

---

Εύτυπον 1  
Σεπτέμβριος 1998  
Π Ε Ρ Ι Ε Χ Ο Μ Ε Ν Α

Έκδοση του Συλλόγου Ελλήνων Φίλων του  $T_{E}X$  \*εφτ\*  
28ης Οκτωβρίου 366  
671 00 Ξάνθη

---

## Χαιρετισμός του καθηγητή Knuth

---

Donald E. KNUTH

*Stanford University  
Computer Science Department - Gates 4B  
Stanford, CA 94305-9045, Η.Π.Α*

Greetings to the people who can pronounce  $\TeX$  better than anyone else in the world!

When I visited your country in 1985, Melina Mercury granted me the honor of being able to deliver a lecture in the ancient theater of Epidaurus. At that time I spoke about how my “practical” work on  $\TeX$  and METAFONT has had a significant payoff also in the “theoretical” work I do in computer science and mathematics. A Greek translation of that lecture was published in *Μαθηματική Επιθεώρηση*, τεύχος **30** (1986), 3-15.

Perhaps I should apologize to you for playing tricks with your letter. If you look closely at the Computer Modern programs for lowercase mathematical Greek, you will see that I used the number .5772156649 twice in the construction of  $\gamma$ , and 3.14159 twice in the construction of  $\pi$ <sup>1</sup>.

Nothing makes me happier than to see fine typesetting being done with  $\TeX$  in all parts of the world. GO FORTH now and create masterpiece of τέχνη!

Sincerely,

Donald E. Knuth  
Professor

---

<sup>1</sup> Σ.Ε.Σ.: Ο πρώτος αριθμός είναι προσέγγιση της σταθεράς  $\gamma$  του Euler, ενώ ο δεύτερος του γνωστού μας αριθμού  $\pi$ .

---

# Πώς μεταφράζεται ἡ λέξη *ligature* στὰ Ἑλληνικά; Ἡ ἀλλιῶς: Ἡ περιπέτεια μιᾶς μετάφρασης

---

Δημήτριος Ἄ. Φιλίππου

Κάτω Γατζέα  
385 00 Βόλος

«Ἡ μετάφραση δὲν εἶναι εὐκόλο πράγμα!», θὰ ἔπρεπε νὰ μοῦ πεῖ κάποιος πρὶν ἀποφασίσω νὰ καταπιασθῶ μαζί της. Ὅμως δὲν μοῦ εἶπε κανένας κάτι τέτοιο. Ἔτσι, βάζοντας ἓνα στοίχημα μὲ τὸν ἑαυτό μου, ξεκίνησα, κάπου πρὸς τὸ τέλος τοῦ 1993 ἢ τὶς ἀρχὲς τοῦ 1994, νὰ μεταφράζω τὸ ἐγχειρίδιο τοῦ Michael Doob *Μία εὐκόλη εἰσαγωγή στὸ T<sub>E</sub>X* (στὸ πρωτότυπο: *A Gentle Introduction to T<sub>E</sub>X*), τὸ ὁποῖο κυκλοφοροῦσε τότε καὶ ἐξακολουθεῖ νὰ κυκλοφορεῖ καὶ σήμερα μέσῳ τοῦ Internet.

Τὴν ιδέα τῆς μετάφρασης τοῦ ἐγχειριδίου τοῦ Michael Doob μοῦ τὴν ἔδωσε ἀκουσίως ὁ Γιάννης Χαραλάμπους. Πρὸς τὸ μέσο τοῦ 1993 — ἐὰν θυμᾶμαι καλά — ὁ Χαραλάμπους μοῦ εἶχε δώσει πρόσβαση σὲ μία ἠλεκτρονικὴ λίστα ἀλληλογραφίας ὅπου τὰ μέλη μίας ομάδας T<sub>E</sub>Xνιτῶν μὲ τὸ ὄνομα TWGMLC (Technical Working Group on Multi-Lingual Coordination) συζητοῦσαν τὴν δημιουργία ἑνὸς πολὺγλωσσου T<sub>E</sub>X/L<sup>A</sup>T<sub>E</sub>X. Σὲ κάποια ἀπὸ τὰ μηνύματά του, ὁ Χαραλάμπους εἶχε τονίσει τὴν ἀνάγκη δημιουργίας ἢ μετάφρασης ἠλεκτρονικῶν ἐγχειριδίων τοῦ T<sub>E</sub>X καὶ τοῦ L<sup>A</sup>T<sub>E</sub>X σὲ διάφορες γλῶσσες τὰ ὁποῖα θὰ διανέμονται δωρεάν. Ὡς πλέον κατάλληλο ἠλεκτρονικὸ ἐγχειρίδιο τοῦ T<sub>E</sub>X, ὁ Χαραλάμπους εἶχε ὑποδείξει αὐτὸ τοῦ M. Doob.

Ἄδεια δὲν ζήτησα ἀπὸ κανέναν. Ἐπιπλέον δὲν εἶπα κουβέντα σὲ κανέναν γιὰ τὴν πρόθεσή μου νὰ μεταφράσω τὸ ἐγχειρίδιο τοῦ M. Doob. Ἦθελα νὰ δοκιμάσω τίς δυνάμεις μου, νὰ δῶ μεχρὶ ποῦ μποροῦσαν νὰ φθάσουν οἱ ικανότητές μου καὶ ποιὲς εἶναι οἱ ἀδυναμίες μου στὸ παιχνίδι τῆς ἀκροβασίας ἀνάμεσα σὲ δύο γλῶσσες. Ἐὰν τὰ κατάφερνα, θὰ ἔδινά τὴν μετάφραση στὸ κοινὸ μέσῳ τοῦ Internet· ἐὰν τὸ σχέδιό μου ἀποτύχαινε, ποιός θὰ ἐνοχλοῦνταν ἐφ' ὅσον δὲν εἶχα ὑποσχεθεῖ τίποτα σὲ κανέναν;

Ξεκίνησα την μετάφραση τὸν ἴδιο καιρὸ πού τελείωνα καὶ τὴν διδακτορικὴ διατριβή μου. Τί νὰ προφθάσω ὁ ἄμοιρος; Εἶχα καὶ τὸν Στρατὸ πού μὲ καλοῦσε νὰ παρουσιασθῶ στὸ τέλος τοῦ Μαρτίου 1994. Πέρασαν ὅλες αὐτὲς οἱ μπόρες — καὶ μερικὲς ἄλλες πιὸ προσωπικὲς — καὶ τελικὰ, τὸν Μάιο τοῦ 1997, δηλαδὴ τρία χρόνια μετὰ τὸ ξεκίνημα, κατόρθωσα νὰ βάλω στοὺς κόμβους τοῦ CTAN τὴν πρώτη ἐκδοσὴ τῆς μετάφρασης, τὴν ὑπ' ἀριθμὸν 0,992. Ἐκτοτε, ἡ μετάφραση τοῦ ἐγχειριδίου τοῦ Michael Doob στὰ Νέα Ἑλληνικὰ ἔχει κυκλοφορήσει σὲ ἄλλες ἑξὶ ἐκδόσεις μὲ μικρὲς βελτιώσεις ἀπὸ τὴν μία στὴν ἄλλη. Ἡ πιὸ πρόσφατη ἐκδοσὴ φέρει τὸν ἀριθμὸ 0,998 καὶ κυκλοφόρησε τὴν 27η Ἰουλίου 1998. Εὐελπιστῶ πὼς μία ἡμέρα θὰ κατορθώσω νὰ βγάλω καὶ τὴν ἐκδοσὴ ὑπ' ἀριθμὸν 1!

Πέραν ἀπὸ τὰ προσωπικὰ τρεχάματα, οἱ πρακτικὲς δυσκολίες πού εἶχα νὰ ἀντιμετωπίσω στὴν μετάφραση ἄρχισαν μὲ τὴν ἐπιλογὴ τῶν γραμματοσειρῶν. Οἱ γραμματοσειρὲς τοῦ Γιάννη Μοσχοβάκη καὶ τὸ πακέτο τοῦ greektex ἀποκλείσθηκαν ἀμέσως, ἐπειδὴ δὲν εἶναι ἰδιαίτερα εὐχρηστα παρὰ μόνον σὲ ὑπολογιστὲς πού λειτουργοῦν μὲ τὸ παλιὸ σύστημα MS-DOS. Συνεπῶς, γιὰ λόγους εὐκολίας μεταφορᾶς τῆς μετάφρασης ἀπὸ ἓνα λειτουργικὸ σύστημα σὲ ἄλλο, ἔπρεπε νὰ ἐπιλέξω ἀνάμεσα στὶς γραμματοσειρὲς rgr τοῦ Γιάννη Χαραλάμπους ἢ τὶς γραμματοσειρὲς kd τοῦ Κωστῆ Ἰ. Δρυλλεράκη (ὅταν ξεκίνησα τὴν μετάφραση, οἱ γραμματοσειρὲς cb τοῦ Claudio Beccari δὲν εἶχαν ἐμφανισθεῖ ἀκόμη). Ἐπέλεξα τελικὰ τὶς δευτέρες μιᾶς καὶ ἀποτελοῦν μέρος ἑνὸς πιὸ ὀλοκληρωμένου πακέτου μὲ μακροεντολές, τοῦ GREEKTEX. Ἄλλωστε, ἡ μόνη ἐμφανὴς διαφορὰ μεταξὺ τῶν γραμματοσειρῶν rgr καὶ kd εἶναι ἡ κωδικοποίησις τῆς περισπωμένης· στὴν πρώτη περίπτωσις ὁ χρήστης βάζει στὸν κώδικά του τὸ ἴσο = γιὰ νὰ λάβει τὴν περισπωμένη, ἐνῶ στὴν δευτέρη βάζει τὸ σύμβολο τῆς περισπωμένης ~. Ἔτσι, ἐὰν κάποιος θέλει νὰ ἀλλάξει τὶς ἑλληνικὲς γραμματοσειρὲς τῆς μετάφρασης ἀπὸ kd σὲ rgr, δὲν ἔχει παρὰ νὰ ἀλλάξει στὸν κώδικα τὰ ὀνόματα τῶν γραμματοσειρῶν στὶς ἐντολὲς \font καὶ νὰ ἀντικαταστήσει τὶς περισπωμένες μὲ τὸ ἴσον (μὲ προσοχὴ ὅμως, γιὰτὶ σὲ ἐλάχιστες περιπτώσεις ἡ περισπωμένη ~ εἶτε ἀποτελεῖ μέρος τῆς ἐντολῆς \catcode, εἶτε χρησιμοποιεῖται ὡς σύμβολο-ἐντολὴ μὲ τὴν ἔννοια ἀδιάσπαστου κενοῦ διαστήματος σὲ ἀγγλικὸ κείμενο).

Ἐχοντας ἐπιλέξει γραμματοσειρὲς τοῦ Κ. Δρυλλεράκη, τὸ ἐπόμενο δίλημά μου ἦταν ἐὰν θὰ ἔπρεπε νὰ χρησιμοποιήσω καὶ τὸ σχετικὸ ἑλληνικὸ ἀρχεῖο μορφῆς (φόρμα) .fmt τοῦ GREEKTEX γιὰ νὰ ἀποφύγω προβλήματα συλλογισμοῦ τοῦ ἑλληνικοῦ κειμένου. Ἀποφάσισα νὰ μὴν κάνω κάτι τέτοιο, ὥστε ἡ μετάφραση νὰ μὴν περιέχει ἐντολὲς ἀποκλειστικὲς τοῦ πακέτου GREEKTEX πού νὰ περιορίζουν τὴν μεταφορὰ τῆς μετάφρασης. Ἔτσι, ἡ ἐπεξεργασία τοῦ ἀρχείου gent1-gr.tex πού περιέχει ὅλον τὸν κώδικα-κείμενο τῆς μετάφρασης παραμένει ἀπλή. Ἐφ' ὅσον λοιπὸν ὑπάρχουν τὰ ἀρχεῖα τῶν ἑλληνικῶν γραμματοσειρῶν kd στὴν τελευταία τους ἐκδοσὴ (4.0α), ἀρκεῖ νὰ ἐκτελέσουμε τὴν παρακάτω ἐντολὴ στὴν γραμμὴ ἐντολῶν τοῦ λειτουργικοῦ συστήματος:

`tex gentl-gr`

Βεβαίως, σὲ πολλὰ σημεῖα τοῦ κώδικα, γιὰ νὰ ἀποφύγω τὶς ἐνοχλητικὲς ξέχειλες ἀράδες (overfull boxes), χρειάσθηκε νὰ βοηθήσω τὸ T<sub>E</sub>X στὸν συλλαβισμό σπάζοντας μεγάλες λέξεις σὲ συλλαβές. Γιὰ παράδειγμα, ἀντὶ

`stoiqeiojeto~ume`

χρειάσθηκε νὰ γράψω

`stoi\~qei\~o\~j\~e\~t\~o\~u\~l\~m\~e`

Αὐτὸ τὸ μάλλον ἐπώδυνο γράψιμο τοῦ κώδικα ἦταν ἐπιβεβλημένο προκειμένου νὰ ἐπιτύχω τὴν μέγιστη δυνατότητα μεταφορᾶς τοῦ κώδικα-κειμένου τῆς μετάφρασης σὲ ὅλα τὰ πιθανὰ συστήματα T<sub>E</sub>X.

Μία ἄλλη μεγάλη δυσκολία ποὺ συνάνησα ἦταν ἡ ἀπόδοση T<sub>E</sub>Xνικῶν ὄρων ἀπὸ τὰ Ἀγγλικά στὰ Ἑλληνικά. Πῶς λοιπὸν νὰ ἀποδίδεται σωστὰ στὰ Ἑλληνικά ὁ ὄρος *ligature*; Πολλὲς χρήσιμες ἀποδείχθηκαν λίγες γνώσεις Ἑλληνικῆς Τυπογραφικῆς Ὁρολογίας ποὺ εἶχα ἀποκομίσει ὅταν, σπουδαστὴς στὴν Ἀθήνα, εἶχα ἐργασθεῖ ὡς διορθωτὴς σὲ ἓνα ἐπιστημονικὸ περιοδικό. Δὲν ἦταν ὅμως ἀρκετὲς αὐτὲς οἱ γνώσεις. Ὁ ὄρος *ligature* μοῦ ἦταν παντελῶς ἄγνωστος μέχρι νὰ ἀρχίσω νὰ ἀσχολοῦμαι μὲ τὸ T<sub>E</sub>X (καὶ νομίζω ὅτι παραμένει ἐπίσης ἄγνωστος στοὺς περισσότερους Ἕλληνας τυπογράφους, μιᾶς καὶ δὲν ἀντιμετωπίζουν συχνὰ συνδυασμοὺς γραμμάτων ὅπως ffi ἢ ff). Χρειάσθηκε νὰ καταφύγω σὲ ἑλληνικὰ βιβλία περὶ Τυπογραφίας (π.χ., στὸ βιβλίο *Γιὰ τὴν τυπογραφικὴ δεοντολογία* τοῦ Νίκου Ε. Σκιαδᾶ, Ἐκδόσεις Gutenberg, Ἀθήνα 1992, ISBN 960-01-0340-2), στὰ ἑλληνικὰ μενοῦ ἐμπορικῶν ἐπεξεργαστῶν κειμένου (π.χ., MS-Word), σὲ φίλους καὶ γνωστούς (προσφάτως καὶ στὴν ἠλεκτρονικὴ λίστα e<sub>f</sub>t τοῦ Συλλόγου Ἑλλήνων Φίλων τοῦ T<sub>E</sub>X), καὶ στὸν προσωπικὸ αὐτοσχεδιασμό. Αὐτοσχεδιάζοντας, ἀπέδωσα τοὺς ὄρους τοῦ T<sub>E</sub>X *control word* καὶ *control symbol* ὡς λέξη ἐλέγχου καὶ σύμβολο ἐλέγχου ἀντιστοίχως. Δὲν ἐπέλεξα τὸν ὄρο *ἐντολή*, γιὰτὶ ὁ ὄρος αὐτὸς εἶναι μάλλον πιὸ περιορισμένος καὶ δὲν καλύπτει ὅλη τὴν ἐννοιολογικὴ σημασία τοῦ *control word* καὶ τοῦ *control symbol*. Ἐπίσης αὐτοσχεδιάζοντας ἀπέδωσα τὸν ὄρο *ligature* ὡς πολλαπλὸ στοιχεῖο ἢ σύνθετο στοιχεῖο. Ἀργότερα, εἶδα στὸ ἑλληνικὸ γλωσσάρι τοῦ τόμου τῆς Ἑλληνικῆς Ἑταιρείας Τυπογραφικῶν Στοιχείων *Greek Letters: from Tablets to Pixels* [Michael S. Macrakis (editor), Oak Knoll Press, New Castle (Delaware, USA) 1996, ISBN 1-884718-27-2] τὸν ὄρο *ligature* νὰ μεταφράζεται ὡς *σύνδεσμος*. Ὅμως ἡ λέξη *σύνδεσμος* ταιριάζει καλύτερα στὸν ὄρο *tie*, δηλαδὴ στὸ ἀδιάκοπτο κενὸ διάστημα ποὺ ὀρίζεται μὲ τὶς ἀκόλουθες πρωτόγονες ἐντολὲς τοῦ T<sub>E</sub>X:

---

```
\def\nobreakspace{\penalty10000\ }
```

[Στὸ plain T<sub>E</sub>X τοῦ Knuth γιὰ ἀγγλικὸ κείμενο, τὸ σύμβολο τῆς περισπωμένης ~ ἀποτελεῖ ἐνεργὸ χαρακτήρα (active character) ποὺ χρησιμοποιεῖται ὡς σύνδεσμος.]

Ἐνα ἀκόμη πρόβλημα ποὺ χρειάστηκε νὰ ἀντιμετωπίσω ἦταν ἡ ἔλλειψη ἀπὸ τὸ πρωτότυπο ἐγχειριδίου τοῦ Michael Doob κάποιου κειμένου ποὺ νὰ ἀναφέρεται στὴν στοιχειοθεσία ἐλληνικοῦ κειμένου. Γιὰ τὸν λόγο αὐτό, πρόσθεσα στὴν μετάφραση ἓνα ἐπιπλέον κεφάλαιο γιὰ τὴν στοιχειοθεσία ἐλληνικῶν κειμένων μὲ τὶς γραμματοσειρὲς τοῦ K. Δρυλλεράκη ἢ μὲ τὸ ὅλο πακέτο GREEK T<sub>E</sub>X. Ὁ λόγος γιὰ τὸν ὁποῖο δὲν ἀναφέρθηκα στὴν ἐλληνικὴ ἐπιλογή τοῦ babel ἦταν ἡ ἀνυπαρξία αὐτῆς τῆς ἐπιλογῆς ὅταν πρωτοξεκίνησα τὴν μετάφραση. Στόχος μου εἶναι σὲ κάποια ἐπόμενη ἐκδοση τῆς μετάφρασης νὰ καλύψω αὐτὴν τὴν παράλειψη.

Πρὶν θέσω τὴν μετάφραση σὲ κυκλοφορία, ἡ τελευταία ἀπορία μου εἶχε νὰ κάνει μὲ τὰ συγγραφικὰ δικαιώματα, τὸ γνωστὸ copyright. Ὁ Doob δὲν ἀναφέρει τίποτα περὶ copyright στὸ πρωτότυπο. Ὅμως ὁ ἴδιος ἔχει κυκλοφορήσει καὶ ἓνα σχεδὸν ὅμοιο βιβλίο (Michael Doob, *T<sub>E</sub>X Starting from One*, Springer-Verlag, New York, ISBN 3-540-56441-1 ἢ 0-387-56441-1). Γιὰ τὴν ἀποφυγὴ ὁποιοῦνδήποτε παρεξηγήσεων, φρόντισα νὰ ἔλθω σὲ ἐπαφὴ μὲ τὸν συγγραφέα καὶ νὰ λάβω τὴν ἄδειά του γιὰ τὴν δωρεὰν κυκλοφορία τῆς μετάφρασης μὲσω τοῦ Internet. Μὲ τὴν βοήθεια ἑνὸς ἔμπειρου καθηγητῆ, τοῦ William M. Williams τοῦ καναδικοῦ Πανεπιστημίου McGill, συνέταξα καὶ ἓνα σύντομο κείμενο ποὺ μπῆκε στὶς πρῶτες σελίδες τῆς μετάφρασης καὶ τὸ ὁποῖο καθορίζει ὑπὸ ποιὲς συνθήκες ἐπιτρέπεται ἡ ἐλεύθερη ἀναπαραγωγὴ τῆς μετάφρασης.

