Σχεδίαση αλγορίθμων

Η έννοια του αλγορίθμου

- Αλγόριθμος: σύνολο βημάτων για την παραγωγή ενός αποτελέσματος

πληροφορίες εισόδου (π.χ., εργάσιμες ώρες, ύψος αποδοχών)

μετασχηματισμός της εισόδου σε έξοδο (επεξεργασία)

πληροφορίες εξόδου (π.χ., μισθός)

Αναπαράσταση ενός αλγορίθμου

- Έκφραση με πεπερασμένο πλήθος δηλώσεων
- Κατανόηση κατά μοναδικό τρόπο της σημασίας των δηλώσεων από τους αναγνώστες
- Κατανόηση κατά μοναδικό τρόπο της σημασίας των δηλώσεων από κάποιο επεξεργαστή
- Ικανότητα εκτέλεσης των δηλώσεων από κάποιο επεξεργαστή

Δομικές ιδιότητες ενός αλγορίθμου

- Εκτελεσιμότητα κάθε συγκεκριμένου βήματος
- Τερματισμός υπό οποιεσδήποτε συνθήκες?

Αναπαράσταση σε φυσική γλώσσα

- Δύο αλληλένδετες ιδιότητες: εκφραστικότητα & ασάφεια
- Λεξιλογικός πλούτος
- Πολυπλοκότητα συντακτικών κανόνων
- Εξάρτηση από τα συμφραζόμενα
- Ιδιωματισμοί
- Χρήση σχημάτων λόγου

Αναπαράσταση σε επινοημένη γλώσσα

- Περιορισμός της εκφραστικότητας προκειμένου να αποφευχθεί η ασάφεια
- Περιορισμένο λεξιλόγιο: λίγες λέξεις, όχι συνώνυμα
- Περιορισμένοι συντακτικοί κανόνες και όχι εναλλακτική σύνταξη

Γλώσσες προγραμματισμού

- Επινοημένες γλώσσες αναπαράστασης αλγορίθμων κατά τρόπο κατανοητό (επομένως εκτελέσιμο) από ηλεκτρονικό υπολογιστή

- Πληθώρα γλωσσών (100x) από δεκαετία '60 : Fortran, Basic, Pascal, C, Java, Prolog

- Βασικοί λόγοι παραγωγής νέων γλωσσών :
  α) εύρος και διαφορές πεδίων εφαρμογής
  β) νέα υποδείγματα προγραμματισμού (δομημένος, παράλληλος, διαδικτυακός)
  γ) τάση να ξαναανακαλύπτουμε τον τροχό

- Κάθε γλώσσα προγραμματισμού έχει το δικό της λεξιλόγιο (μαθηματικά σύμβολα -αγγλικές λέξεις) & τους γραμματικούς κανόνες.
  π.χ., multiply price by quantity giving cost
  cost:=price*quantity

Στόχοι μιας γλώσσας προγραμματισμού

- Απλότητα λεξιλογίου και σύνταξης
- Εύκολη και περιεκτική έκφραση αλγορίθμων στη συγκεκριμένη περιοχή εφαρμογών
- Η γλώσσα πρέπει να είναι άμεσα κατανοητή από τον υπολογιστή
- Ελαχιστοποίηση των λαθών μεταγραφής ενός αλγορίθμου σε πρόγραμμα
- Αμεσότητα κατανόησης από τον άνθρωπο
- Δυνατότητα επίδειξης της ορθής εκτέλεσης με την απλή ανάγνωση

Όλοι οι στόχοι είναι πάντα συμβατοί???

Συντακτικό & σημασιολογία

- Συντακτικό (syntax) μιας γλώσσας: το σύνολο των γραμματικών κανόνων που καθορίζουν τη «νόμιμη χρήση» των συμβόλων μιας γλώσσας

- Συντακτικά ορθό πρόγραμμα (syntactically correct)
- Συντακτικό λάθος (syntax error): Απόκλιση από το συντακτικό της γλώσσας

- Σημασιολογία (semantics): Η σημασία συγκεκριμένων μορφών έκφρασης σε μία γλώσσα

Αλληλεξάρτηση ως προς την κατανόηση :

- η κατανόηση της σημασίας στηρίζεται στην κατανόηση της σύνταξης, και αντίστροφα

Ημι-ανεξαρτησία ως προς την ορθότητα :

- η συντακτική ορθότητα δεν οδηγεί απαραίτητα σε σημασιολογική ορθότητα

Η έννοια της λογικής ορθότητας

- Λογικά ορθό αποτέλεσμα : ένα αποτέλεσμα αλγορίθμου το οποίο είναι ορθό, σύμφωνα με
  (α) την πρόθεση του κατασκευαστή, και
  (β) το υφιστάμενο γνωστικό και θεωρητικό μας υπόβαθρο

- περιφέρεια κύκλου := ακτίνα * π ?
- διάρκεια αεροπ. ταξιδιού := ώρα άφιξης – ώρα αναχώρησης?

- λογικό σφάλμα: η παραγωγή λογικά μη ορθών αποτελεσμάτων για συγκεκριμένες εισόδους ή υπό συγκεκριμένες συνθήκες
- λογική ορθότητα: η απουσία λογικών σφαλμάτων

Έλεγχος σημασίας μιας ορθής σύνταξης

- Γνώση των εννοιών κάποιου κόσμου, των γνωρισμάτων που έχουν οι έννοιες αυτές και των σχέσεων μεταξύ τους
- Ταυτοποίηση των λέξεων και εκφράσεων με έννοιες, γνωρίσματα και σχέσεις
- Έλεγχος εναρμόνισης εννοιών, γνωρισμάτων και σχέσεων

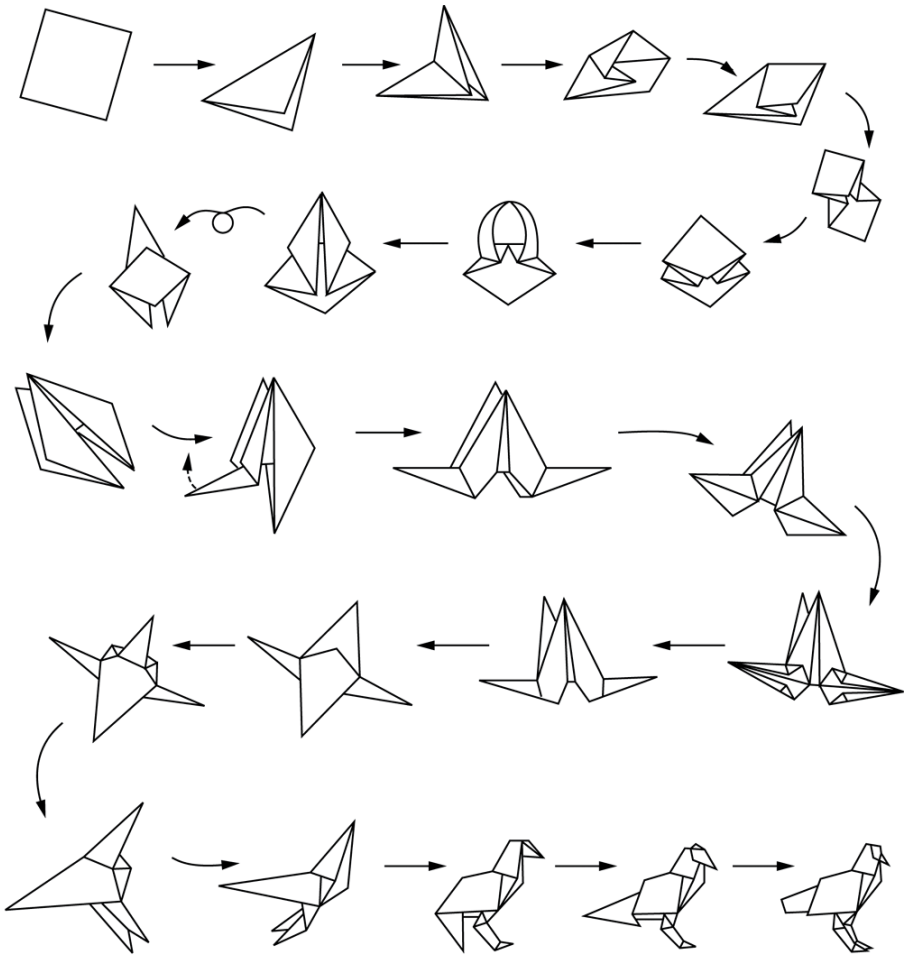
- Συντακτικά ορθές προτάσεις :

  - Ο ελέφαντας έφαγε το φιστίκι,
    Το φιστίκι έφαγε τον ελέφαντα
  - Τύπωσε το όνομα του 13ου μήνα
  - Σκέψου ένα αριθμό από το 1 έως το 13;
    Κάλεσε αυτό τον αριθμό
    Τύπωσε το όνομα του Νιοστού μήνα του χρόνου

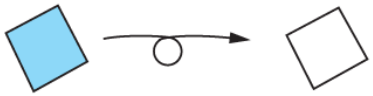
Αναπαράσταση Αλγορίθμου

- Απαιτεί καλά ορισμένα αρχέτυπα (primitives)
- Μια συλλογή αρχετύπων συνιστά μια γλώσσα προγραμματισμού.

Κατασκευή πουλιού από ένα τετράγωνο κομμάτι χαρτί



Αρχέτυπα οριγκάμι

Σύνταξη	Σημασιολογία
	Αναποδογύρισμα του χαρτιού όπως εδώ 
Σκίαση στη μία πλευρά του χαρτιού	Ξεχωρίζει τις δύο διαφορετικές πλευρές του χαρτιού όπως εδώ 
	Αναπαριστά ένα εσωτερικό τσάκισμα ("κοιλάδα") έτσι ώστε το  αναπαριστά 
	Αναπαριστά ένα εξωτερικό τσάκισμα ("κορυφή") έτσι ώστε το  αναπαριστά 
	Δίπλωμα έτσι ώστε το  παράγει 
	Εφαρμογή πίεσης έτσι ώστε το  παράγει 

Συνοψίζοντας

- Ένας επεξεργαστής πρέπει να είναι ικανός να:
  1. Κατανοεί τα σύμβολα με τα οποία ένα βήμα είναι εκφρασμένο
  2. Προσδίδει σημασία στο βήμα σαν συνάρτηση των λειτουργιών που πρέπει να πραγματοποιηθούν
  3. Πραγματοποιεί τις κατάλληλες λειτουργίες

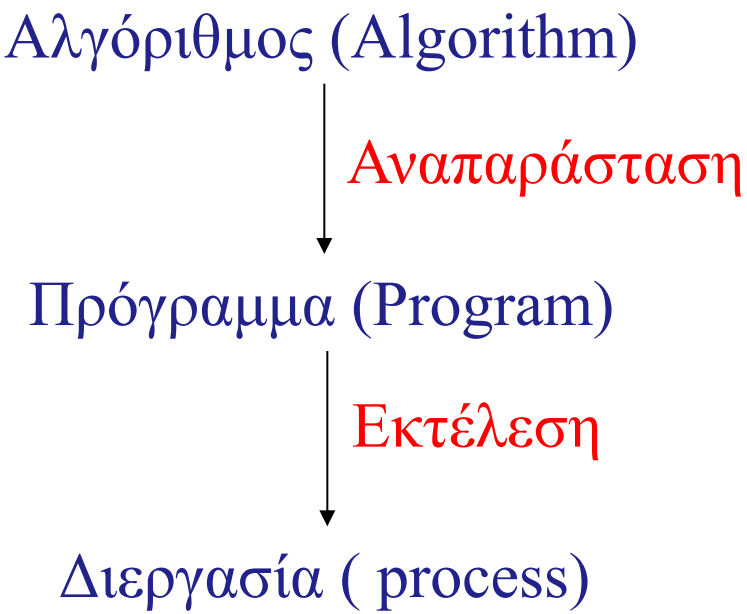
Εντοπισμός συντακτικών λαθών (εκτέλεση από μεταφραστή - translator)

Εντοπισμός ορισμένων σημασιολογικών λαθών (εκτέλεση από μεταφραστή - translator)

