μακροεντολές επίσης μπορεί να αντιστοιχούν σε μια ακολουθία από δεδομένα (που βρίσκονται μέσα στο πηγαίο). Μετά τη δήλωση μιας μακροεντολής (με τη χρήση της κατάλληλης ψευδολειτουργίας), το όνομά της μπορεί να χρησιμοποιείται αντί μιας πραγματικής εντολής της συμβολικής γλώσσας. Οταν ο assembler επεξεργάζεται μια τέτοια εντολή, την αντικαθιστά με το σύνολο των εντολών, στη συμβολική γλώσσα, τις οποίες η μακροεντολή προσδιορίζει. Με τον τρόπο αυτό προγράμματα σε συμβολική γλώσσα (assembly language programs) μοιάζουν να είναι πολύ μικρότερα (απαιτούν λιγότερες γραμμές πηγαίου κώδικα – όπως γίνεται με τις γλώσσες υψηλού επιπέδου). Αρκετοί macro-assemblers επιτρέπουν επίσης τη δήλωση σύνθετων δομών δεδομένων. Ορισμένοι επίσης, παρέχουν προκαθορισμένες μακροεντολές (macros) για την κλήση λειτουργιών του συστήματος (system calls).

Compiler (μεταγλωττιστής)

Ένα εργαλείο (εκτελέσιμο πρόγραμμα) που μεταφράζει ένα πηγαίο πρόγραμμα από κάποια γλώσσα υψηλού επιπέδου σε γλώσσα μηχανής (object code), για να συνδεθεί στη συνέχεια με βιβλιοθήκες (που επιλύουν τις εξωτερικές αναφορές) και να γίνει εκτελέσιμο, ονομάζεται μεταγλωττιστής (compiler). Στην απλούστερη μορφή του, ένας μεταγλωττιστής ολοκληρώνει το έργο του με την παραγωγή του object file. Στην πραγματικότητα, οι περισσότεροι μεταγλωττιστές, παρέχουν και τις υπόλοιπες ενέργειες (βήματα) μέχρι τη δημιουργία του τελικού εκτελέσιμου αρχείου. Δηλαδή, πέρα από τη μετάφραση του πηγαίου προγράμματος σε object file εκτελούν και τα βήματα του φορτωτή (linker ή library loader, εξετάζεται παρακάτω).

Βιβλιοθήκη (library)

Βιβλιοθήκη (library) είναι μια συλλογή υποπρογραμμάτων (ή λειτουργιών – επιμέρους ενεργειών – που εκτελούνται σε ένα πρόγραμμα) που μπορούν να χρησιμοποιηθούν από άλλα προγράμματα. Υπάρχουν δύο τύποι από βιβλιοθήκες: πηγαίου κώδικα και αντικειμενικού (object) κώδικα. Οι βιβλιοθήκες πηγαίου κώδικα είναι συλλογές από αρχεία, όπου κάθε αρχείο είναι (συνήθως) ένα μικρό πρόγραμμα. Με τη βοήθεια ενός συντάκτη (γνωστού και ως εκδότη) κειμένου (όπως ο vi στο Unix ή όπως το notepad στα Windows) μπορούμε να ενσωματώσουμε (άμεσα ή έμμεσα) ένα πηγαίο αρχείο στο δικό μας πηγαίο πρόγραμμα. Αυτή είναι μία απλή αλλά, σε κάποιες περιπτώσεις, επιβαρυντική αντιμετώπιση.

Οι βιβλιοθήκες αντικειμενικού κώδικα (object libraries) είναι συλλογές από μεταγλωττισμένα υποπρογράμματα τα οποία μπορούν να συνδεθούν με άλλα προγράμματα. Οι εξωτερικές αναφορές ενός προγράμματος στις μεταγλωττισμένες λειτουργίες προγραμμάτων (που βρίσκονται στις βιβλιοθήκες) επιλύονται από τους linkers (ή library loaders ή link editors) που εξετάζονται στη συνέχεια. Μπορούμε να πούμε ότι οι βιβλιοθήκες αντικειμενικού (object) κώδικα είναι περισσότερο αξιόπιστες λύσεις για την επαναχρησιμοποίηση κώδικα, σε σχέση με τις βιβλιοθήκες πηγαίου κώδικα.

Συνδέτης (linker, library loader, link editor)

Συνδέτης (linker) ονομάζεται ένα εργαλείο (εκτελέσιμο πρόγραμμα) που μας επιτρέπει να συνδέσουμε (ενοποιήσουμε) σε ένα εκτελέσιμο αρχείο, τόσο τον δικό μας αντικειμενικό κώδικα (αυτόν που προέρχεται από τη μεταγλώττιση του δικού μας πηγαίου προγράμματος), όσο και τον αντικειμε-

νικό κώδικα που εξυπηρετεί (ανταποκρίνεται) στις εξωτερικές αναφορές που κάνει το δικό μας πρόγραμμα. Συνοπτικά, ο συνδέτης είναι ένα εργαλείο που παράγει το εκτελέσιμο πρόγραμμα. Ο κώδικας που εξυπηρετεί τις εξωτερικές αναφορές του προγράμματός μπορεί να ευρίσκεται σε αλλα αντικειμενικά (object) αρχεία ή σε βιβλιοθήκες (object libraries). Το εργαλείο αυτό ονομάζεται επίσης library loader και link editor. Στο λειτουργικό σύστημα Unix παρέχεται το εργαλείο ld, το οποίο συνδυάζει μετατοπίσημα αντικειμενικά αρχεία (relocatable object files), εκτελεί τη μετατόπισή τους (performs relocation) και επιλύει τις εξωτερικές αναφορές (resolves external symbols).