Πιστεύω ὅτι αὐτὴ τὴν στιγμὴ ἡ μετάφραση τοῦ ἠλεκτρονικοῦ ἐγχειριδίου τοῦ Michael Doob γιὰ τὸ T<sub>E</sub>X — ἔτσι ὅπως κυκλοφορεῖ — εἶναι πλήρης, ἔστω κι ἂν δὲν συμπεριλαμβάνει κάποιες πληροφορίες γιὰ τὴν ἐλληνικὴ ἐπιλογή τοῦ babel. Ὅποιοσδήποτε μπορεῖ ἐπίσης εὐκόλα νὰ τὴν προσαρμῶσει στὶς δικές του ἀπαιτήσεις καὶ συστήματα (διαφορετικὲς ἐλληνικὲς γραμματοσειρὲς καὶ μακροεντολές). Ὡστόσο στὰ μελλοντικά μου σχέδια συμπεριλαμβάνεται καὶ ἡ κυκλοφορία σὲ βιβλίο τῆς μετάφρασης τοῦ *T<sub>E</sub>X Starting from One*, πού, ὅπως προαναφέρθηκε, ἔχει γραφεῖ ἐπίσης ἀπὸ τὸν Michael Doob.

Κλείνοντας ἐτούτη τὴν ἀναφορὰ στὴν *Εὐκόλη εἰσαγωγὴ στὸ T<sub>E</sub>X*, θὰ ἤθελα νὰ εὐχαριστήσω τὸν Γιάννη Χαραλάμπους γιὰ τὴν ἔμμεση ἰδέα τῆς μετάφρασης, τὸν Michael Doob γιὰ τὴν καλωσύνη του νὰ μοῦ ἐπιτρέψει τὴν ἐλεύθερη κυκλοφορία τῆς μετάφρασης μὲσω τοῦ Internet, τὸν Κωστὴ Ἴ. Δρυλλεράκη γιὰ τὶς γραμματοσειρὲς ποὺ χρησιμοποίησα στὴν μετάφραση, τὸν Ἀπόστολο Συρόπουλο γιὰ τὶς διορθώσεις του, καὶ τὸν καθηγητὴ William M. Williams γιὰ τὴν βοήθειά του σχετικὰ μὲ τὶς συγγραφικὲς καὶ μεταφραστικὲς εὐθύνες καὶ δικαιώματα. Χωρὶς

---

τὴν συνδρομὴ αὐτῶν τῶν γνωστῶν καὶ φίλων ἴσως νὰ μὴν τὰ εἶχα καταφέρει ποτὲ  
στὰ μεταφραστικὰ καμώματά μου!

ΣΗΜΕΙΩΣΗ: Τὸ ἐγχειρίδιο τοῦ Michael Doob *Μία εὐκόλη εἰσαγωγή στὸ T<sub>E</sub>X* σὲ μετάφραση Δ.  
Ἄ. Φιλίππου θὰ τὸ βρεῖτε στοὺς κόμβους τοῦ CTAN <ftp.dante.de> καὶ <ftp.tex.ac.uk>, στὸν  
κατάλογο: <tex-achive/help/greek/gent1-gr>. Ἐπίσης θὰ τὸ βρεῖτε στὸν κόμβο τοῦ \*εφτ\*  
<obelix.ee.duth.gr> στὸν κατάλογο: <pub/TeXDocs>.



---

# Εισαγωγή στο PICTEX: Μέρος πρώτο

---

Απόστολος Συρόπουλος

28ης Οκτωβρίου 366  
671 00 Ξάνθη

## 1. Εισαγωγή

Το PICTEX είναι μια συλλογή από *μακροεντολές* του TEX με τις οποίες κάποιος χρήστης του μπορεί να το καθοδηγήσει να δημιουργήσει όμορφες εικόνες ως τμήματα των κειμένων που ετοιμάζει. Οι εικόνες αυτές δεν μπορεί να είναι πολύπλοκα τρισδιάστατα σχήματα, αλλά απλά σχήματα και γραφήματα του είδους που παρουσιάζονται σε μαθηματικά κείμενα. Το βασικό χαρακτηριστικό του PICTEX είναι ότι θεωρεί πως τα σχήματα αποτελούνται από σημεία και γραμμές. Αυτό όμως έχει ως αποτέλεσμα απλά σχήματα να απαιτούν πολύ μνήμη αλλά και αρκετό χρόνο για να σχεδιαστούν, τουλάχιστον παλαιότερα. Παρόλο αυτά είναι ένα χρήσιμο εργαλείο για όποιο επιθυμεί να φτιάξει εύκολα και γρήγορα κάποιο σχήμα.

Το PICTEX μπορεί να χρησιμοποιηθεί σε συνδυασμό με το plain TEX αλλά και με το L<sup>A</sup>TEX. Αν προτιμάτε να το χρησιμοποιήσετε με το plain TEX, τότε θα πρέπει να βάλετε την παρακάτω εντολή κάπου στην αρχή του αρχείου σας:

```
\input pictex
```

Αν όμως προτιμάτε να το χρησιμοποιήσετε σε συνδυασμό με το L<sup>A</sup>TEX, τότε στο θα πρέπει να βάλετε στο πρόλογο του κώδικα τις παρακάτω εντολές:

```
\input{prepictex.tex}
\input{pictex.tex}
\input{postpictex.tex}
```

Επιπλέον, επειδή το PICTEX χρησιμοποιεί τον παλιό τρόπο επιλογής γραμματοσειρών, πρέπει πριν από τις προηγούμενες εντολές να γράψετε και την παρακάτω εντολή:

```
\font\fivevm=cmr5
```

Τέλος, θα πρέπει να σημειώσουμε ότι το PICTEX σχεδιάστηκε από τον Michael J. Wichura στη δεκατία του 1980.

## 2. Το σύστημα συντεταγμένων του PICTEX

Για το PICTEX κάθε σχήμα είναι μια *εικόνα* (picture, στην ορολογία του συστήματος). Έτσι όταν θέλουμε να σχεδιάσουμε κάτι, ξεκινάμε με την εντολή `\beginpicture`, ασχέτως του αν δουλεύουμε με το plain TEX ή το L<sup>A</sup>TEX, ενώ η εντολή `\endpicture` οροθετεί το τέλος του σχήματος.

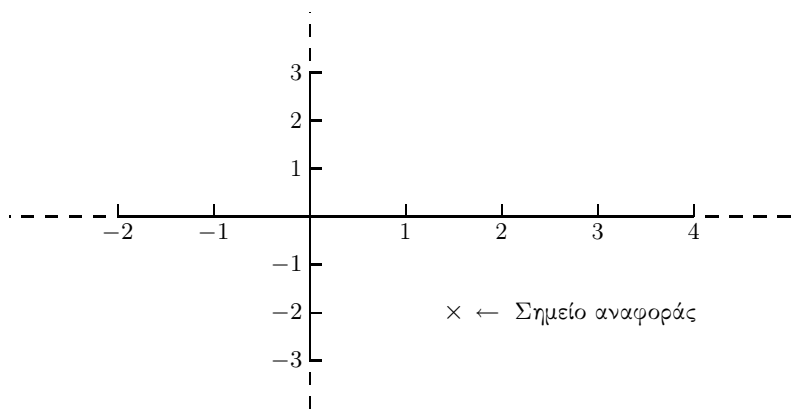
Κάθε σχήμα τοποθετείται σ' ένα καθορισμένο σύστημα αξόνων με την εντολή

```
\setcoordinatesystem units <x-μονάδα, y-μονάδα>
point at x συντέτ. y συντέτ.
```

Όταν δώσουμε την εντολή αυτή μετά την `\beginpicture` αυτό σημαίνει ότι το σύστημα συντεταγμένων αφορά μόνο την παρούσα εικόνα, αλλιώς αφορά όλες τις επόμενες. Η παράμετρος `units` αναφέρεται στο πραγματικό μήκος που θα αντιστοιχεί η μονάδα μήκους της εικόνας, τόσο οριζόντια αλλά και κάθετα. Αν παραλείψουμε την παράμετρο αυτό, το PICTEX θεωρεί ότι οι μονάδες είναι 1 pt. Η παράμετρος `point at` καθορίζει τη θέση ενός αρχικού σημείου αναφοράς. Αν την παραλείψουμε, τότε αυτή ταυτίζεται με την αρχή των αξόνων. Για παράδειγμα η εντολή

```
\setcoordinatesystem units <.5in,.25in> point at 1.5 -2
```

δημιουργεί ένα σύστημα συντεταγμένων όπως αυτό του παρακάτω σχήματος



ενώ τοποθετεί και το αρχικό σημείο αναφοράς στην θέση  $(1.5, -2)$ . Αξίζει να σημειώσουμε ότι κάθε φορά που το TEX εκτελεί μια εντολή `\setcoordinatesystem`, δημιουργεί εσωτερικά ένα φύλο χαρτιού με διαστάσεις  $1097,28 \text{ cm} \times 1097,28 \text{ cm}$ .

### 3. Τοποθέτηση κειμένου σε σχήματα

Όποιος είναι εξοικιωμένος με την χρήση του περιβάλλοντος `picture` του L<sup>A</sup>T<sub>E</sub>X, ασφαλώς θα γνωρίζει ότι μπορούμε να τοποθετήσουμε σε οποιοδήποτε σημείο του σχήματος μας με την εντολή `\put` κάποιο κείμενο ή σχήμα. Αντίστοιχη εντολή διαθέτει και το P<sub>CT</sub>EX, η σύνταξη της οποίας φαίνεται παρακάτω:

$$\backslash\text{put} \{\text{κείμενο}\} [o_x o_y] \text{ at } x\text{-συντέτ. } y\text{-συντέτ.}$$

Το αποτέλεσμα της εντολής είναι η τοποθέτηση του κειμένου στη θέση ( $x$ -συντέτ.,  $y$ -συντέτ.). Επειδή, ως γνωστό το T<sub>E</sub>X χειρίζεται πλαίσια (ή κουτιά), οι κατ' επιλογή παράμετροι  $[o_x o_y]$  καθορίζουν τη θέση του κειμένου στο πλαίσιο. Οι δυνατές τιμές των παραμέτρων και η αντίστοιχη λειτουργικότητά των φαίνεται στον παρακάτω πίνακα:

Παράμετρος	Λειτουργικότητα
l	αριστερό άκρο
r	δεξιό άκρο
t	πάνω άκρο
B	γραμμή βάσης
b	κάτω άκρο

Αν παραλείψουμε την παράμετρο  $o_x$  έχουμε κεντράρισμα οριζόντιο, ενώ αν παραλείψουμε την παράμετρο  $o_y$  έχουμε κάθετο κεντράρισμα. Η εντολή δέχεται και ένα επιπλέον κατ' επιλογή όρισμα το οποίο καθορίζει την οριζόντια και κάθετη μετάθεση του πλαισίου από την θέση που θα πήγαινε αλλιώς. Το νέο αυτό όρισμα μπαίνει ακριβώς πριν από το σημείο τοποθέτησης του κειμένου και πάντα ανάμεσα από τα σύμβολα `<` και `>`. Για παράδειγμα η λέξη `κείμενο` του παρακάτω σχήματος τοποθετήθηκε στο σημείο (2, 2),

Κείμενο

ενώ για την τοποθέτηση της μαύρης και της γκριζας κουκίδας χρησιμοποιήσαμε τις παρακάτω εντολές αντίστοιχα:

$$\backslash\text{put} \{\backslash\text{Large}\text{textbullet}\} [\text{rt}] \langle -10\text{pt}, 0\text{pt} \rangle \text{ at } 2 \ 2$$

$$\backslash\text{put} \{\backslash\text{Large}\text{graybullet}\} [\text{rt}] \text{ at } 2 \ 2$$

όπου `\graybullet` μια δικιά μας εντολή που δημιουργεί την γκριζα κουκίδα.

Σ' αρκετές περιπτώσεις θα θέλαμε να μπορούμε με μια εντολή να τοποθετήσουμε πολλά αντίγραφα κάποιου κειμένου σε πολλά διαφορετικά σημεία. Δηλαδή, αντί να γράφουμε πολλές φορές την εντολή `\put`, απλά να γράφουμε μια νέα εντολή και τα σημεία στα οποία θα τοποθετηθεί το κείμενο. Μια τέτοια εντολή είναι η `\multiput` η οποία συντάσσεται όπως και η `\put` με δύο μικρές διαφορές:

1. Τα σημεία σημειώνονται ως ζεύγη, δηλ. 3 4 5 6 7 8, ενώ πάντα στο τέλος θα πρέπει να μπαίνει το σύμβολο / και
2. Μπορούμε να σημειώνουμε ομάδες σημείων τα οποία απέχουν μεταξύ των μια καθορισμένη απόσταση. Έτσι η εντολή

```
\multiput {.} at 0 0 *10 .2 .2/
```

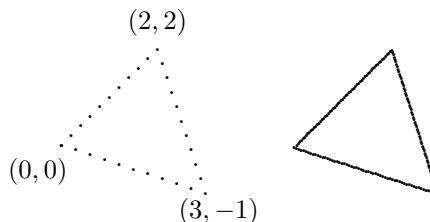
αντιστοιχεί στις εντολές

```
\put {.} at 0 0
\put {.} at .2 .2
\put {.} at .4 .4
⋮
συνολικά 10 φορές
⋮
\put {.} at 2 2
```

Δηλαδή, η παράμετρος `*ndxdy` έχει το συνδυασμένο αποτέλεσμα των παρακάτω εντολών:

```
x = x + dx
y = y + dy
\put {.} at x y
```

Τα σχήματα που ακολουθούν σχεδιάστηκαν χρησιμοποιώντας την εντολή `\multiput`:



Για παράδειγμα το αριστερό σχήμα σχεδιάστηκε με τις παρακάτω εντολές

```
\setcoordinatesystem units <.25cm,.25cm>
\multiput {.} at
0 0 *10 .2 .2 *10 .1 -.3 *10 -.3 .1/
```

(Ως άσκηση μπορείτε να προσπαθήσετε να τοποθετήσετε τις ετικέτες του σχήματος.)

Αν έχετε κάποιο πρόγραμμα που παράγει τις συντεταγμένες των σημείων κάποιου σχήματος, μπορείτε να αποθηκεύσετε τα σημεία σε κάποιο αρχείο και στη συνέχεια να χρησιμοποιήσετε το P<sub>l</sub>CT<sub>E</sub>X για τον σχεδιασμό του σχήματος. Η μαγική εντολή που αναλαμβάνει το δύσκολο αυτό έργο είναι η `\multiput`, όπου αντί για σημεία βάζουμε το όνομα ενός αρχείου που περιέχει τα σημεία. Το όνομα το αρχείου θα πρέπει να μπαίνει σε αγγλικά εισαγωγικά, π.χ.:

```
\multiput {.} at "\data.file"
```

Η δυνατότητα αυτή μπορεί, για παράδειγμα, να χρησιμοποιηθεί για τον σχεδιασμό fractal με το P<sub>l</sub>CT<sub>E</sub>X. (Στο βιβλίο L<sup>A</sup>T<sub>E</sub>X<sup>1</sup> του συγγραφέα του παρόντος υπάρχει ένα τέτοιο παράδειγμα καθώς και ένα πρόγραμμα σε Perl που παράγει τα σημεία.)

Όπως το L<sup>A</sup>T<sub>E</sub>X παρέχει την εντολή `\shortstack`, έτσι και το P<sub>l</sub>CT<sub>E</sub>X παρέχει την εντολή `\stack {κατάλογος}`, όπου ο κατάλογος είναι μια σειρά από γράμματα ή λέξεις που χωρίζονται με κόμα. Αν θέλουμε η απόσταση μεταξύ των γραμμάτων/λέξεων να είναι διαφορετική από αυτή που προϋπολογίζει το P<sub>l</sub>CT<sub>E</sub>X, τότε βάζουμε την τιμή της πριν από τον κατάλογο όπως φαίνεται παρακάτω:

```
\stack <μήκος> {κατάλογος}
```

Επιπλέον, αν θέλουμε τα γράμματα/λέξεις να στοιχίζονται στα δεξιά ή αριστερά, τότε το δηλώνουμε αυτό ως εξής:

```
\stack [δ] {κατάλογος}
```

όπου δ είναι είτε το γράμμα l (στοίχιση στα αριστερά), είτε το γράμμα r (στοίχιση στα δεξιά). Τέλος, μπορείτε να χρησιμοποιηθεί την εντολή `\shortstack` και σε

<sup>1</sup> Εκδόσεις Παρατηρητής, Θεσσαλονίκη 1998.

κείμενα, αρκεί να μην ξεχνάτε να βάζεται το σύμβολο / αμέσως μετά την εντολή,

```
ΕΛΒΕΤΙΑ
Λ
Λ
Α
Δ
```

π.χ., η λέξη `\ΑΜΕΡΙΚΗ` δημιουργήθηκε με την παρακάτω εντολή:

```
\stack [1] {ΕΛΒΕΤΙΑ,Λ,Λ,Α,Δ,ΑΜΕΡΙΚΗ} \
```

Εκτός από λέξεις και γράμματα μπορούμε να τοποθετούμε και ολόκληρες προτάσεις μία πάνω στην άλλη. Αυτό όμως δεν γίνεται με την εντολή `\stack` αλλά με δύο άλλες εντολές: την εντολή `\lines` και την εντολή `\Lines`. Και οι δύο εντολές παίρνουν τις ίδιες παραμέτρους με την εντολή `\stack`, με τη διαφορά ότι οι προτάσεις χωρίζονται μεταξύ των με την εντολή `\cr`. Επιπλέον, οι δύο εντολές διαφέρουν στο ότι η `\lines` τοποθετεί την τελευταία γραμμή στην γραμμή βάσης, ενώ η εντολή `\Lines` τοποθετεί την πρώτη γραμμή στη γραμμή βάσης. (Δοκιμάστε μόνοι σας τις δυνατότητες των νέων εντολών!)

#### 4. Σχεδιασμός αξόνων

Στην ενότητα αυτή θα μάθουμε τους τρόπους με τους οποίους μπορούμε να σχεδιάζουμε τους άξονες σε μια γραφική παράσταση. Πριν όμως από αυτό θα πρέπει να μάθουμε την χρήση της εντολής `\setplotarea`, με την οποία καθορίζουμε το χώρο που καταλαμβάνει η γραφική μας παράσταση. Η σύνταξη της εντολής φαίνεται παρακάτω:

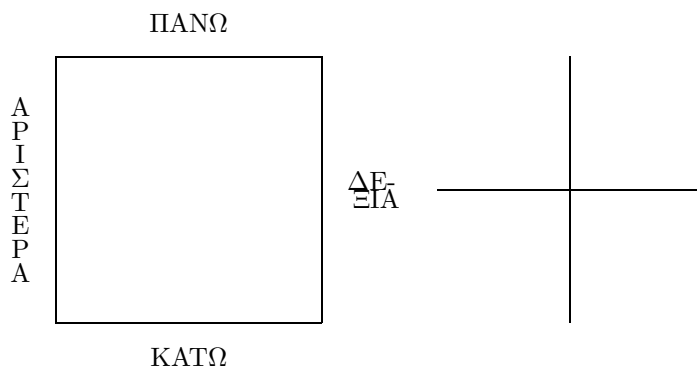
```
\setplotarea x from  $x_1$  to  $x_2$ , y from  $y_1$  to  $y_2$ 
```

Η δε σημασία της είναι ότι το σχήμα μας θα εκτείνεται οριζόντια από το  $x_1$  ως το  $x_2$  και κάθετα από το  $y_1$  ως το  $y_2$ . Έτσι η εντολή

```
\setplotarea x from 0 to 100 , y from -50 to 100
```

καθορίζει ότι ο οριζόντιος άξονας θα ξεκινάει από το 0 και θα φτάνει ως το 100, ενώ ο κάθετος θα ξεκινάει από το -50 και θα φτάνει μέχρι το 100.

Ο σχεδιασμός των αξόνων ενός σχήματος του  $\text{P}_\text{I}\text{C}_\text{T}_\text{E}_\text{X}$  γίνεται με την εντολή `\axis` η οποία είναι η πιο πολύπλοκη εντολή του  $\text{P}_\text{I}\text{C}_\text{T}_\text{E}_\text{X}$ . Δίνοντας παρακάτω ορισμένα παραδείγματα, θα εξηγήσουμε τον τρόπο χρήσης της καθώς και τις διάφορες παραμέτρους που δέχεται η εντολή. Ας δούμε δύο απλά παραδείγματα χρήσης της εντολής καθώς και τον κώδικα που τα παράγει.



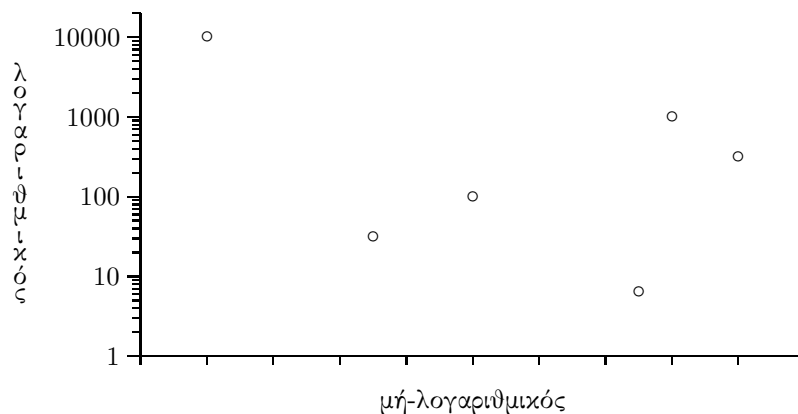
Το αριστερό σχήμα παράγεται με τον παρακάτω κώδικα:

```
\setplotarea x from 0 to 100, y from 0 to 100
\axis top label {ΠΑΝΩ} /
\axis bottom label {ΚΑΤΩ} /
\axis left label {\stack{A,P,I,Σ,T,E,P,A}} /
\axis right label {\lines{ΔΕ- \cr ΞΙΑ\cr}} /
```

Για κάθε σχήμα μπορούμε να σχεδιάσουμε τέσσερις άξονες, ένα αριστερά (left), ένα δεξιά (right), ένα πάνω (top) και ένα κάτω (bottom). Άρα βάζοντας αμέσως μετά την εντολή `\axis` την λέξη που καθορίζει τη θέση του άξονα, σχεδιάζεται ο άξονας στην ανάλογη θέση. Η παράμετρος `label` καθορίζει την ετικέτα του άξονα, το δε κείμενο, το οποίο μπορεί να είναι απλό κείμενο, σειρά από γραμμές, κ.τλ., μπαίνει αμέσως μετά σ' άγκιστρα. Το δεξιό σχήμα δημιουργεί ο παρακάτω κώδικας:

```
\setplotarea x from 0 to 100, y from 0 to 100
\axis top shiftedto x=50 /
\axis right shiftedto y=50 /
```

Το ενδιαφέρον σημείο εδώ είναι ότι μπορούμε να μετακινήσουμε κάποιο άξονα, πάνω-κάτω ή δεξιά-αριστερά ανάλογα της θέσης του. Η παράμετρος `shiftedto x=X` μετακινεί ένα οριζόντιο άξονα κατά  $X$  μονάδες, ενώ η παράμετρος `shiftedto y=Y` ένα κάθετο άξονα κατά  $Y$  μονάδες. Φυσικά επιτρέπονται και αρνητικές τιμές, οι οποίες έχουν το αναμενόμενο αποτέλεσμα. Ας δούμε ένα ακόμη ενδιαφέρον παράδειγμα:



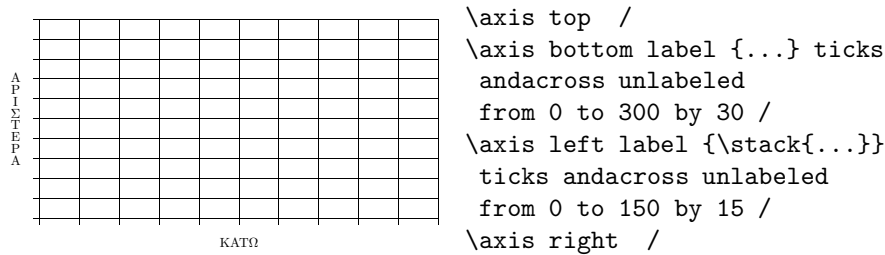
Όπως γίνεται κατανοητό το σχήμα αυτό είναι ημιλογαριθμικό, δηλ. ο ένας άξονάς του είναι λογαριθμικός. Επιπλέον στο σχήμα αυτό βάλουμε και ορισμένα σημεία (για γραμμές θα μιλήσουμε σε επόμενη ενότητα). Ας δούμε όπως τον κώδικα που παράγει το σχήμα αυτό:

```

\setcoordinatesystem units <2.5pt,30pt>
\setplotarea x from 0 to 100, y from 0 to 4.3
\axis left label {\stack{...}\
ticks logged numbered at 1 10 100 1000 10000 /
unlabeled short from 2 to 9 by 1
                    from 20 to 90 by 10
                    from 200 to 900 by 100
                    from 2000 to 9000 by 1000
at 20000 / /
\axis bottom label {...} /
ticks out withvalues 10 30 50 70 90 110
130 150 170 190 210 / short unlabeled quantity 11 /
\put {\circ} at 10 4
\put {\circ} at 50 2 \put {\circ} at 80 3
\put {\circ} at 90 2.5 \put {\circ} at 75 0.8
\put {\circ} at 35 1.5

```

Επειδή ο κάθετος άξονας είναι λογαριθμικός αυτό σημαίνει ότι η μονάδα μήκους θα πρέπει να είναι μεγάλη. Αυτός είναι και ο λόγος για τον οποίο βλέπουμε να υπάρχει τόσο μεγάλη διαφορά στις δύο μονάδες. Όπως βλέπουμε στον αριστερό άξονα χρησιμοποιούμε την λέξη `logged` για να δηλώσουμε στο `PfTeX` ότι θα πρέπει να χρησιμοποιήσει την εσωτερική του ρουτίνα υπολογισμού δεκαδικού λογαρίθμου. Επίσης, καθορίζουμε τις τέσσερις βασικές υποδιαίρεσεις του



Σχήμα 1: Παράδειγμα σχεδιασμού αξόνων.

άξονα αλλά και τις ενδιάμεσες, οι οποίες όμως σημειώνονται με μικρή γραμμή (παράμετρος `short`). Επειδή δεν θέλουμε να εμφανίζονται οι ενδιάμεσοι αριθμοί, αλλά μόνο οι υποδιαιρέσεις χρησιμοποιούμε την παράμετρο `unlabeled`. Με την παράμετρο `at` καθορίζουμε την παραπέρα σημείωση ορισμένων σημείων. Όσον αφορά τον κάτω άξονα παρατηρούμε ότι οι υποδιαιρέσεις δείχνουν προς τα έξω (παράμετρος `out`). Επίσης ότι βάζουμε 11 γραμμούλες οι οποίες αντιστοιχούν σε υποδιαιρέσεις που δεν φαινόνται. Προσέξτε όταν χρησιμοποιούμε την παράμετρο `withvalues` ορίζουμε που θα μπουν οι γραμμούλες, αλλά πρέπει πάντα να ακολουθεί η παράμετρος `quantity` με τον ακριβή αριθμό σημείων. Στο σχήμα 1 βλέπουμε ένα ακόμη παράδειγμα σχεδιασμού αξόνων. Το παράδειγμα αυτό δείχνει τη δυνατότητα σχεδιασμού γραμμών κατά μήκος (ή ύψος) των αξόνων. Αν προσέξετε τον κώδικα θα διαπιστώσετε την χρήση της παραμέτρου `andacross` η οποία είναι υπεύθυνη για το αποτέλεσμα της κατά μήκος των αξόνων τμηματοποίησης.

Εκτός από τις παραμέτρους που μόλις περιγράψαμε υπάρχουν ακόμη μερικές:

`invisible` Χρήση της παραμέτρου συνεπάγεται ό,τι οι άξονες δεν θα φαίνονται.

`visible` Έχει το ακριβώς αντίθετο αποτέλεσμα από την παράμετρο `invisible`. Παράλειψή της, σημαίνει την αυτόματη χρήση της.

`length <. .>` Με την παράμετρο αυτή καθορίζουμε το μήκος των μικρών γραμμών. Το μήκος μπαίνει ανάμεσα στα `<` και `>`.

`width <. .>` Με την παράμετρο αυτή καθορίζουμε το πλάτος των μικρών γραμμών. Το πλάτος μπαίνει ανάμεσα στα `<` και `>`.

Πριν κλείσουμε την παρούσα ενότητα αξίζει να αναφέρουμε δύο αρκετά χρήσιμες εντολές: την `\plotheading` και την `\grid {c} {r}`. Η πρώτη εντολή χρησιμοποιείται για την στοιχειοθεσία της επικεφαλίδας ενός σχήματος, το δε όρισμά του μπαίνει σε άγκιστρα. Η δεύτερη εντολή χρησιμοποιείται για τη δημιουργία

ενός πλέγματος  $c$  στηλών και  $r$  γραμμών. Προφανώς το σχήμα 1 θα μπορούσε να σχεδιαστεί ευκολότερα με την εντολή `\grid {10} {10}`, χρησιμοποιώντας βέβαια τις ίδιες παραμέτρους στην εντολή `\setplotarea`.


## 5. Γραμμές και σχήματα που αποτελούνται από γραμμές

Το  $\LaTeX$ , και προφανώς και το  $\TeX$ , μπορεί και σχεδιάζει γραμμές οριζόντιες και κάθετες. Στην ενότητα αυτή παρουσιάζουμε τις δυνατότητες σχεδιασμού γραμμών, αλλά και σχημάτων που απαρτίζονται από γραμμές, που παρέχει το  $\Pictex$ . Το πάχος των γραμμών καθορίζεται από την τιμή της μεταβλητής `\linethickness`<sup>2</sup>.

Μία γραμμή μπορεί να σχεδιαστεί με την εντολή

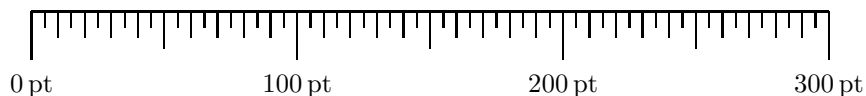
```
\putrule from  $x_1$   $y_1$  to  $x_2$   $y_2$ 
```

όπου τα  $x_1$  και  $y_1$  καθορίζουν το αρχικό σημείο και τα  $x_2$  και  $y_2$  το τελικό. Με άλλα λόγια: η γραμμή ξεκινάει από το σημείο  $(x_1, y_1)$  και τελειώνει στο σημείο  $(x_2, y_2)$ . Σημειώστε ότι τα  $x_1 \neq x_2$  και  $y_1 \neq y_2$ , δηλ. οι αρχικές και τελικές

συντεταγμένες δεν θα πρέπει να είναι ίδιες. Έτσι για παράδειγμα ο δυναμίτης  δημιουργήθηκε με τον παρακάτω κώδικα:

```
\setcoordinatesystem units <1pt,1pt>
\putrule from 0 0 to 0 15
\linethickness=6pt
\putrule from 0 0 to 0 12
```

Προσέξτε την χρήση της εντολής `\linethickness`. Δείτε ακόμη ένα παράδειγμα: ο παρακάτω χάρακας:



σχεδιάστηκε με τον παρακάτω κώδικα:

<sup>2</sup> Μπορείτε να αλλάξετε την τιμή της με μια *ανάθεση*, π.χ., η ανάθεση `\linethickness=10pt` ορίζει ότι η τιμή της θα είναι 10 pt.

```

\setcoordinatesystem units <1pt,1pt>
\putrule from 0 0 to 300 0
\multiput {\beginpicture
\putrule from 0 0 to 0 18 \endpicture}
[t] at 0 0 *3 100 0 /
\multiput {\beginpicture
\putrule from 0 0 to 0 14 \endpicture}
[t] at 0 0 *6 50 0 /
\multiput {\beginpicture
\putrule from 0 0 to 0 10 \endpicture}
[t] at 0 0 *30 10 0 /
\multiput {\beginpicture
\putrule from 0 0 to 0 6 \endpicture}
[t] at 5 0 *29 10 0 /
\put {$0\,\mathrm{pt}$} [t] at 0 -24
\put {$100\,\mathrm{pt}$} [t] at 100 -24
\put {$200\,\mathrm{pt}$} [t] at 200 -24
\put {$300\,\mathrm{pt}$} [t] at 300 -24

```

Προσέξτε την χρήση της εντολής `\multiput` αλλά και την χρήση *εγκιβωτισμένων εικόνων*.

Εκτός όμως από γραμμές το P<sub>CT</sub>E<sub>X</sub> μπορεί εύκολα να σχεδιάζει και παραλληλόγραμμα με την εντολή

```
\putrectangle corners at  $x_1 y_1$  and  $x_2 y_2$ 
```

όπου  $(x_1, y_1)$  οι συντεταγμένες της πάνω αριστερής κορυφής του, ενώ  $(x_2, y_2)$  οι συντεταγμένες της κάτω δεξιάς πλευράς του. Για παράδειγμα το παρακάτω παραλληλόγραμμα:



σχεδιάστηκε με τον παρακάτω κώδικα:

```

\setcoordinatesystem units <1cm,1cm>
\putrectangle corners at 0 2 and 4 0

```

Ως άσκηση δοκιμάστε να τοποθετήσετε τις κουκίδες (•) στο πλαίσιο που παράγουν οι προηγούμενες δύο εντολές.

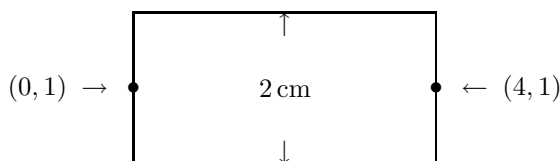
Η εντολή

```
\putbar breadth <β> xα yα to xσ yσ
```

σχεδιάζει ένα ορθογώνιο το οποίο έχει ως κέντρα απέναντι πλευρών μήκους  $\beta$  τα σημεία  $(x_\alpha, y_\alpha)$  και  $(x_\sigma, y_\sigma)$ . Θα πρέπει να πούμε ότι είτε  $x_\alpha = x_\sigma$ , είτε  $y_\alpha = y_\sigma$ . Επίσης αν  $\beta = 0$  pt, τότε το αποτέλεσμα της εντολής `\putbar` είναι το ίδιο με αυτό της εντολής `\putrule`. Για παράδειγμα, οι εντολές

```
\setcoordinatesystem units <1cm,1cm>
\putbar breadth <2cm> from 0 1 to 4 1
```

σχεδιάζουν το παρακάτω σχήμα:



Επίσης είναι δυνατό να βάλουμε ένα κείμενο σε πλαίσιο, με τον ίδιο ακριβώς τρόπο που μπορούμε να το κάνουμε με την εντολή `\fbox` του  $\LaTeX$ . Η αντίστοιχη εντολή του  $\text{P}\text{T}\text{E}\text{X}$  είναι η

```
\fram [<διάκενο>] {κείμενο}
```

όπου το *διάκενο* είναι ένα μήκος (θετικό ή αρνητικό) που καθορίζει το την απόσταση μεταξύ του κουτιού που περιέχει το κείμενο (το οποίο μπορεί να είναι σχεδόν ο,τιδήποτε) και των πλευρών του πλαισίου.

Η εντολή `\rectangle <π> <υ>` σχεδιάζει ένα ορθογώνιο πλάτους  $\pi$  και ύψους  $\upsilon$ .

Στο επόμενο τεύχος θα παρουσιάσουμε τον σχεδιασμό: ιστογραμμάτων, γραμμών και καμπυλών. Επίσης θα παρουσιάσουμε τεχνικές για τον σκιασμό σχημάτων αλλά και τον σχεδιασμό διακεκομένων γραμμών.

---

## Short History of the **cb** Greek Fonts

---

Claudio Beccari

*Dipartimento di Elettronica  
Politecnico di Torino  
Torino, Italia*

Γράμματα μὲν δὴ πρῶτος Ὀρφεὺς ἐξήνεγκε, παρὰ Μουσῶν μαθῶν, ὡς καὶ τὰ ἐπὶ τῷ μνήματι αὐτοῦ δηλοῖ ἐπιγράμματα: «Μουσῶν πρότολον τῆδ' Ὀρφέα Θρηῆκες ἔθηκαν, ὃν κτάνεν ὑψιμέδων Ζεὺς πολόεντι βέλει, Οἰάγρου φίλον υἱόν, ὃς Ἡρακλῆ ἔξεδίδαξεν, εὐρῶν ἀνθρώποις γράμματα καὶ σοφίην».

I happen to be the author of the so called **cb** Greek fonts that are being used to typeset the Greek part of this newsletter.

I would like to tell the story of how I came to spend a lot of time for producing these fonts; since I have an elementary knowledge of classical Greek, and I am so ignorant about modern demotic Greek that I can hardly decrypt easy texts with the help of a dictionary, it is out of question the possibility that I use my fonts for writing Greek.

Well, you may believe it or not, but this is my small contribution for paying my debt of gratitude to the Greek culture, that permeates all the western civilization.

I was lucky enough to frequent my junior and high schools when classical subjects were praised by the whole educated society; I studied the whole (translated) Iliad and Odyssey in 7th and 8th grades respectively, and translated from Greek both works in 9th and 10th grades. Plato, Aristoteles were common readings in high school, and I eventually sat for the final examination at the end of the 13th grade with Αἱ Χορηφόροι τοῦ Αἰσχύλου; I knew that tragedy almost by hart and I could read it both as plain text and as prosodic poetry. My children, unfortunately, don't event understand what I am speaking about when I tell them these facts.

I ended up in the engineering school, I became an electronics engineer, eventually I became a professor of electronic circuit theory, and in the past

years I acted also as vice dean of the engineering faculty. This did not keep me away from the classics, though.

Some twelve years ago I started working with L<sup>A</sup>T<sub>E</sub>X; in 1991 I even wrote a book on it, that soon became obsolete because of the advent of L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>. In that time I got my METAFONT book, but I had no time to really learn the language; then in 1996 a friend of mine, teacher of (classical) Greek in high school, triggered my interest in the possibility of writing classical Greek with L<sup>A</sup>T<sub>E</sub>X. I explored the international archives, but, certainly due to my poor Internet “surfmanship”, I missed to find the recent contributions by several other Greek and non-Greek authors. I found only the fonts designed by Silvio Levy and their variations designed by Γιάννης Χαράλαμβους, that were only in 10 point size and both, substantially, reflected the classical Didot design.

The new L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> had become available since a couple of years; it resorted to the New Font Selection Scheme, which in turn required the availability of at least three families, two series, and not less than four shapes; J. Knappen had also improved the dc Latin fonts by N. Schwarz, and produced the ec fonts by embedding a new idea, that is to code the design size into the font name, so as to produce, by means of METAFONT fonts that are not enlargements of smaller ones, but fonts designed to that very size.

Since Levy’s times also METAFONT had improved and contained more sophisticated means for describing character ligatures, that eventually proved essential with the dimorphism of the Greek letter ‘sigma’.

Therefore I started working with the idea of producing a complete collection of Greek fonts that could match the collection of the ec ones. I was lucky enough that through the Internet I came to know Απόστολος Συρόπουλος, who appreciated my preliminary work and encouraged me on going on; with his advice I ended up with the collection now known as cb fonts; thank you Απόστολε.

The cb collection includes six families: regular, sans-serif, typewriter type, outline, slides regular, slides typewriter type; two series: normal and bold (extended), the latter one applying only to a family subset; five shapes: normal, slanted, italic, upright italic, small caps (not all of them applying to all family/series combinations, though); there are a total of 65 main family/series/shape combinations that can be built at virtually every size within the range 5pt–99,99pt (the same as the ec fonts).