Εντοπισμός επιπλέον σημασιολογικών λαθών & λογικών λαθών

Ορισμός του Αλγορίθμου

Ο αλγόριθμος είναι ένα **διατεταγμένο** σύνολο σαφώς **ορισμένων**, **εκτελέσιμων** βημάτων, το οποίο ορίζει μια **τερματιζόμενη(?)** διαδικασία



Πώς λύνουμε προβλήματα

G. Polya 1945

- Κατανοούμε το πρόβλημα
- Βρίσκουμε μια αλγοριθμική διαδικασία που θα μπορούσε να λύνει το πρόβλημα
- Μορφοποιούμε τον αλγόριθμο και τον εκτελούμε
- Αξιολογούμε την απόδοση της λύσης ως προς την ορθότητα και την αποδοτικότητα να λύνει παρόμοια προβλήματα

Κατανοώντας ένα πρόβλημα

- ο Α πρέπει να βρει τις ηλικίες των τριών παιδιών του Β. Ο Β λέει ότι το γινόμενο των ηλικιών των παιδιών είναι 36. ο Α ζητάει άλλο ένα στοιχείο και ο Β του λέει το άθροισμα των ηλικιών. Ο Α καταλαβαίνει ότι χρειάζεται ένα ακόμα στοιχείο και ο Β του λέει ότι το μεγαλύτερο παιδί παίζει πιάνο. Ο Α βρίσκει το αποτέλεσμα.

- Ποιες είναι οι ηλικίες των παιδιών;

Κατανοώντας ένα πρόβλημα

a. Triples whose product is 36

(1,1,36)	(1,6,6)
(1,2,18)	(2,2,9)
(1,3,12)	(2,3,6)
(1,4,9)	(3,3,4)

b. Sums of triples from part (a)

1 + 1 + 36 = 38	1 + 6 + 6 = 13
1 + 2 + 18 = 21	2 + 2 + 9 = 13
1 + 3 + 12 = 16	2 + 3 + 6 = 11
1 + 4 + 9 = 14	3 + 3 + 4 = 10

Το πρώτο βήμα

- Πριν από ένα αγώνα οι Α,Β,Γ,Δ έκαναν τις ακόλουθες προβλέψεις
  - Ο Α προέβλεψε ότι θα νικούσε ο Β
  - Ο Β προέβλεψε ότι ο Δ θα ήταν τελευταίος
  - Ο Γ προέβλεψε ότι ο Α θα ήταν τρίτος
  - Ο Δ προέβλεψε ότι η πρόβλεψη του Α θα ήταν σωστή

- Μόνο μία από αυτές τις προβλέψεις ήταν σωστή και αυτή ήταν η πρόβλεψη του νικητή. Με ποια σειρά τερμάτισαν;

- Απαιτείται δημιουργική συνεισφορά από τον επίδοξο λύτη
- Τεχνικές (να λυθεί το πρόβλημα προς τα πίσω, αναζήτηση σχετικού προβλήματος, βηματική εκλέπτυνση, top down, bottom up κτλ)
- ΔΕΝ βγάζουμε μια λύση για ένα συγκεκριμένο πρόβλημα αλλά ένα γενικό αλγόριθμο που αντιμετωπίζει όλες τις περιπτώσεις

Ανάλυση ενός αλγορίθμου σε βήματα

- Από την αρχική διατύπωση ενός αλγορίθμου μπορεί να λείπουν λεπτομέρειες που υπονοούνται από τη γνώση του κατασκευαστή (π.χ., οδηγίες μετακίνησης σε πόλη)

- Επίτευξη επιθυμητού αποτελέσματος υπό συνθήκες (π.χ., υπολογισμός χρόνου ταξιδιού)

- Ανάγκη προσθήκης λεπτομερειών μέχρι να αποκατασταθεί η ορθή εκτελεσιμότητα

- Ανάγκη διάσπασης σε λεπτομερή και εκτελέσιμα βήματα (κατά βήμα ανάλυση ή από πάνω προς τα κάτω σχεδίαση – top down design)

- Ύπαρξη σχεδιαστικής προσέγγισης από κάτω προς τα πάνω (bottom up design)

