# Κεφάλαιο 3
# Σημασιολογία των Γλωσσών Προγραμματισμού

Π. Ροντογιάννης

Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών
Τμήμα Πληροφορικής και Τηλεπικοινωνιών

# Εισαγωγή

Γνώση γλώσσας από τη σκοπιά

- Του **συντακτικού** (syntax)
  - Περιγραφή με γραμματικές χωρίς συμφραζόμενα ή με επεκτάσεις τους
- Της **σημασιολογίας** (semantics)
  - Περιγραφή με τυπικές (μαθηματικές) μεθόδους

# Παράδειγμα 3.1

- Περίπτωση βρόγχου while, με μορφή
  - `while <λογική συνθήκη> do <εντολές>`
- Άτυπη περιγραφή σημασιολογίας
  - «Αρχικά γίνεται έλεγχος της λογικής συνθήκης. Αν το αποτέλεσμα είναι αληθές, τότε γίνεται είσοδος στο βρόγχο και εκτέλεση των εντολών που βρίσκονται μέσα σε αυτόν, μία φορά. Έλεγχος και πάλι της συνθήκης. Για ψευδή συνθήκη, ο βρόγχος παρακάμπτεται και ο έλεγχος μεταφέρεται στην πρώτη εντολή που ακολουθεί τη δομή του βρόγχου».

# Παράδειγμα 3.2

- **Ερώτηση**: Είναι στην ALGOL 60 ισοδύναμες οι εκφράσεις `x+f(x)` και `f(x)+x` ;
- **Απάντηση**: Εξαρτάται από το αν η `f` αλλάζει την τιμή του `x`. Εάν ναι, τότε η τιμή των δύο παραπάνω εκφράσεων είναι διαφορετική ανάλογα με τη σειρά που η υλοποίηση της γλώσσας υπολογίζει τις τιμές των `x` και `f(x)`.

# Μαθηματική Θεωρία

Αναγκαία για την εξασφάλιση

- Ορθής **περιγραφής** και **υλοποίησης** γλωσσών προγραμματισμού
- Συστηματικής **ανάπτυξης** και **απόδειξης ορθότητας** προγραμμάτων
- **Ανάλυσης** και **αξιολόγησης** γνωστών γλωσσών προγραμματισμού
- **Σχεδιασμού** νέων, απλούστερων και πιο ισχυρών γλωσσών προγραμματισμού

# Μέθοδοι Σημασιολογίας

- Μηχανική (operational semantics)
- Αξιωματική (axiomatic semantics)
- Δηλωτική (denotational semantics)

# Μηχανική Σημασιολογία - 1

- «Αφηρημένο μοντέλο μηχανής», που αποτελείται
  - από την κατάσταση (state)
  - από σύνολο βασικών εντολών
- Η σημασιολογία μιας γλώσσας προγραμματισμού ορίζεται με βάση το μοντέλο αυτό.

# Μηχανική Σημασιολογία - 2

- Βασική ιδέα
  - Απλό μοντέλο
  - Δεν αφήνει περιθώρια για παρανοήσεις
- Η σημασιολογία μιας γλώσσας καθορίζεται ως μετάφραση βασικών εντολών της στη γλώσσα του αφηρημένου μοντέλου
- Γλώσσες PL/1 (μέθοδος VDM), ALGOL 68, κλπ

# Αξιωματική Σημασιολογία - 1

Ένα αξίωμα

- συνδέεται με κάθε είδος εντολής της γλώσσας προγραμματισμού
- προσδιορίζει τι θεωρείται αληθές μετά την εκτέλεση της εντολής, σε σχέση με όσα ίσχυαν πριν από την εκτέλεση της

# Αξιωματική Σημασιολογία - 2

## Παράδειγμα

*Βρόχος*: `while B do C`

B: *έκφραση*

C: *εντολή*

*Όταν ο βρόγχος ολοκληρωθεί (εάν ποτέ ολοκληρωθεί) η τιμή της έκφρασης* B *θα είναι ψευδής.*

# Αξιωματική Σημασιολογία - 3

Προτέρημα

- Τα αξιώματα, που εισάγονται, εκφράζονται στην ίδια γλώσσα προγραμματισμού (με ελάχιστες αλλαγές στο συντακτικό της).
- Εάν η απόδειξη της ορθότητας ενός προγράμματος γίνει μέλημα του ίδιου του προγραμματιστή, τότε η απόδειξη και ο προγραμματισμός γίνονται στην ίδια γλώσσα.