Although the fonts carry the version number 2.x, I suppose many corrections have still to be made; only you Greek users are likely to use all the families, series, shapes, sizes, pixel densities, etc. Only you can feed back the necessary information to improve them. I thank you in advance for your cooperation.

---

## Η κατηγορία εγγράφου apa: Μια γνήσια λύση L<sup>A</sup>T<sub>E</sub>X σε ένα σύγχρονο πρόβλημα προετοιμασίας κειμένου

---

Αθανάσιος Πρωτόπαπας

*Scientific Learning Corporation  
Berkeley, CA, USA*

Η τυποποιημένη επικοινωνία μεταξύ επιστημόνων, βασισμένη κατά κύριο λόγο στη δημοσίευση άρθρων σε εξειδικευμένα περιοδικά (παράλληλα με τη λιγότερο «επίσημη» διαδικασία των συνεδρίων), αναγκάζει τους ερευνητές να προσαρμόζουν τα κείμενά τους στις απαιτήσεις των εκάστοτε επιμελητών, στοιχειοθετών, και τυπογράφων που παράγουν το κείμενο στην τελική (δημοσιευμένη) μορφή του. Οι ιδιόρρυθμες μέθοδοι και συνήθειες των κλάδων αυτών έχουν ως αποτέλεσμα επιθυμητές μορφές κειμένου που είναι σπάταλες, ακαλαίσθητες, και δύσκολες στην ανάγνωση και την κατανόηση. Για παράδειγμα, τυπικά απαιτείται η χρήση διπλού διαστήματος μεταξύ των γραμμών και παράθεση των σχημάτων και πινάκων όχι κοντά στο σημείο του κειμένου το οποίο επεξηγούν ή συμπληρώνουν αλλά στο τέλος, και συχνά χωριστά από τους υπότιτλους τους. Μια πιθανώς ακραία περίπτωση της μορφής αυτής παρατηρείται στο εγχειρίδιο προετοιμασίας κειμένων για δημοσίευση της Αμερικανικής Ψυχολογικής Ένωσης (American Psychological Association, εξού και APA), όπου δίνονται λεπτομερείς οδηγίες όχι μόνο για τυποποίηση της επιμέλειας και στοιχειοθεσίας αλλά και για κάθε πιθανή λεπτομέρεια από τη φρασεολογία και τη χρήση συντομεύσεων μέχρι το πλάτος των περιθωρίων και αν οι υπότιτλοι των πινάκων προηγούνται ή έπονται του πίνακα!

Δεν είμαι σε θέση να εκτιμήσω τη χρησιμότητα του πλήθους των υποδείξεων των ειδικών της Ένωσης για τους παραγωγούς των επιστημονικών περιοδικών, είμαι όμως σε θέση να κρίνω τόσο το αισθητικό αποτέλεσμα όσο και τη δυσκολία στην προσαρμογή κάθε κειμένου στις απαιτήσεις του εγχειριδίου. Ελλείψει επαγγελματία δακτυλογράφου-γραμματέως που θα αναλάμβανε την προσαρμογή των χειρογράφων μου στο στρουφνό κόσμο της τυπογραφίας, και οπλισμένος με τον ηλεκτρονικό μου υπολογιστή, σχεδίασα ένα σύνολο μακροεντολών για το σύστημα L<sup>A</sup>T<sub>E</sub>X με τους εξής στόχους:

- Παραγωγή κειμένου σύμφωνα με τις προδιαγραφές του εγχειριδίου περί διαστάσεων και θέσης κάθε λειτουργικής μονάδας στο κείμενο χωρίς να χρειάζεται να ασχολούμαι με τη δημιουργία των διαφόρων ενοτήτων, τίτλων, κ.λπ. σε κάθε κείμενο (ορισμός κατάστασης man, από τη λέξη «manuscript»).
- Παραγωγή κειμένου που να διαβάζεται εύκολα και ευχάριστα, στη μορφή που έχουμε συνηθίσει στον κλάδο μας (δηλαδή αυτή των περιοδικών της Ένωσης), κατάλληλου για διάδοση (ηλεκτρονικά ή ταχυδρομικά) κατά τη διάρκεια του (δυστυχώς μακρού) διαστήματος που συνήθως μεσολαβεί από την αποστολή μέχρι τη δημοσίευση του άρθρου (ορισμός κατάστασης jou, από τη λέξη «journal»).
- Απλή μετάβαση μεταξύ των δύο μορφών με μια παράμετρο στην εντολή κλήσης του πακέτου και με πλήρη συμβατότητα σε εντολές και επεξεργασία.
- Αυτοματισμό των ανωτέρω και προσαρμογή στο σύνηθες πρωτόκολλο και σύνολο εντολών του L<sup>A</sup>T<sub>E</sub>X ώστε να μη χρειάζεται να μεταβάλλω τις συνήθειες μου στην προετοιμασία ενός κειμένου και με την προσθήκη όσο το δυνατό λιγότερων νέων μακροεντολών.

Αν και η επιλογή του συστήματος L<sup>A</sup>T<sub>E</sub>X έγινε για λόγους ανεξάρτητους των παραπάνω προϋποθέσεων (απλά ήταν το σύστημα με το οποίο ήμουν εξοικειωμένος και πολύ ευχαριστημένος) είναι προφανές ότι προσφέρεται με το παραπάνω λόγω του θεμελιώδους διαχωρισμού μεταξύ δομής και περιεχομένου που ενθαρρύνει. Μ' άλλα λόγια, είναι απόλυτα φυσικό μέσα στο πλαίσιο του L<sup>A</sup>T<sub>E</sub>X να γράφεται ένα κείμενο με συμβολικά επισήματα δομής, π.χ., έναρξης θεματικών ενοτήτων, και η τελική εμφάνιση να εξαρτάται αποκλειστικά από την κατηγορία εγγράφου με την οποία θα γίνει η επεξεργασία από το L<sup>A</sup>T<sub>E</sub>X. Ο συντάκτης του κειμένου αποφεύγει (μάλιστα, δε χρειάζεται) να ασχολείται με γραμματισμούς, διαστάσεις, στοίχιση, ή σελιδοποίηση, ενώ το έργο του προγραμματιστή της κατηγορίας κειμένου διευκολύνεται σε σημείο που να είναι δυνατό με μια απλή παράμετρο στο πρόγραμμα το ίδιο ακριβώς κείμενο να αλλάζει εντελώς παρουσιαστικό.

Ένα μεγάλο μέρος της προγραμματιστικής δουλειάς μπορούσε να γίνει επανορίζοντας απλώς μακροεντολές και μεταβλητές του συστήματος ώστε τα περιθώρια π.χ. και οι επικεφαλίδες να ακολουθούν τις προδιαγραφές του εγχειριδίου. Όσο για τις ειδικότερες λειτουργίες (π.χ. αντικατάσταση της έμφασης με υπογράμμιση, αναβολή των πινάκων και σχημάτων για το τέλος του κειμένου, διπλοδιάστημα μεταξύ γραμμών κ.ά.), οι περισσότερες είχαν για καλή μου τύχη ήδη υλοποιηθεί, χωριστά η μία από την άλλη, και βρίσκονταν διαθέσιμες στους κόμβους του CTAN, έτοιμες να ενσωματωθούν με μικρές τροποποιήσεις στο αρχείο ορισμού της κατηγορίας εγγράφου apa.

Ένα από τα πιο «καυτά» ζητήματα στην προετοιμασία κειμένων για να υποβληθούν για δημοσίευση σε περιοδικά της Αμερικανικής Ψυχολογικής Ένωσης αφορά στην παρουσίαση της βιβλιογραφίας, τόσο μέσα στο κείμενο (παραπομπές), όσο και στο βιβλιογραφικό κατάλογο στο τέλος. Συγκεκριμένα, η μορφή που απαιτείται είναι η γνωστή ως *author (year)* διότι στο κείμενο αναφέρεται το όνομα του συγγραφέα ακολουθούμενο από το έτος δημοσίευσης σε παρένθεση. Πλήθος (στην κυριολεξία) κανόνων παρέχονται στο εγχειρίδιο της Ένωσης για κάθε πιθανή ειδική περίπτωση που μπορεί κάποιος να αντιμετωπίσει. Οι περισσότεροι από τους κανόνες αυτούς είναι δυνατό να αυτοματοποιηθούν—επιπλέον, οι δυνατότητες εκμετάλλευσης του συστήματος BibTeX για το χειρισμό της βιβλιογραφίας χωρίς να χρειάζεται δακτυλογράφηση των ίδιων στοιχείων πάνω από μια φορά είναι προφανείς. Η διεθνής κοινότητα χρηστών του L<sup>A</sup>T<sub>E</sub>X για μια ακόμη φορά στάθηκε στο ύψος της, μια και ήδη υπήρχαν μερικές απόπειρες κωδικοποίησης του συστήματος παραπομπών: *newapa.bst*, *newapa2.bst*, *theapa.bst*, και κατά πάσα πιθανότητα κι άλλες που απλώς δεν ήταν διαθέσιμες στο CTAN. Επιπλέον των παραπομπών τα ανωτέρω πακέτα προσέφεραν και αυτοματοποίηση της εξειδικευμένης μορφής αριθμημένων καταλόγων που υπαγορεύονται από το εγχειρίδιο της Ένωσης. Έκτοτε ο χειρισμός των παραπομπών και της βιβλιογραφίας συμπληρώθηκε και τελειοποιήθηκε με το σύνολο μακροεντολών *apacite*, το οποίο χρησιμοποιείται στις πρόσφατες εκδόσεις του *apa.cls*.

Αν και ιστορικά κατά τι ανακριβής, η ανωτέρα παράθεση έχει τους εξής σκοπούς: Πρώτον, την περιγραφή ενός πραγματικού προβλήματος που ανέκυψε σε συνθήκες φυσιολογικής ακαδημαϊκής λειτουργίας. Δεύτερον, την αντιμετώπιση του προβλήματος στο πλαίσιο του συστήματος L<sup>A</sup>T<sub>E</sub>X, δηλαδή ως ζητήματος παρουσίασης ανεξάρτητα από το περιεχόμενο. Και τρίτον, την αναγνώριση της τεράστιας συμβολής των ανά τον κόσμο χρηστών του L<sup>A</sup>T<sub>E</sub>X, που έχει ως αποτέλεσμα την ύπαρξη συνόλων μακροεντολών σχεδόν για κάθε πιθανή στοιχειοθετική απαίτηση που μπορεί κάποιος να έχει.

Στη σημερινή μορφή του, το πακέτο *apa.cls* καλείται με την εντολή `\documentclass` με μια παράμετρο εκ των *man*, *jou*, ή *doc* για την επιλογή της επιθυμητής μορφής του κειμένου, π.χ.,

```
\documentclass[man]{apa}
```

(πέραν των δύο βασικών καταστάσεων, *man* και *jou*, που περιγράφηκαν παραπάνω, η επιλογή *doc* ακολουθεί τις βασικές προδιαγραφές της Ένωσης όσον αφορά στις παραπομπές και τις ενότητες, αναγνωρίζει κι ενδεχομένως αγνοεί τις ειδικές μακροεντολές του *apa.cls* που δεν υπάρχουν στο L<sup>A</sup>T<sub>E</sub>X, και παράγει κείμενο της συνήθους μορφής L<sup>A</sup>T<sub>E</sub>X).

Για τη χρήση του πακέτου απαιτείται ο ορισμός των συνήθων πεδίων *author* και *title* (συγγραφέα και τίτλου του άρθρου) και αρκετών συμπληρωμα-

τικών πεδίων που απαιτούνται (ή είναι απλώς προαιρετικά) από το εγχειρίδιο: `affiliation`, `acknowledgements`, `abstract`, `shorttitle`, `riphthead`, `lefthead`. Ακολουθεί η εντολή έναρξης κειμένου (`\begin{document}`) και αμέσως μετά η εντολή `\maketitle` που στοιχειοθετεί τις βασικές ενότητες ορισμού του κειμένου διαφορετικά σε κάθε μια από τις τρεις καταστάσεις μορφής. Για παράδειγμα, το παρόν κείμενο θα μπορούσε να γραφεί ως εξής:

```
\title{Η κατηγορία εγγράφου apa...}
\author{Αθανάσιος Πρωτόπαπας}
\affiliation{\textlatin{Scientific Learning Corporation}\...}
\shorttitle{Η κατηγορία εγγράφου \textlatin{apa}}
\righthheader{Η κατηγορία εγγράφου \textlatin{apa}}
\lefthead{A. Πρωτόπαπας}
```

Στη συνέχεια δακτυλογραφείται το κείμενο του άρθρου στην γνωστή μορφή του και χωρίς ο συγγραφέας να χρειάζεται να ασχοληθεί πλέον με την τελική μορφή του κειμένου, τη θέση των σχημάτων, την τακτοποίηση των βιβλιογραφικών παραπομπών, ή άλλα ομοίως δευτερεύοντα (ως προς το περιεχόμενο) ζητήματα. Για τις θεματικές ενότητες του κειμένου χρησιμοποιούνται οι οικείες μακροεντολές του  $\text{\LaTeX}$ : `\section`, `\subsection`, κτλ. Ακόμα και το μέγεθος και η διεύθυνση των σχημάτων που ορίζονται σε εξωτερικά αρχεία τύπου ενσωματωμένου `postscript` έχει πλέον αυτοματοποιηθεί με τη χρήση της μακροεντολής `\fitfigure`, που παράγει διαφορετικά αποτελέσματα στην κατάσταση `journal` απ'ό,τι στην κατάσταση `man`. Προφανώς η τήρηση των εκφραστικών και άλλων τυπογραφικών κανόνων που εκφράζονται στο εγχειρίδιο παραμένει ευθύνη του συγγραφέα αλλά δυστυχώς δεν έχει εφευρεθεί ακόμα το πρόγραμμα αυτοματισμού φρασολογίας που όλοι περιμένουμε.

Με την αναγνώριση αυτού του περιορισμού μπορούμε να κλείσουμε εδώ σημειώνοντας ότι το πακέτο `apa.cls` δείχνει πώς το σύστημα  $\text{\LaTeX}$  μπορεί να κάνει τη ζωή μας, τουλάχιστον όσων υποβάλλουν άρθρα προς δημοσίευση στην Αμερικανική Ψυχολογική Ένωση, ευκολότερη.

---

## ΒΙΒΛΙΟ—ΠΑΡΟΥΣΙΑΣΗ

---

Δημήτριος Ἄ. Φιλίππου

*Κάτω Γατζέα*

*385 00 Βόλος*

**Greek Font Society, *Greek Letters: from Tablets to Pixels* (edited by Michael S. Macrakis).** Oak Knoll Press, New Castle, Delaware, USA (1996). 325 + xxviii σελίδες (περιλαμβάνει γλωσσάρι και εύρετήριο), 29 cm × 22,5 cm, δερματόδετο. ISBN 1-884718-27-2.

Διατίθεται από την Oak Knoll Press, 414 Dealware Street, New Castle, Delaware 19720, USA. <http://www.oakknoll.com>.

Τιμή: 49,95 άμερ. δολ. (Στήν τιμή δέν συμπεριλαμβάνονται τά ταχυδρομικά έξοδα και πιθανοί δασμοί.)

Κατά τὸ διάστημα 5–9 Ἰουνίου 1995, ἡ Ἑλληνική Ἐταιρεία Τυπογραφικῶν Στοιχείων (Greek Font Society) διοργάνωσε ἕνα διεθνές συμπόσιο με θέμα: “Greek Letters: from Tablets to Pixels” (σὲ ἐλεύθερη μετάφραση: «Ἑλληνικά γράμματα: ἀπὸ τὰ ἀρχαία πινάκια στis ψηφιακές κουκίδες τοῦ ὑπολογιστή»). Τὸ συμπόσιο ἔλαβε χώρα στὸ Γαλλικὸ Ἰνστιτοῦτο Ἀθηνῶν καὶ συμμετείχαν σ’ αὐτὸ πολλοὶ γνωστοὶ τυπογράφοι, χαράκτες καὶ σχεδιαστὲς τυπογραφικῶν στοιχείων. Πρὶν ἕνα ἔτος περίπου κυκλοφόρησαν τὰ πρακτικά τοῦ συμποσίου ἀπὸ τὴν Oak Knoll Press (ΗΠΑ) σὲ ἕναν ὀγκώδη ἀλλὰ καλαίσθητο τόμο, τὸν ὁποῖο ἐπιμελήθηκε ὁ Μιχαήλ Σ. Μακράκης.

Ἡ δουλειὰ τοῦ Μακράκη στὴν παραγωγή τοῦ τόμου τῶν πρακτικῶν τοῦ συμποσίου ἀξίζει πολλοὺς ἐπαίνους. Ἡ εἰσαγωγή του εἶναι πολὺ κατατοπιστική καὶ δίνει στὸν ἀναγνώστη νὰ καταλάβει τόσο τὸ θέμα τοῦ συμποσίου ὅσο καὶ τὴν περιπετειώδη πορεία τῆς Ἑλληνικῆς Τυπογραφίας στὴν Βαβέλ τῆς σύγχρονης ἠλεκτρονικῆς στοιχειοθεσίας. Ἡ στοιχειοθεσία τῶν εἰσηγήσεων τοῦ συμποσίου εἶναι ἐξαιρετική σὲ ὅλη της τὴν λεπτομέρεια. Ὁ ἐπίλογος, τὸ Ἄβγό (Ἰδιόν) τοῦ ἀρχαίου ἑλληνα ποιητῆ καὶ φιλολόγου Σιμμία τοῦ Ροδίου, εἶναι χάσμα ὀφθαλμῶν: μία σελίδα στοιχειοθετημένη στὴν μορφή ἀβγοῦ με τοὺς σύνθετους καλλιγραφικούς χαρακτήρες Wilson Greek σχεδιασμένους ἀπὸ τὸν ἀμερικανὸ χαρακτήρα καὶ τυπογράφο Matthew Carter. Τέλος, στὰ παραρτήματα τοῦ τόμου ὑπάρχει ἕνα σύντομο γλωσσάρι ποὺ μπορεῖ νὰ ἀποδειχθεῖ πολὺ χρήσιμο γιὰ ὅσους ἀσχολοῦνται με τὴν τυπογραφία (καὶ ὄχι μόνον).

Ἐπίσημος προσκεκλημένος τοῦ συμποσίου ἦταν ὁ διάσημος γερμανὸς καλλιγράφος καὶ σχεδιαστὴς τυπογραφικῶν στοιχείων Hermann Zapf. (Γιὰ ὅσους δὲν γνωρίζουν τὸν Zapf, ἀρκεῖ νὰ ποῦμε ὅτι ἔχει σχεδιάσει τὴν γραμματσοειρὰ Palatino, ἐνῶ βοήθησε καὶ τὸν Knuth στὸν σχεδιασμὸ τῶν γραμματσοειρῶν Computer Modern γιὰ τὸ T<sub>E</sub>X.) Στὴν εἰσήγησή του, ὁ Zapf ἀναφέρεται στὴν ἱστορία τῆς Ἑλληνικῆς Τυπογραφίας, ἀπὸ τὰ βυζαντινὰ χειρόγραφα ἕως τὴν φωτοσύνθεση, καὶ δίνει τὴν προσωπικὴ του ἀποψη σὲ θέματα σχεδιασμοῦ ἑλληνικῶν τυπογραφικῶν στοιχείων.

Οἱ εἰσηγήσεις τῶν P. Kyle MacCarter Jr., Stephen V. Tracy, Anna Pontani, Ἀγαμέμωνα Τσελίκα, Nicolas Barker, John A. Lane καὶ John H. Bowman ἀναφέρονται ἐπίσης στὴν ἐξέλιξη τῆς Ἑλληνικῆς Γραφῆς καὶ Τυπογραφίας ἀπὸ τὴν Ἀρχαιότητα ἕως τὴν μονοτυπία καὶ τὴν λισοτυπία. Γιὰ τὸν μελετητὴ τῆς Ἑλληνικῆς Τυπογραφίας, ξεχωρίζει ἡ εἰσήγηση τοῦ John H. Bowman ποὺ πραγματεύεται τὴν συνδρομὴ τῶν βρετανῶν χαρακτῶν στὸν σχεδιασμὸ ἑλληνικῶν τυπογραφικῶν στοιχείων.

Οἱ Jérôme Peignot, Τάκης Κατσουλίδης, Ἐμμανουὴλ Χ. Κάσδαγλης, Matthew Carter καὶ Γεώργιος Δ. Μαθιόπουλος πραγματεύονται στὶς εἰσηγήσεις τους τὰ προβλήματα τοῦ σύγχρονου σχεδιασμοῦ ἑλληνικῶν τυπογραφικῶν στοιχείων: τὴν φυσιογνωμία τῶν τύπων, τὸν ἐπαναστατικὸ σχεδιασμὸ τῆς ἑλληνικῆς γραμματσοειρᾶς Θεόκριτος ἀπὸ τὸν χαρακτὴ Γιάννη Κεφαλληνό (1894–1956) τὴν σχέση ἑλληνικῶν καὶ λατινικῶν τύπων καί, τέλος, τὴν οὐσιαστικὴ ἀνυπαρξία πλάγιων–καλλιγραφικῶν τύπων (*italic*) γιὰ τὴν στοιχειοθεσία ἑλληνικῶν κειμένων.

Τὰ προβλήματα τῆς στοιχειοθεσίας ἑλληνικῶν κειμένων στὴν ἐποχὴ τῆς φωτοσύνθεσης καὶ τοῦ ἠλεκτρονικοῦ ὑπολογιστῆ ἀποτελοῦν τὸ ἀντικείμενο τῶν εἰσηγήσεων τῶν Νικόλαου Μ. Παναγιωτάκη, Jeffrey Rusten, Sylvio Levy, Pierre A. MacKay, Κωνσταντίνου Μυλωνᾶ, Louis Rosenblum, Ἀλεξίου Ζάβρα, Εὐαγγέλου Ε. Μελαγράκη καὶ Σταύρου Μ. Μακράκη. Εἰδικώτερα ὁ Νικόλαος Μ. Παναγιωτάκης κατακρίνει τὴν μᾶλλον ἐπιπόλαια ἀπόφαση τῆς Ἑλληνικῆς Κυβέρνησης νὰ καταργήσει τὸ πολυτονικὸ σύστημα γραφῆς μέσα σὲ μίαν νύχτα. (Πράγματι, ἡ Βουλὴ τῶν Ἑλλήνων ψήφισε τὴν κατάργηση τῶν τόνων ὡς μέρος ἐνὸς νομοσχεδίου γιὰ τὴν δευτοροβάθμια ἐκπαίδευση τὴν νύχτα τῆς 11ης πρὸς 12η Ἰανουαρίου 1982 μὲ παρόντες μόνον τριάντα μισοκοιμισμένους βουλευτές!) Οἱ εἰσηγήσεις τοῦ Sylvio Levy, τοῦ Pierre A. MacKay καὶ τοῦ Κωνσταντίνου Μυλωνᾶ ἀναφέρονται σὲ γραμματσοειρὲς καὶ μακροεντολὲς ποὺ σχεδίασαν γιὰ τὴν στοιχειοθεσία ἑλληνικῶν κειμένων μὲ τὸ T<sub>E</sub>X. Ὁ Levy καὶ ὁ Μυλωνᾶς δίνουν μίαν σύντομη περιγραφὴ τῶν γραμματσοειρῶν ποὺ σχεδίασαν μὲ τὸ METAFONT, ἐνῶ ὁ MacKay περιγράφει τὸ πακέτο μακροεντολῶν *ibycus*, ποὺ δημιούργησε γιὰ τὴν στοιχειοθεσία ἀρχαίων ἑλληνικῶν κειμένων ποὺ περιλαμβάνονται στὸ CD-ROM *Thesaurus Linguae Graecae*.

Ο τόμος κλείνει με τὰ πρακτικὰ μίας συζήτησης στογγυλῆς τραπέζης πού πραγματοποιήθηκε στὸ τέλος τοῦ συμποσίου. Βασικὸ ἀντικείμενο τῆς συζήτησης ἀποτέλεσαν οἱ γραμματοσειρές GFS Bodoni, GFS Didot, GFS Artemisia, Epigraphic-Neohellenic, GFS Porson Italic καὶ GFS Olga Italic, οἱ ὁποῖες σχεδιάσθηκαν ἀπὸ τὸν Τάκη Κατσουλίδη κατόπιν παραγγελίας τῆς Ἑλληνικῆς Ἑταιρείας Τυπογραφικῶν Στοιχείων. Ἄς σημειωθεῖ ὅτι οἱ γραμματοσειρές GFS διατίθονται ἤδη σὲ μορφή PostScript Type 1 καὶ TrueType γιὰ Macintosh καὶ MS-Windows ὡς μέρος τοῦ ἐμπορικοῦ πακέτου GreekKeys [περισσότερες πληροφορίες στὶς διευθύνσεις: Ἑλληνικὴ Ἑταιρεία Τυπογραφικῶν Στοιχείων, Σπύρου Μερκούρη 33, 116 35 Ἀθήνα (τηλ. 72 51 979), καὶ Scholars Press, Customer Service, P.O. Box 133089, Atlanta, Georgia 30333-3089, USA (<http://scholar.cc.emory.edu>)].

Ο τόμος *Greek Letters: from Tablets to Pixels* ἀξίζει ὅπωςδήποτε νὰ ὑπάρχει στὴν βιβλιοθήκη ἑνὸς σχεδιαστῆ ἐλληνικῶν γραμματοσειρῶν μὲ τὸ METAFONT καὶ μακροεντολῶν γιὰ στοιχειοθεσία ἐλληνικοῦ κειμένου μὲ τὸ TEX. Ὡστόσο ἡ ὑψηλὴ τιμὴ του (ξεπερνᾷ τὶς 16.000 δρχ.) μᾶλλον θὰ ἀποθαρρύνει ὅσους ἀσχολοῦνται ἐρασιτεχνικὰ μὲ τὴν τυπογραφία. Μὲ λίγα λόγια, πρόκειται γιὰ ἐξαιρετικῆς ποιότητος βιβλίον πού ἀπευθύνεται στὸν εἰδικό, ἀλλὰ ὄχι στὸν μέσο χρήστη τοῦ TEX πού ἀναζητᾷ ἀπλῶς ἓνα πρακτικὸ ἐγχειρίδιο καθημερινῆς χρήσης.



---

## Συνέντευξη του Donald Knuth στα Βιβλιοπωλεία Computer Literacy (7 Δεκεμβρίου 1993)

---

Μετάφραση: Αντώνης Τσολομούτης

Πανεπιστήμιο Κρήτης  
Τμήμα Μαθηματικών  
Λεωφ. Κνωσού  
71409 Ηράκλειο, Κρήτη  
email: [atsol@itia.math.uoh.gr](mailto:atsol@itia.math.uoh.gr)

Ο Donald Knuth θεωρείται ο διασημότερος άνθρωπος στον χώρο της επιστήμης των η/υ παγκοσμίως. Οι πρώτοι τρεις τόμοι της σειράς βιβλίων με γενικό τίτλο *The Art Of Computer Programming*, εργασία που θεωρείται πλήρης για εδώ και 30 χρόνια, του χάρισε το ACM Turing Award<sup>1</sup> το 1974 και το National Medal of Science το 1979. Ο Knuth έχει επίσης αναπτύξει εφαρμογές υψηλότερου επιπέδου στη στοιχειοθεσία με χρήση υπολογιστή (« $\TeX$ » και « $\text{METAFONT}$ ») και ανάπτυξης λογισμικού (« $\text{CWEB}$ »), και έχει πάνω από 100 επιστημονικές δημοσιεύσεις.

Τώρα, ως επίτιμος καθηγητής στο πανεπιστήμιο του Stanford, ο Knuth διοχετεύει την ενέργειά του σε συγγραφική δουλειά. Ο Dan Doernberg του πήρε συνέντευξη τον Δεκέμβριο του 1993 για να δει με τι ασχολείται τελευταία και άρα τι θα πρέπει εμείς να περιμένουμε από τνω δουλειά του.

**CLB:** Μόλις δημοσιεύσατε δύο βιβλία πάνω στο  $\text{CWEB}$  και στη Stanford GraphBase δύο περιοχές των ερευνητικών σας ενδιαφερόντων. Ας αρχίσουμε με το  $\text{CWEB}$ , που συνδέει την [γλώσσα προγραμματισμού] C με το  $\TeX$  ώστε να είναι εύκολη η τεκμηρίωση των προγραμμάτων.

**Knuth:** Το σύστημα  $\text{CWEB}$  είναι ένα κομμάτι που προστίθεται στη C και κάνει το πρόγραμμα καλύτερο κατά πολύ από οποιαδήποτε άλλη γνωστή μέθοδο. Πρέπει απλά να μιλήσω τίμια και να πω ότι είναι το καλύτερο πρόγραμμα για αυτό τον σκοπό. Το βιβλίο *The  $\text{CWEB}$  system of structured documentation* [*To*

---

<sup>1</sup> Σ.Σ.Ε.: Ανώτατη τιμητική διάκριση της *Συνεργασίας για τις υπολογιστικές μηχανές* (ACM), που είναι κάτι σα Νόμπελ Πληροφορικής.

σύστημα *CWEB* δομημένης τεκμηρίωσης] είναι ένα πλήρες εγχειρίδιο χρήσης και επεξηγήσεων, πληρέστερο από όσο θα το χρειαζόταν κανείς.

**CLB:** Έχετε πει ότι το *CWEB* βελτιώνει την παραγωγή ενός προγραμματιστή κατά μία τάξη μεγέθους. Πως συμβαίνει αυτό;

**Knuth:** Ίσως όχι μία τάξη μεγέθους αλλά μάλλον διπλασιάζει την παραγωγή. Όσοι έχουν χρησιμοποιήσει το *CWEB* παρατήρησαν ότι γράφουν με αυτό καλύτερα προγράμματα, ότι τα προγράμματά τους είναι περισσότερο ανεξάρτητα υπολογιστικής πλατφόρμας, ότι ευκολότερα διορθώνονται και βελτιώνονται. . . και παίρνουν λιγότερο χρόνο για να τα γράψουν.

**CLB:** Το *CWEB* έχει χρησιμοποιηθεί μόνο στο **Stanford** ή και σε εταιρείες γενικότερα;

**Knuth:** Χρησιμοποιείται σε όλο τον κόσμο. Αρχικά είχαμε το *WEB* την αρχική έκδοση (για *Pascal*) σε μία ποικιλία υπολογιστικών συστημάτων και στη συνέχεια όλο και περισσότεροι άνθρωποι άρχισαν να «κολλάνε το μικρόβιο». Το *TeX* γράφτηκε με το *WEB*. Ο *Silvio Levy* το μετέτρεψε σε *CWEB* το 1987. Ήταν πειραματικό για πολύ καιρό και τώρα μπορώ να πω: «το πείραμα πέτυχε!». Το *CWEB* είναι πολύ καλύτερο από το *WEB* γιατί η *C* είναι πολύ καλύτερη γλώσσα για προγραμματισμό [από την *Pascal*]. Δεν θα μπορούσα να καταλάβω γιατί κάποιος που ενδιαφέρεται για προγραμματισμό θα προτιμούσε ένα άλλο σύστημα από αυτό.

**CLB:** Εύκολο στη χρήση, τρέχει γρήγορα, όλα αυτά τα όμορφα πράγματα;

**Knuth:** Ακριβώς, και επιπλέον σου δίνει χαρά αφού τελειώσεις το πρόγραμμα!

**CLB:** Ακόμα και αν γράφεις ένα κακό πρόγραμμα;

**Knuth:** Σχεδόν. . . η [σύζυγος μου η] *Jill* θα σας πει ότι συχνά βγαίνω από το γραφείο μου φωνάζοντας: «Ο προγραμματισμός με το *CWEB* είναι πολύ διασκεδαστικός». Είναι αλήθεια. Δεν τον χορταίνω. Όταν γράφεις ένα πρόγραμμα με το *CWEB* αισθάνεσαι ότι μιλάς με έναν άνθρωπο εξηγώντας του το τι πρέπει να κάνει ο υπολογιστής, αντί να νοιώθεις ότι τα λες σε έναν υπολογιστή. Επιτυγχάνεις τον στόχο σου ευκολότερα όταν μιλάς σε να μιλάς σε άνθρωπο. Αυτή η προσέγγιση βοηθάει ακόμα και για ένα πρόγραμμα που ίσως να σου είναι άχρηστο μετά από μία ώρα. Το *CWEB* είναι ένα εργαλείο που το συνιστώ ακόμα και αν γράφεις ένα πρόγραμμα για τον εαυτό σου μόνο, για τα μάτια σου μόνο.

**CLB:** Το *CWEB* φαίνεται να μοιάζει με τα μοντέλα δομημένου προγραμματισμού της δεκαετίας των 70. . .

**Knuth:** Σωστά, είναι το επόμενο βήμα. Με τον δομημένο προγραμματισμό κάποιοι έλεγαν προγραμματίστε: Από πάνω προς το κάτω [top-down]. Ενώ κάποιοι άλλοι πρότειναν την αντίστροφη πορεία. Με τα *WEB/CWEB* μπορεί κανείς

να γράφει όπως εκείνος νοιώθει ότι είναι καλό για το πρόγραμμα ή για το τμήμα του προγράμματος με το οποίο ασχολείται. Η μεθοδολογία του δομημένου προγραμματισμού ήταν πολύ καλή, αλλά ο τρόπος με τον οποίο πρέπει να την προσεγγίσει κανείς διαφέρει από απλό συνταγολόγιο, αλλά έχει να κάνει πιο πολύ με τη σχέση μεταξύ υψηλού και χαμηλού επιπέδου όψεις του προγράμματος. Αυτό το κάνεις με το να αντιμετωπίζεις το πρόγραμμα σα ιστό [web], σαν ένα σύνολο από μικρά κομμάτια απλά και αυτόνομα με απλές συνδέσεις μεταξύ τους. Αυτός ο τρόπος του να αντιμετωπίζεις κανείς το περίπλοκο «όλον» βλέποντας τα απλά μέρη του και τον απλό τρόπο με τον οποίο συνδέονται, υποστηρίζεται από το WEB. Μπορεί κανείς να γράφει τα μέρη του προγράμματος με οποιαδήποτε σειρά αυτός επιθυμεί. Μερικές φορές γράφεις από το τέλος προς την αρχή. Αυτό σημαίνει ότι πιστεύεις ότι θα χρειαστείς μία υπορουτίνα οπότε την γράφεις τη στιγμή που αισθάνεσαι έτοιμος για αυτό. Με αυτή την αντίστροφη πορεία ο προγραμματισμός γίνεται ισχυρότερος αφού όταν φτάσει κανείς στην ένατη σελίδα έχει αναπτύξει περισσότερα εργαλεία για την δέκατη σελίδα. Με την άλλη πορεία (από την αρχή προς το τέλος) ξεκινάς λέγοντας πρώτα θα φτιάξω αυτό, μετά εκείνο, κ.λπ., αλλά πρέπει να προγραμματίσεις την κατασκευή αυτών των κομματιών. Πολύ πιθανό να φτάσεις να γράφεις 100 σελίδες πριν ανακαλύψεις πως θα μπορούσες να είχες γράψει αυτά τα κομμάτια. Αυτή η πορεία φαίνεται καλή στις πρώτες σελίδες αλλά μετά γίνεται πολύ δύσκολη δουλειά. Η αντίστροφη πορεία επίσης φαίνεται καλά στις πρώτες σελίδες αλλά τελικά αποδυναμώνεται ισχυρότερη αφού λαμβάνει υπόψη της την τρέχουσα ψυχολογία του προγραμματιστή.

Αυτό έκανα και με το TeX, ένα πολύ μεγάλο πρόγραμμα με περισσότερες από 500 σελίδες κώδικα. Σε όλη την πορεία της συγγραφής του προγράμματος το επόμενο βήμα ήταν μοναδικό, προκαθορισμένο από το τι είχα γράψει μέχρι εκείνη τη στιγμή. Καμία μεθοδολογία δεν θα μπορούσε να μου μάθει πως να γράψω ένα τέτοιο πρόγραμμα, αν έπρεπε να την ακολουθήσω αυστηρά. Αν όμως θελήσω να εξηγήσω το πρόγραμμα σε έναν καλό προγραμματιστή υπάρχει μόνο ένας –φυσικός– τρόπος να το κάνω. Η σειρά με την οποία εμφανίζετε ο κώδικας στο βιβλίο είναι η σειρά με την οποία τον έγραφα.

**CLB:** Σε ποιά βαθμό ακολουθήτε τον ιερό πόλεμο για την μεθοδολογία παραγωγής προγραμμάτων;

**Knuth:** Δεν παρακολούθησα σε κάθε λεπτομέρεια αυτή την ιστορία αλλά ήμουν ενήμερος για τις ιδέες που κυριαρχούσαν. Νομίζω πως ήταν υπερβολή να γίνει αυτό το θέμα θρησκεία. Εκείνα τα χρόνια υπήρχε θέμα πολιτικής ορθότητας για το πως έπρεπε να γραφτεί ένα πρόγραμμα. Υπήρξε κάτι ανάλογο στην κοινότητα των μαθηματικών στην δεκαετία του 20 οπότε οι άνθρωποι έλεγαν ότι οι καλοί μαθηματικοί έπρεπε να αποδεικνύουν με συγκεκριμένο τρόπο. Δεν σου επιτρεπόταν να χρησιμοποιήσεις συγκεκριμένες τεχνικές απόδειξης επειδή κάποιος πίστευαν ότι υπήρχε ο κίνδυνος να οδηγηθείς σε αντιφάσεις. Ήταν σα να προσπαθείς να κάνεις μαθηματικά με το ένα σου χέρι δεμένο πίσω από την πλά-

τη σου. Ομοίως ο «πολιτικά ορθός» δομημένος προγραμματισμός εμπόδιζε τον κόσμο από το να γράφει καλά προγράμματα, ενώ γνώριζαν τι έκαναν· απλά ο τρόπος με τον οποίο προσέγγιζαν το πρόβλημα δεν συμφωνούσε με την ιδέα του «σωστού».

Η επιστήμη των υπολογιστών είναι όπως οποιαδήποτε άλλη επιστήμη· ακολουθεί την μόδα. Κάποιες ιδέες είναι καλές, αλλά σχεδόν όλες τις καλές ιδέες τις συνιθίζου οι άνθρωποι διαφορετικά από ότι θα έπρεπε. Για παράδειγμα κοιτάζετε τι έγινε με τις γεννήτριες τυχαίων αριθμών. Δεν είχαμε ιδέα για το πως να παράγουμε τυχαίους αριθμούς για 15 χρόνια. Μετά κάποιος απέδειξε ένα μικρό αποτέλεσμα για μία συγκεκριμένη τεχνική: αν υπολογίσεις την μέση τιμή του γραμμικής συσχετίσης μιάς περιόδου ενός δισεκατομυρίου αριθμών το αποτέλεσμα είναι μηδέν. Και ξαφνικά στράφηκαν εκεί. Πήραν όλες τις παλιές τους ρουτίνες και τις προσαρμόσαν σε αυτή τη μέθοδο γιατί ήταν το μόνο διαθέσιμο θεωρητικό εργαλείο. Αποδείχθηκε ότι ήταν μία πολύ κακή μέθοδος γιατί δεν είχε προβλέψει η θεωρία ότι ο μέσος όρος αυτής της περιόδου με το πρώτο μισό δισεκατομύριο ίσο με +1 και το υπόλοιπο μισό ίσο με -1 έκανε πάλι μηδέν!

Σε όλη την πορεία της ιστορίας οι άνθρωποι δανίστηκαν ιδέες και δεν αντιλήφθηκαν τα όρια αυτών των ιδεών.

**CLB:** Ποιά ήταν αυτή η μέθοδος για την οποία μιλάτε;

**Knuth:** Λεγόταν RANDU στις περισσότερες βιβλιοθήκες υπορουτίνων. Είναι αδύνατο να την βρείτε κάπου· αν παρόλα αυτά κάποιος δει μια υπορουτίνα με όνομα RANDU το καλύτερο που έχει να κάνει είναι να απαλλαγεί αμέσως από αυτήν!