Παραδείγματα χρήσης μεταγλωττιστή και linker στο Unix

Το πρώτο μας παράδειγμα χρησιμοποιεί τον προκαθορισμένο μεταγλωττιστή της γλώσσας C για να δημιουργήσει αρχικά το αντικειμενικό αρχείο (first.o) και στη συνέχεια (με τη βοήθεια της εντολής ld) να το συνδέσει με τη runtime βιβλιοθήκη της C (που βρίσκεται στο object αρχείο /lib/crt0.o) και τη standard βιβλιοθήκη της C (που βρίσκεται στο object αρχείο /lib/libc.a). Το αποτέλεσμα αποθηκεύεται στο εκτελέσιμο αρχείο first (χωρίς κατάληξη):

```
% cc –c first.c
% ld –o first /lib/crt0.o first.o –lc
```

Το επόμενο παράδειγμα επιτυγχάνει το ίδιο ακριβώς αποτέλεσμα, με τη βοήθεια του μεταγλωττιστή αλλά και της εντολής mv που παρέχει το λειτουργικό σύστημα. Η πρώτη εντολή παράγει το εκτελέσιμο αρχείο με όνομα a.out (το οποίο είναι το προκαθορισμένο όνομα εκτελέσιμου που δημιουργεί ο μεταγλωττιστής cc). Η δεύτερη εντολή μετονομάζει το αρχείο a.out σε first (χωρίς κατάληξη):

```
% cc first first.c
% mv a.out first
```

Η επόμενη εντολή επιτυγχάνει το ίδιο ακριβώς αποτέλεσμα με τη χρήση των επιπρόσθετων δυνατοτήτων φορτώματος (loading) που παρέχει ο μεταγλωτιστής:

```
% cc -o first first.c
```

Το επόμενο παράδειγμα χρησιμοποιεί σε βήματα (σε δύο εντολές) τον GNU C μεταγλωττιστή για να δημιουργήσει το εκτελέσιμο πρόγραμμα hello (χωρίς κατάληξη) από το πηγαίο hello.c:

```
% gcc -c hello.c
% gcc hello.o -o hello
```

Η επόμενη εντολή δημιουργεί το εκτελέσιμο αρχείο myprogram (χωρίς κατάληξη), με τη χρήση του προκαθορισμένου μεταγλωττιστή της γλώσσας C. Στην εντολή αυτή συνδυάζεται το αντικειμενικό μας πρόγραμμα (myprogram.o, που έχει δημιουργηθεί σε κάποιο προγενέστερο βήμα), η runtime βιβλιοθήκη της C (/lib/crt0.o, αυτόματα από την εντολή cc), η standard βιβλιοθήκη της C (/lib/libc.a, αυτόματα από την εντολή cc) και οι βιβλιοθήκες /lib/libmylib.a /lib/libX11.a (που προσδιορίζονται από το χρήστη):

```
% cc -o myprogram myprogram.o -lmylib -lX11
```

Παράδειγμα χρήσης μεταγλωττιστή και linker στο DOS

Το πρώτο μας παράδειγμα στο DOS χρησιμοποιεί το μεταγλωττιστή της Turbo C (tcc) για να δημιουργήσει αρχικά το αντικειμενικό αρχείο (first.obj) και στη συνέχεια (με τη βοήθεια του Turbo Linker, tlink) να το συνδέσει με object αρχείο lib/c0s.obj και τη βιβλιοθήκη lib/cs.lib. Το αποτέλεσμα αποθηκεύεται στο εκτελέσιμο αρχείο first.exe:

```
C:\TC>tcc -ms -c -ofirst.obj first.c
C:\TC>tlink /x first.obj lib\c0s.obj, first.exe, , lib\cs.lib
```

Η επόμενη εντολή επιτυγχάνει το ίδιο ακριβώς αποτέλεσμα, με τη χρήση των επιπρόσθετων δυνατοτήτων φορτώματος (loading) που παρέχει ο μεταγλωτιστής της Turbo C (tcc):

```
C:\TC>tcc first.c
```

Ένα ολοκληρωμένο παράδειγμα

Δημιουργούμε (π.χ. με το notepad) ένα αρχείο κειμένου με όνομα trapezio.c και το επόμενο περιεχόμενο:

```
1.   #include <stdio.h>
```

```
2.   main () {
3.      float mikri_vasi;
4.      float megali_vasi;
5.      float ypsos;
6.      float emvado;
7.
8.      mikri_vasi = 40.0;
9.      megali_vasi = 50.0;
10.     ypsos = 29.5;
11.
12.     emvado = ((mikri_vasi + megali_vasi) / 2.0) * ypsos;
13.
14.     printf("Το εμβαδό του τραπεζίου είναι %g\n", emvado);
15.  }
```

Μεταγλωττίζουμε:

```
C:\TC>tcc trapezio.c
```

Εκτελούμε: