Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Αναγνώριση και εντοπισμός δραστηριότητας σε βίντεο

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΕΥΣΤΑΙΟΣ ΓΑΛΑΝΑΚΗΣ

Επιβλέπων : Νικόλαος Σ. Παπασπύρου
Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, Δεκέμβριος 2019

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Αναγνώριση και εντοπισμός δραστηριότητας σε βίντεο

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΕΥΣΤΑΙΟΣ ΓΑΛΑΝΑΚΗΣ

Επιβλέπων : Νικόλαος Σ. Παπασπύρου
Αν. Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την -1η Δεκεμβρίου 2019.

Νικόλαος Σ. Παπασπύρου Αν. Καθηγητής Ε.Μ.Π.	Πέτρος Χ. Παπαδόπουλος Επίκ. Καθηγητής Ε.Μ.Π.	Γεώργιος Χ. Νικολάου Καθηγητής Ε.Κ.Π.Α.

Αθήνα, Δεκέμβριος 2019

Ευστάιος Γαλανάκης

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Abstract

The purpose of this thesis is a design a system which gets as input a video and tries to locate and classify any actions appear in it.

As Machine Learning has contribute rapid improvements in the performance of systems who recognise v Στη σημερινή εποχή, η ανάγκη για αξιόπιστο και πιστοποιημένα ασφαλή κώδικα γίνεται διαρκώς ευρύτερα αντιληπτή. Τόσο κατά το παρελθόν όσο και πρόσφατα έχουν γίνει γνωστά προβλήματα ασφάλειας και συμβατότητας προγραμμάτων που είχαν ως αποτέλεσμα προβλήματα στην λειτουργία μεγάλων συστημάτων και συνεπώς οικονομικές επιπτώσεις στους οργανισμούς που τα χρησιμοποιούσαν. Τα προβλήματα αυτά οφείλονται σε μεγάλο βαθμό στην έλλειψη δυνατότητας προδιαγραφής και απόδειξης της ορθότητας των προγραμμάτων που χαρακτηρίζει τις σύγχρονες γλώσσες προγραμματισμού. Για το σκοπό αυτό, έχουν προταθεί συστήματα πιστοποιημένων εκτελέσιμων, στα οποία έχουμε τη δυνατότητα να προδιαγράφουμε την ορθότητα των προγραμμάτων, και να παρέχουμε μία τυπική απόδειξη αυτής, η οποία μπορεί να ελεγχθεί μηχανιστικά πριν το χρόνο εκτέλεσης.

Τα συστήματα που έχουν προταθεί είναι ενδιάμεσου επιπέδου οπότε η διαδικασία προγραμματισμού σε αυτά είναι ιδιαίτερα πολύπλοκη. Οι γλώσσες υψηλού επιπέδου που συνοδεύουν αυτά τα συστήματα, ενώ είναι ιδιαίτερα εκφραστικές, παραμένουν δύσκολες στον προγραμματισμό. Μία απλούστερη γλώσσα υψηλού επιπέδου, όπως αυτή που προτείνουμε σε αυτή την εργασία, θα επέτρεπε ευρύτερη εξάπλωση του συγκεκριμένου ιδιώματος προγραμματισμού.

Στη γλώσσα που προτείνουμε, ο προγραμματιστής προδιαγράφει τη μερική ορθότητα του προγράμματος, δίνοντας προσυνθήκες και μετασυνθήκες για τις παραμέτρους και τα αποτελέσματα των συναρτήσεων που ορίζει. Επίσης δίνει ένα σύνολο θεωρημάτων βάσει του οποίου κατασκευάζονται αποδείξεις της ορθής υλοποίησης και χρήσης των συναρτήσεων αυτών. Ως μέρος της εργασίας, έχουμε υλοποιήσει σε γλώσσα OCaml ένα μεταφραστή αυτής της γλώσσας στο σύστημα πιστοποιημένων εκτελέσιμων NFLINT.

Επιτύχαμε να διατηρήσουμε τη γλώσσα κοντά στο ύφος των ευρέως διαδεδομένων συναρτησιακών γλωσσών, καθώς και να διαχωρίσουμε τη φάση προγραμματισμού από τη φάση απόδειξης της ορθότητας των προγραμμάτων. Έτσι ένας μέσος προγραμματιστής μπορεί εύκολα να προγραμματίζει στη γλώσσα που προτείνουμε με τον τρόπο που ήδη γνωρίζει, και ένας γνώστης μαθηματικής λογικής να αποδεικνύει σε επόμενη φάση την μερική ορθότητα των προγραμμάτων. Ως απόδειξη της πρακτικότητας της προσέγγισης αυτής, παραθέτουμε ένα σύνολο παραδειγμάτων στη γλώσσα με απόδειξη μερικής ορθότητας.

Key words

Γλώσσες προγραμματισμού, Προγραμματισμός με αποδείξεις, Ασφαλείς γλώσσες προγραμματισμού, Πιστοποιημένος κώδικας.

Abstract

The purpose of this diploma dissertation is on one hand the design of a simple high-level language that supports programming with proofs, and on the other hand the implementation of a compiler for this language. This compiler will produce code for an intermediate-level language suitable for creating certified binaries.

The need for reliable and certifiably secure code is even more pressing today than it was in the past. In many cases, security and software compatibility issues put in danger the operation of large systems, with substantial financial consequences. The lack of a formal way of specifying and proving the correctness of programs that characterizes current programming languages is one of the main reasons why these issues exist. In order to address this problem, a number of frameworks with support for certified binaries have recently been proposed. These frameworks offer the possibility of specifying and providing a formal proof of the correctness of programs. Such a proof can easily be checked for validity before running the program.

The frameworks that have been proposed are intermediate-level in nature, thus the process of programming in these is rather cumbersome. The high-level languages that accompany some of these frameworks, while very expressive, are hard to use. A simpler high-level language, like the one proposed in this dissertation, would enable further use of this programming idiom.

In the language we propose, the programmer specifies the partial correctness of a program by annotating function definitions with pre- and post-conditions that must hold for their parameters and results. The programmer also provides a set of theorems, based on which proofs of the proper implementation and use of the functions are constructed. An implementation in OCaml of a compiler from this language to the NFLINT certified binaries framework was also completed as part of this dissertation.

We managed to keep the language close to the feel of the current widespread functional languages, and also to fully separate the programming stage from the correctness-proving stage. Thus an average programmer can program in a familiar way in our language, and later an expert on formal logic can prove the semi-correctness of a program. As evidence of the practicality of our design, we provide a number of examples in our language with full semi-correctness proofs.

Key words

Programming languages, Programming with proofs, Secure programming languages, Certified code.

Ευχαριστίες

Ευχαριστώ θερμά τον επιβλέποντα καθηγητή αυτής της διατριβής, κ. Γιάννη Παπαδάκη, για τη συνεχή καθοδήγηση και εμπιστοσύνη του. Ευχαριστώ επίσης τα μέλη της συμβουλευτικής επιτροπής, κ.κ. Νίκο Παπαδόπουλο και Γιώργο Νικολάου για την πρόθυμη και πάντα αποτελεσματική βοήθειά τους, τις πολύτιμες συμβουλές και τις χρήσιμες συζητήσεις που είχαμε. Θέλω να ευχαριστήσω ακόμα τον συμφοιτητή και φίλο Πέτρο Πετρόπουλο, ο οποίος με βοήθησε σε διάφορα στάδια αυτής της εργασίας. Θα ήθελα τέλος να ευχαριστήσω την οικογένειά μου και κυρίως τους γονείς μου, οι οποίοι με υποστήριξαν και έκαναν δυνατή την απερίσπαστη ενασχόλησή μου τόσο με την εκπόνηση της διπλωματικής μου, όσο και συνολικά με τις σπουδές μου.

Ευστάιος Γαλανάκης,

Αθήνα, -1η Δεκεμβρίου 2019

Η εργασία αυτή είναι επίσης διαθέσιμη ως Τεχνική Αναφορά CSD-SW-TR-42-14, Εθνικό Μετσόβιο Πολυτεχνείο, Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών, Εργαστήριο Τεχνολογίας Λογισμικού, Δεκέμβριος 2019.

URL: `http://www.softlab.ntua.gr/techrep/`
FTP: `ftp://ftp.softlab.ntua.gr/pub/techrep/`

Περιεχόμενα

Κατάλογος πινάκων

Κατάλογος σχημάτων

Κεφάλαιο 1

Εισαγωγή

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

1.1 Η γλώσσα προγραμματισμού C

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα

μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

1.2 Σημασιολογία γλωσσών προγραμματισμού

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

1.3 Θεωρία πεδίων

Κεφάλαιο 2

Εισαγωγή

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

2.1 Η γλώσσα προγραμματισμού C

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα

μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

2.2 Σημασιολογία γλωσσών προγραμματισμού

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

2.3 Θεωρία πεδίων

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

Μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα, μπλα μπλα, μπλα μπλα μπλα, μπλα, μπλα μπλα μπλα.

Βιβλιογραφία

[Appe00] Andrew W. Appel and Amy P. Felty, "A Semantic Model of Types and Machine Instructions for Proof-Carrying Code", in *Proceedings of the 27th Annual Symposium on Principles of Programming Languages (POPL 2000)*, pp. 243–253, ACM Press, 2000.

[Appe01] A. W. Appel, "Foundational Proof-Carrying Code", in *Proceedings of the 16th Annual IEEE Symposium on Logic in Computer Science*, pp. 247–258, June 2001.

[Bohm85] C. Böhm and A. Bernarducci, "Automatic Synthesis of Typed λ-Programs on Term Algebras", *Theoretical Computer Science*, vol. 39, no. 2–3, pp. 135–154, August 1985.

[Chur32] A. Church, "A Set of Postulates for the Foundations of Logic", *Annals of Mathematics*, vol. 33, no. 1, pp. 346–366, 1932.

[Chur33] A. Church, "A Set of Postulates for the Foundations of Logic", *Annals of Mathematics*, vol. 34, no. 2, pp. 839–864, 1933.

[Gira72] J.-Y. Girard, *Interprétation Fonctionelle et Élimination des Coupures Dans l'Arithmétique d'Ordre Supérieur*, Ph.D. thesis, Université Paris 7, 1972.

[Gira89] J.-Y. Girard, Y. Lafont and P. Taylor, "Proofs and Types", *Tracks in Theoretical Computer Science*, 1989.

[Harp95] Robert Harper and Greg Morrisett, "Compiling Polymorphism Using Intensional Type Analysis", in *Proceedings of the 22nd Annual Symposium on Principles of Programming Languages (POPL 1995)*, pp. 130–141, ACM Press, 1995.

[Morr98] Greg Morrisett, David Walker, Karl Crary and Neal Glew, "From System F to Typed Assembly Language", in *Proceedings of the 25th Annual Symposium on Principles of Programming Languages (POPL 1998)*, pp. 85–97, ACM Press, January 1998.

[Necu96] G. Necula and P. Lee, "Safe Kernel Extensions without Run-Time Checking", in *Proceedings of the 2nd USENIX Symposium on Operating System Design and Implementation*, pp. 229–243, USENIX Association, 1996.

[Necu97] G. Necula, "Proof-Carrying Code", in *Proceedings of the 24th Annual Symposium on Principles of Programming Languages (POPL 1997)*, pp. 106–119, New York, January 1997, ACM Press.

[Necu98] G. Necula, *Compiling with Proofs*, Ph.D. thesis, Carnegie Mellon University, September 1998.

[Paul89] C. Paulin-Mohring, *Extraction de Programmes Dans le Calcul des Constructions*, Ph.D. thesis, Université Paris 7, January 1989.

[Paul93] Christine Paulin-Mohring, "Inductive Definitions in the System Coq: Rules and Properties", in M. Bezem and J. F. Groote, editors, *Proceedings of the 1st Int. Conf. on Typed Lambda Calculi and Applications, TLCA'93, Utrecht, The Netherlands, 16–18 March 1993*, vol. 664, pp. 328–345, Springer-Verlag, Berlin, 1993.

[Pfen90] F. Pfenning and C. Paulin-Mohring, "Inductively defined types in the Calculus of Constructions", in *Proceedings of Mathematical Foundations of Programming Semantics*, vol. 442 of *Lecture Notes in Computer Science*, Berlin, 1990, Springer-Verlag. technical report CMU-CS-89-209.

[Reyn74] John. C. Reynolds, "Towards a Theory of Type Systems", in Ehring et al., editor, *Lecture Notes in Computer Science*, vol. 19, pp. 408–425, Springer-Verlag, 1974.

[Sell94] M. P. A. Sellink, "Verifying process algebra proofs in type theory", in D. J. Andrews, J. F. Groote and C. A. Middelburg, editors, *Proceedings of the International Workshop on Semantics of Specipication Languages (SOSL 1993)*, Springer, 1994.

[Shao02] Z. Shao, B. Saha, V. Trifonov and N. Papaspyrou, "A Type System for Certified Binaries", in *Proceedings of the 29th Annual Symposium on Principles of Programming Languages (POPL 2002)*, pp. 217–232, Portland, OR, USA, January 2002.

Παράρτημα Α

Ευρετήριο συμβολισμών

$A \to B$: συνάρτηση από το πεδίο A στο πεδίο B.

Παράρτημα Β

Ευρετήριο γλωσσών

Haskell : η γλώσσα της ζωής μου.

Παράρτημα C

Ευρετήριο αριθμών

42 : life, the universe and everything.