**CLB:** Την συγχώνευση του WEB με την [γλώσσα προγραμματισμού] C την κάνατε γιατί η C χρησιμοποιείται ευρέως από του προγραμματιστές ή γιατί σας αρέσει η C και την χρησιμοποιείτε;

**Knuth:** Νομίζω ότι η C έχει πολλές σημαντικές δυνατότητες. Ο τρόπος με τον οποίο χειρίζεται τους δείκτες για παράδειγμα είναι μια καταπληκτική καινοτομία· έλυσε πολλά προβλήματα που είχαμε πριν με τον δομομένο προγραμματισμό και βελτίωσε την εμφάνιση των προγραμμάτων. Η C δεν είναι η τέλεια γλώσσα, καμία γλώσσα δεν είναι τέλεια αλλά νομίζω ότι έχει πάρα πολλά καλά στοιχεία και επίσης μπορείς να αποφύγεις τα μέρη που δεν σου αρέσουν. Πράγματι η C μου αρέσει ειδικά γιατί δίνει καλά με το λειτουργικό (αν χρησιμοποιείς Unix για παράδειγμα). Σε όλη μου την ζωή χρησιμοποιούσα πάντα την γλώσσα που έδενε καλύτερα με το debugging του λειτουργικού που χρησιμοποιούσα. Αν είχα καλύτερο debugger για την γλώσσα Ξ και αν η Ξ έδενε καλύτερα με το λειτουργικό θα χρησιμοποιούσα αυτή. Μια ακραία περίπτωση συνέβη κάποτε όταν ούλενα σε ένα εργαστήριο που το λειτουργικό σύστημα ήταν σχεδιασμένο από τον Ned Irons. Το σύστημα ήταν για ένα από τα πρώτα Cray και ο Irons είχε γράψει και έναν συμπιλιστή (compiler) μιας γλώσσας που λέγονταν IMP. Η IMP είχε διάφορα

άσχημα ένα εκ των οποίων το ότι ήταν μία επεκτάσιμη γλώσσα και οποιοσδήποτε στο εργαστήριο μπορούσε να την επεκτείνει. Έτσι ένα πρόγραμμα που δούλευε την Δευτέρα, δεν δούλευε την Τρίτη και το πρώτο πράγμα που έκανε κανείς όταν συνέβαινε αυτό ήταν να κοιτάξει αν ο συμπιλιστής ήταν εντάξει. Το δεύτερο πράγμα για τον IMP ήταν ότι ήταν πολύ στριφνή γλώσσα. Για παράδειγμα στην PASCAL κανείς γράφει `IF X > 0 THEN . . .` ενώ στην IMP θα έπρεπε να γράψει `X+=>`. Με άλλα λόγια το πρόγραμμα ήταν μικρό σε μέγεθος. Ένοιωθες ότι έγγραφες κομψά προγράμματα γιατί αποτελούντουσαν από λιγοστούς χαρακτήρες. Όμως την επόμενη μέρα δεν μπορούσες να τα διαβάσεις!

**CLB:** Αντιλαμβάνομαι ότι τώρα βάζετε έμφαση σε αυτό που λέμε *λογοτεχνικό προγραμματισμό* (literate programming), αλλά σας άρεσε ποτέ μία πιο μαθηματική γλώσσα όπως η APL;

**Knuth:** Αυτό είναι άλλο θέμα. Η APL είναι για εκείνους που έχουν να λύσουν κάποια προβλήματα και δεν τους ενδιαφέρει η αποτελεσματικότητα: θέλουν έναν όμορφο και κομψό τρόπο να διατυπώσουν τη λύση του προβλήματος τους ανεξάρτητα με το αν ήταν εύκολη η δουλειά που είχε να κάνει ένας υπολογιστής για να βρει αυτή τη λύση. Είναι μία γλώσσα για να λύνεις προβλήματα και όχι για προγραμματισμό. . . υπάρχει βέβαια και το APL-WEB. Αλλά θα ήθελα να πω κάτι ακόμα για την IMP. Το τρίτο κακό αυτής της γλώσσας ήταν ότι αν έκανες ένα λάθος, ο συμπιλιστής έκανε κύκλους και σταματούσε στο πρώτο λάθος λέγοντας «ERROR, ERROR, ERROR» και μετά τερμάτιζε: θα έπρεπε να βρεις ποιό ήταν το λάθος. Δεν ήταν καμία καταπληκτική γλώσσα αλλά ούτε και ο συμπιλιστής ήταν καταπληκτικός. Όμως ήταν αυτή που προτιμούσα γιατί ταίριαζε απόλυτα με το λειτουργικό σύστημα. Οι *παρατάξεις* (arrays) ονομαζόταν έτσι που ήταν εύκολο να τα δεις στον debugger και μπορούσες να δείς ποιές θέσεις μνήμης καταλάμβαιναν, ήξερες πως πήγαιναν τα πράγματα και μπορούσες να τρέξεις το πρόγραμμα σου αξιόπιστα γιατί η IMP έδενε καλά με το λειτουργικό. Αυτό δεν μπορούσες να το κάνεις με καμία άλλη γλώσσα. Μπορούσες να γράψεις με μία καλύτερη γλώσσα αλλά θα τελείωνες το πρόγραμμά σου δύο βδομάδες αργότερα από ότι αν χρησιμοποιούσες την IMP.

**CLB:** Χρησιμοποιήθηκε η IMP στο Stanford;

**Knuth:** Ήταν σε ένα εργαστήριο στο Princeton. Ένα χρόνο πριν έρθω στο Stanford, εργαζόμουν εκεί σε ένα απόρρητο ερευνητικό πρόγραμμα κρυπτανάλυσης.

**CLB:** Πείτε μας τώρα για το άλλο σας νέο βιβλίο, το Stanford GraphBase.

**Knuth:** Αυτό το βιβλίο είναι για δύο είδη ανθρώπων. Έχει ένα ερευνητικό στόχο: εκείνοι που εργάζονται στη μελέτη νέων αλγορίθμων για συνδυαστικά προβλήματα χρειάζονται ένα συγκεκριμένο σύνολο *δεδομένων δοκιμής* (test data) για ελέγχους ταχύτητας. Καθώς προετοίμαζα τον τόμο IV της σειράς The Art of Computer Programming, αποφάσισα να κάνω διαθέσιμα σ' όλον τον κό-

σμο τα στοιχεία και τα παραδείγματα που χρησιμοποιούσα. Υπήρχε ανάγκη για κάποιους τυποποιημένους ελέγχους ταχύτητας, και όλα πρέπει να είναι διαταγμένα έτσι που να είναι χρήσιμα με πολλούς τρόπους. Έτσι τώρα έχω μία συλλογή από χιλιάδες τυποποιημένα σύνολα δεδομένων και οποισδήποτε στην Πολωνία μπορεί να έχει ακριβώς τα ίδια σύνολα δεδομένων με κάποιον στην αμερική ή στην Κίνα. Είναι ανεξάρτητη υπολογιστικής πλατφόρμας και κανείς μπορεί να τα αποκτήσει μέσω Internet

Ο δεύτερος λόγος για τον οποίο μου αρέσει το GraphBase είναι γιατί αποτελεί ένα παράδειγμα προγραμματισμού με το CWEB — είναι στην πραγματικότητα 32 παραδείγματα χρήσης του CWEB. Είναι 32 μικρά προγράμματα που δείχνουν το στυλ προγραμματισμού που προτιμώ. Τα παραδείγματα είναι σαν μικρά δοκίμια, μικρές ιστορίες προγραμμάτων που είναι και ευχάριστο να τα διαβάσει κανείς.

**CLB:** Σε τι υπολογιστικό περιβάλλον (υπολογιστή και λογισμικό) δουλεύετε τώρα;

**Knuth:** Χρησιμοποιώ την CWEB για προγραμματισμό. Χρησιμοποιώ επίσης πάρα πολύ τον Emacs, και την γλώσσα METAPOST για τεχνικό σχέδιο. Πρόκειται για μία καινούργια γλώσσα που έφτιαξε ο John Hobby που πιστεύω πως σύντομα θα διατίθεται μέσω του Internet<sup>2</sup>. Βασίζεται στο METAFONT. 75% του κώδικά της είναι δικός μου απο το METAFONT αλλά είναι τροποποιημένη ώστε να παράγει PostScript. Μου αρέσει πάρα πολύ.

Επίσης χρησιμοποιώ MATHEMATICA. Οι άνθρωποι της MAPLE προσπαθούν να με πείσουν να γυρίσω στο MAPLE, ένα πολύ καλό πρόγραμμα. Αυτή τη στιγμή όμως προτιμώ το MATHEMATICA γιατί δεν χρειάζεται να δηλώνεις τους παλλαπλασιασμούς: μπορείς να πεις '2X' αντί για '2 \* X'. Επίσης το εγχειρίδιο του MATHEMATICA είναι εξαιρετικό.

**CLB:** Σας αρέσει το στυλ του Wolfram;

**Knuth:** Ιδιαίτερα το ευρετήριο όρων... μπορείς εύκολα να βρεις τα πάντα στο βιβλίο. Στην πρώτη έκδοση όταν είχα ένα πρόβλημα να λύσω κοιτούσα στο ευρετήριο και σχεδόν πάντα με παρέπεμπε στη σωστή σελίδα. Υπήρξαν μόνο μιά δυό φορές που δεν βρήκα τη λέξη που έψαχνα και τη σημείωσα εγώ στο ευρετήριο για να την βρώ εύκολα την επόμενη φορά που θα τη χρειαζόμουν. Στη δεύτερη έκδοση αυτά είχαν όλα διορθωθεί, παρόλο που δεν τα ανέφερα εγώ σε κανέναν.

**CLB:** Θα ήθελα τώρα να μου πείτε τις εντυπώσεις σας για κάποιες ερευνητικές περιοχές και αν έχετε εργασθεί πάνω σε αυτές. Η πρώτη είναι οι γενετικοί αλγόριθμοι. Πως σας φαίνεται η γενική ιδέα ότι αντί να αποφασίζει ο άνθρωπος για τον αλγόριθμο αποφασίζει μια μηχανή ...

<sup>2</sup> Σ.Ε.Σ. Πράγμα που τώρα συμβαίνει.

**Knuth:** Σχεδιάζω να πειραματιστώ πολύ με αυτό το θέμα όταν θα φτάσω στον τέταρτο τόμο. Υπάρχει γενετική αναπαραγωγή, υπάρχει η μέθοδος της *simulated annealing*, έχουν αναπτυχθεί και άλλες στρατηγικές. Εγώ έχω μία μέθοδο στο βιβλίο *Stanford GraphBase* που την έχω ονομάσει «*stratified greed*». Όλες αυτές οι τεχνικές έχουν στόχο το ίδιο είδος προβλημάτων και θέλω να κάνω πολλές δοκιμές· μερικές μπορούν να δουλέψουν καλύτερα σε ένα πρόβλημα από ότι σε ένα άλλο, και θέλω να αναπτύξω μία διάσθηση για όλα αυτά. Κάποια προβλήματα ανήκουν με μία φυσική έννοια στα νευρωνικά δίκτυα... οι γενετικοί αλγόριθμοι μάλλον θα τα πάνε καλά σε θέματα που έχουν να κάνουν με αναγνώριση φωνής, και κάποιοι λένε ότι θα κάνουν και για χρηματιστηριακές προβλέψεις ή άλλα τέτοια θέματα. Κατα κάποιο τρόπο όσο κοντύτερα είναι το πρόβλημα σε φυσικές διεργασίες τόσο καλύτερα αναμένεται να δουλεύει ένας γενετικός αλγόριθμος, ενώ όσο κοντύτερα είναι το πρόβλημα στη θεωρία αριθμών ή κάτι τεχνητό, τόσο περιμένει κανείς κάποια άλλη προσέγγιση να δουλεύει καλύτερα. Είναι δύσκολο να καταλάβει κανείς πως θα δουλέψουν αυτές οι μέθοδοι σε σχέση με το μέγεθος του προβλήματος. Μπορεί σε ένα μικρό πρόβλημα να τα πάνε καλά και να αποτύχουν σε ένα λίγο μεγαλύτερο. Η ανάποδα βέβαια.

**CLB:** Από ό,τι φαίνεται έχετε πολλά χρόνια δουριάς ακόμα μπροστά σας.

Knuth Η *Stanford GraphBase* μου παρέχει απεριόριστο αριθμό προβλημάτων. Διαβάζω τι ισχυρίζονται άλλοι για τις δικές τους μεθόδους, αλλά τις δοκιμάζω και μόνος μου. Η πρωτότυπη δουλειά που κάνω στο *The Art Of Computer Programming* είναι να πάρω τις μεθόδους δύο διαφορετικών ανθρώπων και να αναλύσω την μέθοδο *A* από τη σκοπιά του *B* και αντίστροφα. Αυτοί τα βλέπουν μόνο από τη δική τους σκοπιά οπότε εγώ προσπαθώ να συμπληρώσω κενά...

**CLB:** Τί έχετε να πείτε για τον *αντικειμενοστραφή* προγραμματισμό;

**Knuth:** Πάντα είχα στο νου μου ένα τέτοιο είδος προγραμματισμού αλλά ποτέ δεν χρησιμοποίησα γλώσσες που απαιτούσαν κάτι τέτοιο. Συνήθως είχα εγώ απαιτήσεις από τις γλώσσες. Τώρα οι γλώσσες μπορούν να σου βρουν τα λάθη σου και έτσι είναι πιο εύκολο να κρύψεις πληροφορίες από το ένα μέρος του προγράμματος στο άλλο. Στα δικά μου προγράμματα με παλιότερες γλώσσες δεν χρησιμοποιούσα κάτι που δεν έπρεπε να χρησιμοποιήσω· έπρεπε να επιβάλλω στον εαυτό μου να χρησιμοποιώ αυτούς τους κανόνες. Μπορούσα και το έκανα. Δεν υπήρχαν προγράμματα που δε μπορούσα να γράψω... αλλά τα νέα εργαλεία και αυτά βοηθάνε.

Το πρόβλημα που έχω με αυτό το θέμα σήμερα είναι ότι... η *C++* είναι πολύ περίπλοκη. Για την ώρα είναι αδύνατο για μένα να γράψω κώδικα που θα δουλεύει σε πολλές υπολογιστικές πλατφόρμες, εκτός και αν αποφύγω όλα τα εξωτικά εργαλεία. Όποτε αυτοί που σχεδίασαν την *C++* όταν είχαν δύο αντίθετες απόψεις για το πως να λύσουν ένα θέμα, έλεγαν εντάξει ας συμπεριλάβουμε και τις δύο. Έτσι η γλώσσα είναι πολύ «*baroque*» κατά τη γνώμη μου. Αλλά κάθε

χρήστης της C++ έχει ένα υποσύνολο της γλώσσας που χρησιμοποιεί οπότε αυτό είναι καλό. Το CWEB βέβαια υποστηρίζει και την C++ αλλά και την C.

**CLB:** Πείτε μας τις σχέσεις σας για περιοχές όπως θεωρία του χάους, fractals (χώροι μη ακέραιας διάστασης) κ.λπ. Η ασάφεια που τα διακρίνει φαίνεται να είναι σε αντίθεση με τις περιοχές που ασχοληθήκατε στο παρελθόν.

**Knuth:** Αρχικά έκανα κάποια δουλειά στους χώρους μη ακέραιας διάστασης, πρόκειται για μία καταπληκτική αφαίρεση. Μπορούν να κατασκευαστούν μοντέλα που πριν δεν μπορούσαμε να σκεφτούμε και ταιριάζουν με πολλά πράγματα στη φύση όταν το ιδιαίτερο χαρακτηριστικό τους έχει να κάνει με ένα θέμα που επαναλαμβάνετε συνεχώς σε διαφορετική κλίμακα. Για παράδειγμα αν κανείς μεγενθύνει το περίγραμμα της ακρογιαλιάς πάλι θα βλέπει το ίδιο σχήμα, και πολλά πράγματα έχουν αυτή την ιδιότητα. Η φύση έχει τους δικούς της επαναληπτικούς αλγορίθμους για να παράγει πράγματα όπως σύννεφα, ελβετικό τυρί κ.λπ. Τώρα έχουμε μαθηματικές τεχνικές για να κατανοούμε αυτές τις διαδικασίες που προχωράνε πέρα από τις διαφορικές εξισώσεις που έχουμε συνηθίσει να χρησιμοποιούμε από τους προηγούμενους αιώνες. Έχουμε ένα ολοκαίνουργιο εργαλείο να χρησιμοποιούμε αν και δεν έχω αναπτυγμένη διαίσθηση για αυτές τις τεχνικές. Γνωρίζω τα όρια της διαίσθησής μου· μερικά προβλήματα μπορώ να τα λύσω καλά αλλά ξέρω ότι άλλοι μπορούν να δουν αμέσως κάτι που εμένα θα μου πάρει πολύ ώρα να δω... δεν είναι το δυνατό μου σημείο.

**CLB:** Σε ποίο βαθμό έχετε παρακολουθήσει τις εξελίξεις στην τεχνητή νοημοσύνη; Το τρίτο σας πρόγραμμα ήταν ένα πρόγραμμα τρίλιζας που μάθαινε από τα λάθη του, και το Stanford είναι ένα από τα κορυφαία ιδρύματα στην έρευνα για την τεχνητή νοημοσύνη...

**Knuth:** Η τεχνητή νοημοσύνη σχετίζεται άμεσα με τον τέταρτο τόμο· οι ερευνητές του θέματος χρησιμοποιούν τις συνδυαστικές τεχνικές που μελετώ, οπότε υπάρχουν αρκετές δημοσιεύσεις που σχετίζονται με το θέμα. Δουλειά μου είναι να συγκρίνω τα αποτελέσματα στην τεχνητή νοημοσύνη με αυτά της κοινότητας των ηλεκτρολόγων μηχανικών, και άλλων ειδικοτήτων· κάθε κοινότητα έχει και ένα διαφορετικό τρόπο προσέγγισης των προβλημάτων. Προσπαθώ να διαβάσω αυτά τα πράγματα και να ενοποιήσω τις ιδέες. Οι πιο δύσκολες εφαρμογές και τα πιο απαιτητικά προβλήματα στην ιστορία των υπολογιστών είναι στην τεχνητή νοημοσύνη. Η τεχνητή νοημοσύνη είναι η πιο παραγωγική πηγή νέων τεχνικών στην επιστήμη των υπολογιστών. Μας οδήγησε σε πολλές και σημαντικές προόδους όπως δομές δεδομένων και ανάλυση λιστών... Πολλά από τα καλύτερα παραδείγματα για debugging και για να κάνεις το λογισμικό να δουλεύει, όλα τα συστήματα συμβολικής άλγεβρας που φτιάχτηκαν, οι πρώτες μελέτες γραφικών και τεχνητής όρασης, κ.λπ. όλα έχουν πολύ βαθιές ρίζες στην τεχνητή νοημοσύνη.

**CLB:** Δηλαδή δεν είστε από αυτούς που υποτιμούν ότι έγινε σε αυτή την περιοχή...

**Knuth:** Καθόλου. Αυτό που έγινε ήταν ότι πολλοί πίστεψαν ότι η τεχνητή νοημοσύνη θα γινόταν πανάκεια. Είναι κάτι σαν να έχεις μία εταιρεία που πέφτουν οι μετοχές της επειδή οι προβλέψεις για τα κέρδη της ήταν 18% ενώ τα πραγματικά της κέρδη ήταν 15%. Αυτό συνέβη γιατί πίστεψαν ότι μια μεθοδολογία θα έλυνε τα πάντα. Κατά πάσα πιθανότητα αυτό θα συμβεί με όλα τα θέματα που τώρα εντυπωσιάζουν: οι άνθρωποι θα καταλάβουν ότι με αυτά δεν μπορούν να απαντήσουν τα πάντα. Πολλά προβλήματα είναι τόσο δύσκολα που ποτέ δεν θα βρούμε μια καλή λύση για αυτά. Οι άνθρωποι απογοητεύονται όταν δεν βρίσκουν την «πηγή της νιότης»...