# Ενδεικτική Σημασιολογία

«Συναρτήσεις υπολογισμού»

- Αντιστοιχίζουν τα συντακτικά στοιχεία της γλώσσας σε γνωστά μαθηματικά αντικείμενα
  - Αριθμούς, τιμές αλήθειας, συναρτήσεις κλπ
- Ορίζονται αναδρομικά
  - Η τιμή, που λαμβάνει μία έκφραση της γλώσσας, καθορίζεται με βάση τις τιμές των υπο-εκφράσεων, που την αποτελούν

# Παρατηρήσεις για τις Μεθόδους

- Αποτελούν αποδεκτές τεχνικές για τον καθορισμό της σημασιολογίας μιας γλώσσας προγραμματισμού
- Διαφέρουν ως προς τη φιλοσοφία τους
  - Μηχανική $\rightarrow$ Υλοποίηση γλώσσας
  - Αξιωματική $\rightarrow$ Αποδεκτή από προγραμματιστές
  - Ενδεικτική $\rightarrow$ Χρησιμοποίηση από σχεδιαστές γλώσσας

# Παράδειγμα - Δυαδικές Ακολουθίες

- Συντακτικό γλώσσας δυαδικών ακολουθιών
  - Περιγραφή με context-free γραμματική

```
S ::= 0
    | 1
    | S0
    | S1
```

# Παράδειγμα - Δυαδικές Ακολουθίες

*Seq*

- Είναι το σύνολο των ακολουθιών, που παράγονται με βάση τον ορισμό
- Αποτελείται από πεπερασμένες ακολουθίες των ψηφίων 0 και 1
- Δεν γνωρίζουμε (ακόμη) το νόημα τους

# Παράδειγμα - Δυαδικές Ακολουθίες

Ενδεικτική σημασιολογία

- Αντιστοίχιση σε κάθε δυαδική ακολουθία ενός μοναδικού φυσικού αριθμού
- Ορισμός συνάρτησης, που μετατρέπει κάθε ακολουθία σε φυσικό αριθμό.
- Συμβολισμός συνάρτησης με $[[\cdot]]$

# Παράδειγμα - Δυαδικές Ακολουθίες

Στη συγκεκριμένη περίπτωση, η $[[\cdot]]$ μετατρέπει ακολουθίες δυαδικών ψηφίων σε φυσικούς αριθμούς ($[[\cdot]] : \mathit{Seq} \to \mathit{Nat}$)
Ορίζεται ως:

$$
\begin{array}{lcl}
[[0]] & = & 0 \\
[[1]] & = & 1 \\
[[S0]] & = & 2 \times [[S]] \\
[[S1]] & = & 2 \times [[S]] + 1
\end{array}
$$

# Παράδειγμα 3.3

Ακολουθία 1100

$$
\begin{aligned}
[[1100]] &= 2 \times [[110]] \\
&= 2 \times (2 \times [[11]]) \\
&= 2 \times (2 \times (2 \times [[1]] + 1) \\
&= 2 \times (2 \times (2 \times 1 + 1)) \\
&= 12
\end{aligned}
$$

# Σχόλια

- Υπάρχει μία εξίσωση για κάθε κανόνα του συντακτικού
- Δεν είναι προφανές ότι οι εξισώσεις ορίζουν μία μαθηματική συνάρτηση
  - Το σύμβολο $[[\cdot]]$ εμφανίζεται και στη δεξιά και στην αριστερή πλευρά των εξισώσεων

# Εξισώσεις $[[0]] = 0$

- Η εξίσωση δεν είναι κυκλική
- Το 0 στα αριστερά είναι σύμβολο της γλώσσας - αντικείμενο (object language)
  - Την μελετάμε και επιθυμούμε να της αποδώσουμε νόημα
  - Σύνολο δυαδικών ακολουθιών ή μία οποιαδήποτε γλώσσα προγραμματισμού (π.χ. C ή Pascal)
- Το 0 στα δεξιά είναι ο φυσικός αριθμός, που αποτελεί το νόημα της ακολουθίας 0.

# Ενδεικτική Σημασιολογία

- Σκοπός η μετατροπή συντακτικών στοιχείων μιας γλώσσας προγραμματισμού σε γνωστά μαθηματικά αντικείμενα
- Τα μαθηματικά αποτελούν τη μεταγλώσσα στην οποία μετατρέπονται τα προγράμματα της γλώσσας αντικείμενο

# Συνθεσιμότητα

- Το νόημα κάθε σύνθετης έκφρασης δίνεται ως συνάρτηση των νοημάτων των υποεκφράσεων που την αποτελούν.

# Παράδειγμα 3.4

- Οι εξισώσεις στον ορισμό της σημασιολογίας δυαδικών ακολουθιών, υπακούουν στον κανόνα της συνθεσιμότητας, γιατί ανάγουν τον υπολογισμό της σημασιολογίας των εκφράσεων S0 και S1 στην σημασιολογία της απλούστερης έκφρασης S.

# Παράδειγμα 3.4

`[[while B do C]]=[[if B then (C; while B do C)]]`

- Ο παραπάνω ορισμός δεν υπακούει στον κανόνα της συνθεσιμότητας.
- Η έκφραση εντός του συμβόλου [[.]] στη δεξιά πλευρά δεν αποτελεί υπο-έκφραση αυτής στην αριστερή.

# Παράδειγμα 3.4

```
[[while B do C]]=
        ...[[B]]...[[C]]...[[while B do C)]]...
```

- Δεν ισχύει και πάλι ο κανόνας της συνθεσιμότητας.
- Η εξίσωση όμως λύνεται
  - Χρήση της λύσης για ορισμό σημασιολογίας του `while`, που υπακούει στον κανόνα της συνθεσιμότητας