**CLB:** Αν τελειώνετε τώρα το πανεπιστήμιο ή το διδακτορικό σας τι είδους έρευνα θα διαλέγατε; Ή μήπως δεν θα διαλέγατε καν έρευνα;

**Knuth:** Νομίζω πως τα πιο ενδιαφέροντα θέματα αυτή τη στιγμή στην επιστήμη των υπολογιστών είναι στη ρομποτική και στη βιοχημεία. Για παράδειγμα ή ρομποτική είναι καταπληκτικό θέμα: να φτιάχνεις αντικείμενα που κινούνται και αλληλοεπικοινωνούν μεταξύ τους. Το Stanford έχει ένα μεγάλο εργαστήριο ρομποτικής και το σχέδιό μας είναι να φτιάξουμε ένα νέο κτήριο που θα έχει εκατό ρομπότ που θα περπατάνε στους διαδρόμους για να ανεβαίνει έτσι το ενδιαφέρον των φοιτητών. Θα μας πάρει περίπου δύο με τρία χρόνια μέχρι να μεταφερθούμε στο νέο κτήριο. Μόνο που θα βλέπεις ρομποτ εκεί θα σου έρχονται ιδέες για καινούργια πράγματα. Αυτά τα προβλήματα επίσης παράγουν ωραίες μαθηματικές και θεωρητικές ερωτήσεις. Και υψηλού επιπέδου γραφικά εργαλεία, μια περιοχή που έχει επίσης ένα τεράστιο αριθμό καταπληκτικών ιδεών. Ναι, θα ήθελα να ασχοληθώ με αυτό.

**CLB:** Γιατί αναφέρατε την βιοχημεία;

**Knuth:** Υπάρχουν εκατομύρια άλυστα προβλήματα εκεί. Η βιολογία είναι πολύ ψηφιακή, πολύ περίπλοκη και απείρως χρήσιμη. Το πρόβλημα με τη βιολογία είναι ότι αν πρέπει να εργαστείς ως βιολόγος είναι βαρετό. Τα πειράματα μπορεί να κρατήσουν και τρία χρόνια και ξαφνικά μπορεί να πέσει το ρεύμα και τα πάντα πεθαίνουν! Πάλι από την αρχή. Στους υπολογιστές φτιάχνεις τον δικό σου κόσμο. Οι βιολόγοι πάντως αξίζουν συγχαρητήρια που τα καταφέρνουν.

Είναι δύσκολο για μένα να πω με σιγουριά μετά από 50 ακόμα χρόνια εκρηκτικής ανάπτυξης των υπολογιστών ότι θα συνεχίσει να έχει ενδιαφέροντα προβλήματα η επιστήμη των υπολογιστών και δεν θα είναι απλά βελτιώσεις ήδη γνωστών πραγμάτων. Ίσως όλα τα ενδιαφέροντα πράγματα να έχουν ανακαλυφθεί. Ίσως και να κάνω λάθος αλλά δεν μπορώ να προβλέψω μία ατελείωτη ανάπτυξη. Δεν μπορώ να είμαι τόσο σίγουρος για τους υπολογιστές όσο μπορώ να είμαι για την βιολογία. Αυτή έχει ενδιαφέροντα προβλήματα για τουλάχιστον άλλα 500 χρόνια, είναι σε αυτό το επίπεδο.

**CLB:** Η χρήση του διαδικτύου (Internet) είναι εκρηκτικά αυξανόμενη τώρα, με όλο και περισσότερες. . .

**Knuth:** Μιά μέρα θα αναρωτηθούμε ποιός πληρώνει για όλα αυτά!

**CLB:** Το χρησιμοποιήτε; Γνωρίζω ότι το χρησιμοποιούσατε στο παρελθόν.

**Knuth:** Δεκαπέντε χρόνια χρησιμοποιούσα ηλεκτρονικό ταχυδρομείο στο ARPANET και στο Internet. Κάποια στιγμή τον Ιανουάριο του 1990 σταμάτησα γιατί μού έπαιρνε πολύ χρόνο να βρω άκρη ανάμεσα σε όλη αυτή τη σαβούρα. Δεν έχω ηλεκτρονική διεύθυνση. Όσοι προσπαθούν να μου γράψουν παίρνουν μία απάντηση που λέει «Ο Καθηγητής Knuth έχει σταματήσει να διαβάζει ηλεκτρονικό ταχυδρομείο· μπορείτε να του γράψετε στην τάδε διεύθυνση»<sup>3</sup>.

Είναι αδύνατο να κλείσεις το ηλεκτρονικό ταχυδρομείο! Στέλνεις ένα μήνυμα σε κάποιον και απαντάει «Σας ευχαριστώ» και εσύ ξανααπαντάς «ΟΚ, σας ευχαριστώ που με ευχαριστήτε. . .»

Το ηλεκτρονικό ταχυδρομείο είναι πολύ καλό για αυτούς που θέλουν να είναι στην αιχμή των πραγμάτων. Αλλά εγώ δουλεύω στην άλλη άκρη. Κοιτάω ιδέες προσεκτικά και προσπαθώ να τις γράψω. . . κινούμε αργά ανάμεσα σε πράγματα που έχουν κάνει άλλοι και προσπαθώ να τα οργανώσω. Αλλά δεν ξέρω τι συμβαίνει αυτο τον μήνα.

Έτσι λοιπόν δεν διαβάζω ηλεκτρονικό ταχυδρομείο πια, εκτος από σπάνιες περιπτώσεις όπως όταν πηγαίνω ένα ταξίδι π.χ. στο Ισραήλ και θέλω να κανονίσω πράγματα της τελευταίας στιγμής. Ξέρω πως να χρησιμοποιώ το ηλεκτρονικό ταχυδρομείο του Emacs αλλά δεν θέλω να γίνω καλός σε αυτό.

**CLB:** Έχετε πολλά ενδιαφέροντα έξω από τους υπολογιστές και τα μαθηματικά—μουσική, θρησκεία, συγγραφή. Η μουσική είναι μία δημιουργική σας διεξοδος, ένας τρόπος για να διασκεδάσετε ή μία θρησκευτική διεξοδος;

**Knuth:** Αυτή τη στιγμή είναι για διασκέδαση. Μου αρέσει να έρχονται φίλοι στο σπίτι και να παίζουμε πιάνο παρέα. Αν μπορούσα θα το έκανα κάθε βδομάδα. Ελπίζω να ζήσω αρκετά ώστε να τελειώσω το έργο της ζωής μου την σειρά βιβλίων The Art Of Computer Programming και μετά ίσως ασχοληθώ με την σύνθεση μουσικής. Ένα όνειρο είναι. . . κακή μουσική βέβαια.

**CLB:** Έχετε γράψει κάποιες συνθέσεις ήδη, σωστά;

**Knuth:** Ναι, αλλά ήταν κυρίως ένα άθροισμα από μουσικά θέματα άλλων ανθρώπων. Όταν ήμουν φοιτητής είχα γράψει μία μικρή μουσική κωμωδία που λεγόταν «Nebbishland». Αυτή η δουλειά είχε διάρκεια περίπου 10 λεπτά, αλλά και η μουσική και οι στοίχοι ήταν δικό μου.

<sup>3</sup> Σ.Ε.Σ. Αυτό δεν είναι αλήθεια, αλλά ο Knuth δε δημοσιοποιεί την ηλεκτρονική του διεύθυνση.

**CLB:** Έχετε κρατήσει την παρτιτούρα;

**Knuth:** Ναι... ή μάλλον όχι! Την έχασα. Έχω μόνο ένα μέρος. Πιστεύω να την βρώ κάπου. Φτιάχνω ένα αρχείο τώρα στον υπολογιστή για όλα τα πράγματα που έχω στο σπίτι μου.

**CLB:** Παίζατε καθόλου με την τεχνολογία MIDI, ή κρατήσατε επίτηδες αποστάσεις από αυτήν.

**Knuth:** Μου αρέσει πολύ. Αγόρασα τα περασμένα Χριστούγεννα ένα συνθεσάιζερ για τον γιό μου και έπαιζα με τις ώρες. Μου άρεσε πάρα πολύ. Παλιότερα είχα παίξει σε ένα συνθεσάιζερ Kurzweil στο σπίτι του Marvin Minsky που ήταν μία προσομοίωση πιάνου. Τελευταία ένας φίλος πήγε στην Αγγλία για τρία χρόνια και δεν ήθελε να πάρει το πιάνο του μαζί του, έτσι αγόρασε ένα Yamaha με έξι φωνές. Όταν τον επισκέφτηκα πέρασα τρεις υπέροχες μέρες παίζοντας τα κομμάτια που ήξερα να παίζω στο πιάνο δοκιμάζοντας διάφορες φωνές. Η προσομοίωση πιάνου που είχε έμοιζε με τσέμπαλο αλλά τα πλήκτρα ήταν ευαίσθητα στο πόσο τα πατούσες οπότε μπορούσες να παίζεις δυνατά ή απαλά πράγμα που δεν μπορείς να κάνεις σε ένα πραγματικό τσέμπαλο. Αυτα τα συνθεσάιζερ είναι πραγματικά πολύ καλά.

**CLB:** Πότε βγήκατε στη σύνταξη από το Stanford;

**Knuth:** Φέτος. Είχα άδεια για δύο χρόνια ώσπου να μπορώ και τυπικά να βγώ στη σύνταξη. Άτυπα είχα βγεί από το 1990 την ίδια μέρα που σταμάτησα να διαβάζω ηλεκτρονικό ταχυδρομείο. Είχα ανακοινώσει τα σχέδιά μου τρία χρόνια νωρίτερα. Διαπίστωσα ότι αυτό που ήθελα να κάνω στη ζωή μου ήταν να τελειώσω το *The Art Of Computer Programming*. Είχα προβλέψει ότι θα μου πάρει 20 χρόνια πλήρους απασχόλησης. Εάν συνεχίζα να κάνω αυτά που έκανα θα μου έπαιρνε 40 ή και 50 χρόνια. Με άλλα λόγια δεν προχωρούσε, και έμενα συνεχώς πίσω. Έτσι αποφάσισα να βγω στη σύνταξη. Βέβαια δεν μου αρέσει που άφησα ότι άλλο έκανα αλλά υπήρχαν και πράγματα που πολύ χάρηκα που απαλλάχτηκα από αυτά, όπως το να γράφω προτάσεις για ερευνητικά προγράμματα.

**CLB:** Έπρεπε να γράφετε και προτάσεις για ερευνητικά προγράμματα; Νόμιζα ότι θα ήσασταν απαλλαγμένος από αυτό.

**Knuth:** Έχετε χιούμορ βλέπω. Δεν πρέπει να το κάνω πια αλλά σαν καθηγητής για να έχω καλο εξοπλισμό για τους φοιτητές μου ή να μπορώ να έχω επισκέπτες για ερευνητικά προγράμματα, έπρεπε να βρίσκω χορηγούς. Είναι πολύ δουλειά να ζητιανεύεις λεφτά. Το System Development Foundation μου είχαν πει ότι θα μου έδιναν ένα εκατομμύριο δολάρια για να τελειώσω το  $\TeX$  ώστε να πιστρέψω στη συγγραφή της σειράς *The Art Of Computer Programming*.

**CLB:** Τα πήρατε;

**Knuth:** Βεβαίως, αλλά πάλι πήρε πάρα πολλά χρόνια για να τελειώσω το T<sub>E</sub>X. Αποφάσισα έτσι ότι ο μόνος τρόπος για να τελειώσω τη σειρά The Art Of Computer Programming ήταν να αφιερωθώ πλήρως σε αυτό. Ήταν δύσκολο να προσαρμοστώ τα πρώτα 2–3 χρόνια. Τώρα όμως νοιώθω εντάξει.

Δίνω διαλέξεις στο Stanford κάθε μήνα με γενικό τίτλο Computer Musings. Σχεδιάζω να συνεχίσω να δίνω τέτοιες ομιλίες σε θέματα και ιδέες που βρίσκω ενδιαφέρουσες για τα επόμενα 20 χρόνια. Παρουσιάζω προβλήματα που δεν μπορώ να λύσω ώστε κάποιος να τα λύσει αντί για εμένα. Αν δεν μπορώ να λύσω ένα πρόβλημα για 2 ώρες το δίνω σε κάποιον άλλον να το λύσει αλλιώς πάλι μένω πίσω. Καθώς γράφω το βιβλίο κινούμε από θέμα σε θέμα παρίπου ανά τρεις βδομάδες.

**CLB:** Είσατε ιδιαίτερα γνωστός για αυτά που έχετε γράψει και για την έρευνά σας· σας άρεσε όμως η διδασκαλία και η συναναστροφή με τους φοιτητές;

**Knuth:** Είχαμε τους καλύτερους φοιτητές του κόσμου. Ακόμα συναναστρεφόμεθα με τους φοιτητές μέσω των διαλέξεών μου αλλά δεν μπορώ να συγκρατήσω τα ονόματά τους πιά. Αυτό είναι ένα πρόβλημα.

Ας υποθέσουμε ότι δίνω μια διάλεξη της σειράς Computer Musings και διατυπώνω ένα ανοιχτό πρόβλημα. Και κάποιος από το ακροατήριο το λύνει, γράφει την διατριβή του, τελειώνει μέσα σε 2 βδομάδες και έρχετε και μου το δείχνει. Θα ενδιαφερθώ για το θέμα, θα το διαβάσω και θα υπογράψω τη διατριβή του... όμως αυτός είναι ο μόνος τρόπος επαφής. Είχα 28 φοιτητές που πήραν διδακτορικό μαζί μου και μάλλον τόσοι θα παραμείνουν εκτός και αν συμβεί κάτι πολύ γρήγορα στις διαλέξεις μου όπως ανέφερα πριν.

**CLB:** Real-time διδακτορικά! Τι διαφορές έχετε παρατηρήσει στους φοιτητές με την πάροδο όλων αυτών των ετών;

**Knuth:** Υπάρχει μιά πολύ σημαντική αλλαγή. Στη δεκαετία του 70 οι φοιτητές ενδιαφερόταν πολύ για μουσική. Το πρώτο πράγμα που τους ρωτούσαμε ήταν «τι μουσικό όργανο παίζετε;». Είχαμε πολλά μουσικά σχήματα κ.λπ. Τώρα σχεδόν κανένας δεν ενδιαφέρετε για την μουσική. Δεν ξέρω αν άλλαξαν οι φοιτητές που διαλέγουν τηνεπιστήμη των υπολογιστών ή το ίδιο συμβαίνει με όλους τους φοιτητές τώρα. Αν ρωτήσεις τώρα τους φοιτητές της επιστήμης των υπολογιστών ποιά είναι τα χόμπυ τους το πιο πιθανό είναι να σου πουν «η ποδηλασία». Πρόσφατα είχαμε κάποιον που έπαιζε φουσαρμόνικα αλλά δεν υπήρχε κανείς άλλος μουσικός.

**CLB:** Καμιά αλλαγή στην ποιότητα των φοιτητών;

**Knuth:** Οχι...εκτός από το ότι δεν ξέρουν τόσο καλά μαθηματικά όσο ήξεραν παλιά. Πρέπει να τους προετοιμάζουμε για αυτό με ειδικά μαθήματα ακόμα και σε ένα πανεπιστήμιο όπως το Stanford.

**CLB:** Αλλαγές στον χώρο; Με τόσο κόσμο και πρόοδο που έχει σήμερα ο κλάδος έχει αλλάξει καθόλου;

**Knuth:** Είναι πολύ διαφορετικός σήμερα. Υπάρχει επίσης ανταγωνισμός· είναι πιο δύσκολα τα πράγματα τώρα από ότι ήταν στην εποχή μου. Όταν άρχισα ήταν πολύ πιο εύκολο να ανακαλύψεις κάτι καινούργιο από ότι είναι σήμερα, όταν έχεις χιλιάδες έξυπνους ανθρώπους που κάνουν τόσα σπουδαία πράγματα. Τότε ίσως να υπήρχαν δέκα πολύ καλά διδακτορικά. Τώρα δεν μπορεί κανείς να παρακολουθήσει όλες τις εξελίξεις.

Ανεξάρτητα από τον κλάδο στον οποίο είσαι, όλοι δυσκολεύονται να παρακολουθούν τις εξελίξεις. Κάθε πεδίο στενεύει συνεχώς αφού κανείς δεν μπορεί να ξέρει όλη την περιοχή του πια. Καθένας διαλέγει δύο μικρές περιοχές μέσα στον κλάδο και μαθαίνει αυτές τις περιοχές· αν κάποιος ξέρει ξέρει την περιοχή Α και Β και κάποιος άλλος την Β και την Γ και ένας άλλος την Γ και την Δ τότε ο χώρος μένει ικανοποιητικά συννεκτικός παρόλο που μεγαλώνει.

**CLB:** Βλέπετε τον εαυτό σας σαν τον τελευταίο *αναγεννητή* της επιστήμης των υπολογιστών;

**Knuth:** Δεν έχω τόσο ευρεία γνώση όσο νομίζετε—δουλεύω σε ένα θέμα κάθε φορά. Νομίζω όμως πως μαθαίνω γρήγορα· μπορώ να γίνω ειδικός σε ένα θέμα αμέσως. Μάξευα διάφορα πράγματα 30 χρόνια τώρα ώστε να μπορώ να διαβάζω την βιβλιογραφία σε κάθε θέμα σε «batch mode»—χωρίς να αλλάζω θέματα συνέχεια. Μπορώ να αποροφήσω ένα θέμα τοπικά και να γίνω καλός σε αυτό για λίγο... αλλά μετά μην μου πείτα να κάνω αυτό που έκανα λίγους μήνες πριν. Επίσης έχω πολλούς ανυρώπους που διορθώνουν τα λάθη μου.

**CLB:** Η τελευταία μου και λιγότερο καλή για σας ερώτηση... ποιο είναι το τρέχον πλάνο για την συμπλήρωση και των επτά τόμων του *The Art Of Computer Programming*;

**Knuth:** Θα μαζεύουμε τέσσερα από αυτά πριν βγάλουμε τα δύο πρώτα κάθε χρόνο· θα κρατάμε κάποια στο pipeline! Να τα αναμένετε το 1995 ή 1996· πρόκειται για beta-test εκδόσεις των πραγματικών βιβλίων. Πιστεύω πως θα τελειώσω τον τέταρτο τόμο το 2003, τον πέμπτο το 2008, μετά θα επανεκδοθεί ο πρώτος, δεύτερος και τρίτος τόμος... Θα υπάρχει μία συνοπτική έκδοση των τόμων από τον πρώτο μέχρι τον πέμπτο.

**CLB:** Πως θα ήταν η καριέρα σας και η ζωή σας αν δεν είχατε ανακοινώσει αυτο το επτάτομο έργο;

**Knuth:** Στην αρχή δεν ανακοίνωσα κάτι τέτοιο. Πίστευα ότι θα έγραφα ένα βιβλίο. Αλλά αν δεν το έκανα αυτό πιστεύω ότι πάλι θα είχα πολύ γράψιμο. Από ότι φαίνεται σε όλη την πορεία μου άρεσε να προσπαθώ να εξηγώ στον αναγνώστη. Όταν ήμουν στο γυμνάσιο ήμουν ο συντάκτης της σχολικής εφημερίδας·

στο πανεπιστήμιο ήμουν ο συντάκτης ενός περιοδικού. Πάντα μου άρεσε να παίζω με τις λέξεις.

---

# The L<sup>A</sup>T<sub>E</sub>X3 Programming Language— a proposed system for T<sub>E</sub>X macro programming

---

David Carlisle, Chris Rowley και Frank Mittelbach<sup>§¶</sup>

*L<sup>A</sup>T<sub>E</sub>X3 project*

*email: latex-l@urz.uni-heidelberg.de*

## 1. Introduction

This paper describes the conventions for a T<sub>E</sub>X-based programming language which is intended to provide a more consistent and rational environment for the construction of large scale systems, such as L<sup>A</sup>T<sub>E</sub>X, using T<sub>E</sub>X macros.

Variants of this language have been in use by The L<sup>A</sup>T<sub>E</sub>X3 Project Team since around 1990 but the syntax specification to be outlined here should *not* be considered final. This is an experimental language thus many aspects, such as the syntax conventions and naming schemes, may (and probably will) change as more experience is gained with using the language in practice.

The next section shows where this language fits into a complete T<sub>E</sub>X-based document processing system. We then describe the major features of the syntactic structure of command names, including the argument specification syntax used in function names.

The practical ideas behind this argument syntax will be explained, together with the semantics of the expansion control mechanism and the interface used to define variant forms of functions. The paper also discusses some advantages of the syntax for parameter names.

---

<sup>§</sup> This paper is based on a talk given by David Carlisle in San Francisco, July 1997, but it describes the work of several people: principally Frank Mittelbach and Denys Duchier, together with Johannes Braams, David Carlisle, Michael Downes, Alan Jeffrey, Chris Rowley and Rainer Schöpf.

<sup>¶</sup> Το κείμενο αυτό εμφανίστηκε σε τεύχος του περιοδικού TUGboat.

As we shall demonstrate, the use of a structured naming scheme and of variant forms for functions greatly improves the readability of the code and hence also its reliability. Moreover, experience has shown that the longer command names which result from the new syntax do not make the process of *writing* code significantly harder (especially when using a reasonably intelligent editor).

The final section gives some details of our plans to distribute parts of this system during the next year. More general information concerning the work of the L<sup>A</sup>T<sub>E</sub>X3 Project can be found in [4].

## 2. Languages and interfaces

It is possible to identify several distinct languages related to the various interfaces that are needed in a T<sub>E</sub>X-based document processing system. This section looks at those we consider most important for the L<sup>A</sup>T<sub>E</sub>X3 system.

**Document mark-up** This comprises those commands (often called tags) that are to be embedded in the document (the `.tex` file).

It is generally accepted that such mark-up should be essentially *declarative*. It may be traditional T<sub>E</sub>X-based mark-up such as L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>, as described in [3] and [2], or SGML-based mark-up such as XML.

One problem with more traditional T<sub>E</sub>X coding conventions (as described in [1]) is that the names and syntax of T<sub>E</sub>X's primitive formatting commands are ingeniously designed to be 'natural' when used directly by the author as document mark-up or in macros. Ironically, the ubiquity (and widely recognised superiority) of logical mark-up has meant that such explicit formatting commands are almost never needed in documents or in author-defined macros. Thus they are used almost exclusively by T<sub>E</sub>X programmers to define higher-level commands; and their idiosyncratic syntax is not at all popular with this community. Moreover, many of them have names that could be very useful as document mark-up tags were they not pre-empted as primitives (e.g., `\box` or `\special`).

**Designer interface** This relates a (human) typographic designer's specification for a document to a program that 'formats the document'. It should ideally use a declarative language that facilitates expression of the relationship and spacing rules specified for the layout of the various document elements.

This language is not embedded in document text and it will be very different in form to the document mark-up language. For SGML-based systems the DSSSL language may come to play this role. For L<sup>A</sup>T<sub>E</sub>X, this level was almost completely missing from L<sup>A</sup>T<sub>E</sub>X 2.09; L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> made some improvements in this area but it

---

is still the case that implementing a design specification in  $\LaTeX$  requires far more ‘low-level’ coding than is acceptable.

**Programmer interface** This language is the implementation language in which the basic typesetting functionality is implemented, building upon the primitives of  $\TeX$  (or a successor program). It may also be used to implement the previous two languages ‘within’  $\TeX$ , as in the current  $\LaTeX$  system.

Only the last of these three interfaces is covered by this paper, which describes a system aimed at providing a suitable basis for coding large scale projects in  $\TeX$  (but this should not preclude its use for smaller projects). Its main distinguishing features are summarised here.

- A consistent naming scheme for all commands, including  $\TeX$  primitives.
- The classification of commands as  $\LaTeX$  functions or  $\LaTeX$  parameters, and also their division into modules according to their functionality.
- A simple mechanism for controlling argument expansion.
- Provision of a set of core  $\LaTeX$  functions that is sufficient for handling programming constructs such as queues, sets, stacks, property lists.
- A  $\TeX$  programming environment in which, for example, all white space is ignored.

### 3. The naming scheme

The naming conventions for this programming language distinguish between *functions* and *parameters*. Functions can have arguments and they are executed. Parameters can be assigned values and they are used in arguments to functions; they are not directly executed but are manipulated by mutator and accessor functions. Functions and parameters with a related functionality (for example accessing counters, or manipulating token-lists, etc.) are collected together into a *module*.

Note that all these terms are only  $\LaTeX$  terminology and are not, for example, intended to indicate that the commands have these properties when considered in the context of basic  $\TeX$  or in any more general programming context.

### 3.1. Examples

Before giving the details of the naming scheme, here are a few typical examples to indicate the flavour of the scheme; first some parameter names.

`\l_tmpa_box` is a local parameter (hence the `l_` prefix) corresponding to a box register.

`\g_tmpa_int` is a global parameter (hence the `g_` prefix) corresponding to an integer register (i.e., a  $\TeX$  count register).

`\c_empty_toks` is the constant (`c_`) token register parameter that is for ever empty.

Now here is an example of a typical function name.

`\seq_push:Nn` is the function which puts the token list specified by its second argument onto the stack specified by its first argument. The different natures of the two arguments are indicated by the `:Nn` suffix. The first argument must be a single token which ‘names’ the stack parameter: such single-token arguments are denoted `N`. The second argument is a normal  $\TeX$  ‘undelimited argument’, which may either be a single token or a balanced, brace-delimited token list (which we shall here call a *braced token list*): the `n` denotes such a ‘normal’ argument form.

`\seq_push:cn` would be similar to the above, but in this case the `c` means that the stack-name is specified in the first argument by a token list that expands, using `\csname...`, to a control sequence that is the *name* of the stack parameter.

The names of these two functions also indicate that they are in the module called `seq`.

### 3.2. Formal syntax of the conventions

We shall now look in more detail at the syntax of these names. The syntax of parameter names is as follows:

$$\backslash\langle access \rangle\_ \langle module \rangle\_ \langle description \rangle\_ \langle type \rangle$$

The syntax of function names is as follows:

$$\backslash\langle module \rangle\_ \langle description \rangle : \langle arg-spec \rangle$$

### 3.3. Modules and descriptions

The syntax of all names contains

$\langle module \rangle$  and  $\langle description \rangle$ :

these both give information about the command.

A *module* is a collection of closely related functions and parameters. Typical module names include `int` for integer parameters and related functions, `—seq—` for sequences and `box` for boxes.

Packages providing new programming functionality will add new modules as needed; the programmer can choose any unused name, consisting of letters only, for a module.

The *description* gives more detailed information about the function or parameter, and provides a unique name for it. It should consist of letters and, possibly, `_` characters.

### 3.4. Parameters: access and type

The  $\langle access \rangle$  part of the name describes how the parameter can be accessed. Parameters are primarily classified as local, global or constant (there are further, more technical, classes). This *access* type appears as a code at the beginning of the name; the codes used include:

`c` constants (global parameters whose value should not be changed);

`g` parameters whose value should only be set globally;

`l` parameters whose value should only be set locally.

The  $\langle type \rangle$  will normally (except when introducing a new data-type) be in the list of available *data-types*; these include the primitive  $\text{\TeX}$  data-types, such as the various registers, but to these will be added data-types built within the  $\text{\LaTeX}$  programming system.

Here are some typical data-type names:

`int` integer-valued count register;

`toks` token register;

`box` box register;

---

**fint** ‘Fake-integer’: (or fake-counter) a data type created to avoid problems with the limited number of available count registers in (standard)  $\text{\TeX}$ ;

**seq** ‘sequence’: a data-type used to implement lists (with access at both ends) and stacks;

**plist** property list

When the  $\langle type \rangle$  and  $\langle module \rangle$  are identical (as often happens in the more basic modules) the  $\langle module \rangle$  part is often omitted for aesthetic reasons.

### 3.5. Functions: argument specifications

Function names end with an  $\langle arg-spec \rangle$  after a colon. This gives an indication of the types of argument that a function takes, and provides a convenient method of naming similar functions that differ only in their argument forms (see the next section for examples).

The  $\langle arg-spec \rangle$  consists of a (possibly empty) list of characters, each denoting one argument of the function. It is important to understand that ‘argument’ here refers to the effective argument of the  $\text{\LaTeX}$  function, not to an argument at the  $\text{\TeX}$ -level. Indeed, the top level  $\text{\TeX}$  macro that has this name typically has no arguments. This is an extension of the existing  $\text{\LaTeX}$  convention where one says that  $\backslash section$  has an optional argument and a mandatory argument, whereas the  $\text{\TeX}$  macro  $\backslash section$  actually has zero parameters at the  $\text{\TeX}$  level, it merely calls an internal  $\text{\LaTeX}$  command which in turn calls others that look ahead for star forms and optional arguments.

The list of possible argument specifiers includes the following.

**n** Unexpanded token or braced token list.

This is a standard  $\text{\TeX}$  undelimited macro argument.

**o** One-level-expanded token or braced token list.

This means that the argument is expanded one level, as is done by  $\backslash expandafter$ , and the expansion is passed to the function as a braced token list. Note that if the original argument is a braced token list then only the first token in that list is expanded.

**x** Fully-expanded token or braced token list.

This means that the argument is expanded as in the replacement text of an  $\backslash undef$ , and the expansion is passed to the function as a braced token list.

- 
- c** Character string used as a command name.  
The argument (a token or braced token list) must, when fully expanded, produce a sequence of characters which is then used to construct a command name (via `\csname`, `\endcsname`). This command name is the single token that is passed to the function as the argument.
  - N** Single token (unlike **n**, the argument must *not* be surrounded by braces).  
A typical example of a command taking an **N** argument is `\def`, in which the command being defined must be unbraced.
  - O** One-level-expanded single token (unbraced).  
As for **o**, the one-level expansion is passed (as a braced token list) to the function.
  - X** Fully-expanded single token (unbraced).  
As for **x**, the full expansion is passed (as a braced token list) to the function.
  - C** Character string used as a command name then one-level expanded.  
The form of the argument is exactly as for **c**, but the resulting token is then expanded one level (as for **O**), and the expansion is passed to the function as a braced token list.
  - p** Primitive  $\TeX$  parameter specification.  
This can be something simple like `#1#2#3`, but may use arbitrary delimited argument syntax such as: `#1,#2\q_stop#3`.
  - T,F** These are special cases of **n** arguments, used for the true and false code in conditional commands.

There are two other specifiers with more general meanings:

- D** This means: **Do not use**. This special case is used for  $\TeX$  primitives and other commands that are provided for use only while bootstrapping the  $\LaTeX$  kernel. If the  $\TeX$  primitive needs to be used in other contexts it will be given an alternative, more appropriate, name with a useful argument specification. The argument syntax of these is often weird, in the sense described next.
- w** This means that the argument syntax is ‘weird’ in that it does not follow any standard rule. It is used for functions with arguments that take non standard forms: examples are  $\TeX$ -level delimited arguments and the boolean tests needed after certain primitive `\if...` commands.

## 4. Expansion control

### 4.1. Simpler means better

Anyone who programs in  $\text{\TeX}$  is frustratingly familiar with the problem of arranging that arguments to functions are suitably expanded before the function is called. To illustrate how expansion control can bring instant relief to this problem we shall consider two examples copied from `latex.ltx`.

```

\global
\expandafter
  \expandafter
\expandafter
  \let
\expandafter
  \reserved@a
\csname \curr@fontshape \endcsname

```

This first piece of code is in essence simply a global `\let`. However, the token to be defined is obtained by expanding `\reserved@a` one level; and, worse, the token to which it is to be let is obtained by fully expanding `\curr@fontshape` and then using the characters produced by that expansion to construct a command name. The result is a mess of interwoven `\expandafter` and `\csname` beloved of all  $\text{\TeX}$  programmers, and the code is essentially unreadable.

Using the conventions and functionality outlined here, the task would be achieved with code such as this:

```

\glet:0c \g_reserved_a_tlp
         \l_current_font_shape_tlp

```

The command `\glet:0c` is a global `\let` that expands its first argument once, and generates a command name out of its second argument, before making the definition. This produces code that is far more readable and more likely to be correct first time.

Here is the second example.

```

\expandafter
  \in@
\csname sym#3%
  \expandafter

```

---

```

\endcsname
\expandafter
{%
\group@list}%

```

This piece of code is part of the definition of another function. It first produces two things: a token list, by expanding `\group@list` once; and a token whose name comes from ‘`sym#3`’. Then the function `\in@` is called and this tests if its first argument occurs in the token list of its second argument.

Again we can improve enormously on the code. First we shall rename the function `\in@` according to our conventions. A function such as this but taking two normal ‘`n`’ arguments might reasonably be named `\seq_test_in:nn`; thus the variant function we need will be defined with the appropriate argument types and its name will be `\seq_test_in:c0`. Now this code fragment will be simply:

```
\seq_test_in:c0 {sym#3} \l_group_seq
```

Note that, in addition to the lack of —, the space after the `}` will be silently ignored since all white space is ignored in this programming environment.

## 4.2. New functions from old

For many common functions the  $\LaTeX$ 3 kernel will provide variants with a range of argument forms, and similarly it is expected that extension packages providing new functions will make them available in the all the commonly needed forms.

However, there will be occasions where it is necessary to construct a new such variant form; therefore the expansion module provides a straightforward mechanism for the creation of functions with any required argument type, starting from a function that takes ‘normal’  $\TeX$  undelimited arguments.

To illustrate this let us suppose you have a ‘base function’ `\demo_cmd:nnn` that takes three normal arguments, and that you need to construct the variant `\demo_cmd:cnx`, for which the first argument is used to construct the *name* of a command, whilst the third argument must be fully expanded before being passed to `\demo_cmd:nnn`. To produce the variant form from the base form, simply use this:

```
\exp_def_form:nnn {demo_cmd} {nnn} {cnx}
```

This defines the variant form so that you can then write, for example:

```
\demo_cmd:cnx {abc} {pq} {\rst \xyz }
```

rather than ... well, something like this!

```
\def \tempa {{pq}}%
\edef \tempb {\rst \xyz}%
\expandafter
  \demo@cmd
\csname abc%
  \expandafter
  \expandafter
  \expandafter
  \endcsname
\expandafter
  \tempa
\expandafter
  {%
  \tempb
  }%
```

As a further example, you may wish to declare a function `\demo_cmd_b:xcxcx`, as a variant of an existing function `\demo_cmd_b:nnnnn`, that fully expands arguments 1, 3 and 5, and produces commands to pass as arguments 2 and 4 using `\csname`. The definition you need is simply

```
\exp_def_form:nnn
  {demo_cmd_b} {nnnnn} {xcxcx}
```

This extension mechanism is written so that if the same new form of some existing command is implemented by two extension packages then the two definitions will be identical and thus no conflict will occur.

## 5. Parameter assignments and accessor functions

### 5.1. Checking assignments

One of the advantages of having a consistent scheme is that the system can provide more extensive error-checking and debugging facilities. For example,

an accessor function that makes a *global* assignment of a value to a parameter can check that it is not passed the name of a *local* parameter as that argument: it does this by checking that the name starts with `\g_`.

Such checking is probably too slow for normal use, but the code can have hooks built in that allow a format to be made in which all functions perform this kind of check.

A typical section of the source<sup>1</sup> for such code might look like this (recall that all white space is ignored):

```
%<!*check>
\let_new:NN
  \toks_gset:Nn \tex_global:D
%</!check>
%<*check>
\def_new:Npn
  \toks_gset:Nn #1
  {
    \chk_global:N #1
    \tex_global:D #1
  }
%</check>
```

In the above code the function `\toks_gset:Nn` takes a single token (N) specifying a token register, and globally sets it to the value passed in the second argument.

A typical use of it would be:

```
\toks_gset \g_xxx_toks {<some value>}
```

In the normal definition, `\toks_gset` can be simply `\let` to `\global` because the primitive  $\text{\TeX}$  token register does not require any explicit assignment function: this is done by the `%<!*check>` code above.

The alternative definition first checks that the argument passed as `#1` is the name of a global parameter and raises an error if it is not. It does this by taking apart the command name passed as `#1` and checking that it starts `\g_`.

---

<sup>1</sup> This code uses the `docstrip` system described in [2], Section 14.3.

## 5.2. Consistency

The primitive  $\text{T}_{\text{E}}\text{X}$  syntax for register assignments has a very minimal syntax and, apart from box functions, there are no explicit functions for assigning values to these registers.

This makes it impossible to implement alternative data-types with a syntax that is both consistent and at all similar to the syntax for the primitives; moreover, it encourages a coding style that is very error prone.

As in the `\toks_gset:Nn` example given above, all  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  data-types are provided with explicit functions for assignment and for use, even when these have essentially empty definitions. This allows for better error-checking as described above; it also allows the construction of further data-types with a similar interface, even when the implementation of the associated functions is very complex.

For example, the ‘fake-counter’ (`fint`) data-type mentioned above will appear at the  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  programming level to be exactly like the data-type based on primitive count registers; however, internally it makes no use of count registers. Typical functions in this module are illustrated here.

```
\fint_new:N \l_tmpa_fint
```

This declares the local parameter `\l_example_fint` as a fake-counter.

```
\fint_add:Nn \l_example_fint \c_thirty_two
```

This increments the value of this fake-counter by 32.

## 6. The experimental distribution

The initial implementations of a  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  programming language using this kind of syntax remain unreleased (and not completely functional); they partly pre-date  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\varepsilon}$ ! The planned distribution will provide a subset of the functionality of those implementations, in the form of packages to be used on top of  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\varepsilon}$ .

The intention is to allow experienced  $\text{T}_{\text{E}}\text{X}$  programmers to experiment with the system and to comment on the interface. This means that *the interface will change*. No part of this system, including the name of anything, should be relied upon as being available in a later release. Please do *experiment* with

---

these packages, but do *not* use them for code that you expect to keep unchanged over a long period.

In view of the intended experimental use for this distribution we shall, in the first instance, produce only a few modules for use with  $\LaTeX$ 2 $\epsilon$ . These will set up the conventions and the basic functionality of, for example, the expansion mechanism; they will also implement some of the basic programming constructs, such as token-lists and sequences. They are intended only to give a flavour of the code: the full  $\LaTeX$ 3 kernel will provide a very rich set of programming constructs so that packages can efficiently share code, in contrast with the situation in the current  $\LaTeX$  where every large package must implement its own version of queues, stacks, etc., as necessary.

In the first release of this experimental system at least the following modules will be distributed.

**l3names** This sets up the basic naming scheme and renames all the  $\TeX$  primitives. If it is loaded with the option `[removeoldnames]` then the old primitive names such as `\box` become *undefined* and are thus available for user definition. Caution: use of this option will certainly break existing  $\TeX$  code!

**l3basics** This contains the basic definition modules used by the other packages.

**l3tlp** This implements a basic data-type, called a *token-list pointer*, used for storing named token lists: these are essentially  $\TeX$  macros with no arguments.

**l3expan** This is the argument expansion module discussed above.

**l3quark** A ‘quark’ is a command that is defined to expand to itself! Therefore they must never be expanded as this will generate infinite recursion; they do however have many uses, e.g., as special markers and delimiters within code.

**l3seq** This implements data-types such as queues and stacks.

**l3prop** This implements the data-type for ‘property lists’ that are used, in particular, for storing key/value pairs.

This distribution will also contain the  $\LaTeX$  source for the latest version of this document, a docstrip install file and two small test files.

In later releases we plan to add further modules and a full-fledged example of the use of the new language: a proto-type implementation for the ideas described in the article ‘Language Information in Structured Documents: A Model for Mark-up and Rendering’ [5].

## Bibliography

- [1] Donald E Knuth *The T<sub>E</sub>Xbook*. Addison-Wesley, Reading, Massachusetts, 1984.
- [2] Goossens, Mittelbach and Samarin. *The L<sup>A</sup>T<sub>E</sub>X Companion*. Addison-Wesley, Reading, Massachusetts, 1994.
- [3] Leslie Lamport. *L<sup>A</sup>T<sub>E</sub>X: A Document Preparation System*. Addison-Wesley, Reading, Massachusetts, second edition, 1994.
- [4] Frank Mittelbach and Chris Rowley. The L<sup>A</sup>T<sub>E</sub>X3 Project. *TUGboat*, ????? ??? 1998.
- [5] Frank Mittelbach and Chris Rowley. Language Information in Structured Documents: A Model for Mark-up and Rendering. *TUGboat*, ????? ??? 1998.