Μείωση της πολυπλοκότητας

- έστω C(p), E(p) η πολυπλοκότητα και ο μόχθος επίλυσης ενός προβλήματος
- προφανώς, $C(p) > C(q) \Rightarrow E(p) > E(q)$

- η επίλυση δύο προβλημάτων p1, p2 ταυτόχρονα είναι προφανώς δυσκολότερη από την επίλυση ενός κάθε φορά
- $C(p1+p2) > C(p1) + C(p2) \Rightarrow E(p1) + E(p2) < E(p1+p2)$

- αν διασπάσουμε ένα πρόβλημα σε μικρότερα υπο-προβλήματα μειώνουμε το μόχθο επίλυσης
- ανάγκη διάσπασης σε μικρότερα υπο-προβλήματα

Παράδειγμα βηματικής ανάλυσης

- φτιάξε ένα φλιτζάνι ζεστού καφέ

{φτιάξε καφέ}
1. βράσε νερό
2. βάλε καφέ στο φλιτζάνι
3. πρόσθεσε νερό στο φλιτζάνι

{1. βράσε νερό}
1.1. γέμισε τη χύτρα
1.2. άναψε το μάτι της κουζίνας
1.3. περίμενε μέχρι να βράσει
1.4. σβήσε τη χύτρα

Παράδειγμα βηματικής ανάλυσης (συνέχ.)

{1.1. γέμισε τη χύτρα}
1.1.1. βάλε τη χύτρα κάτω από τη βρύση
1.1.2. άνοιξε τη βρύση
1.1.3. περίμενε μέχρι να γεμίσει η χύτρα
1.1.4. κλείσε τη βρύση

{1.2. άναψε τη χύτρα}
. . .

{2. βάλε καφέ στο φλυτζάνι }
. . .

παράδειγμα βηματικής ανάλυσης (συνέχ.)

αρχικός αλγόριθμος	*πρώτη ανάλυση*	*δεύτερη (τελική) ανάλυση*
(1) βράσε νερό	(1.1) γέμισε τη χύτρα	(1.1.1) βάλε τη χύτρα κάτω από τη βρύση
		(1.1.2) άνοιξε τη βρύση
		(1.1.3) περίμενε μέχρι να γεμίσει η χύτρα
		(1.1.4) κλείσε τη βρύση
	(1.2) άναψε το μάτι	
	(1.3) περίμενε μέχρι να βράσει	(1.3.1) περίμενε μέχρι να σφυρίξει η χύτρα
	(1.4) σβήσε τη χύτρα	
(2) βάλε καφέ στο φλυτζάνι	(2.1) άνοιξε το δοχείο του καφέ	(2.1.1) πάρε το δοχείο του καφέ από το ντουλάπι
		(2.1.2) βγάλε το καπάκι από το δοχείο
	(2.2) βγάλε ένα κουταλάκι καφέ	
	(2.3) βάλε ένα κουταλάκι καφέ στο φλυτζάνι	
	(2.4) κλείσε το δοχείο	(2.4.1) βάλε το καπάκι στο δοχείο (2.4.2) βάλε το δοχείο στο ράφι
(3) πρόσθεσε νερό	(3.1) βάλε νερό στο φλυτζάνι από τη χύτρα, μέχρι να γεμίσει	

παράδειγμα βηματικής ανάλυσης (συνέχ.)

{φτιάξε ένα φλυτζάνι καφέ}

{πρώτα βράσε νερό}
(1.1.1) βάλε τη χύτρα κάτω από τη βρύση
(1.1.2) άνοιξε τη βρύση
(1.1.3) περίμενε μέχρι να γεμίσει η χύτρα
(1.1.4) κλείσε τη βρύση
(1.2) άναψε τη χύτρα
(1.3.1) περίμενε μέχρι να σφυρίξει η χύτρα
(1.4) σβήσε τη χύτρα

{μετά βάλε καφέ στο φλυτζάνι}
(2.1.1) πάρε το δοχείο του καφέ από το ντουλάπι
(2.1.2) βγάλε το καπάκι από το δοχείο
(2.2) βγάλε ένα κουταλάκι καφέ
(2.3) βάλε ένα κουταλάκι καφέ στο φλυτζάνι
(2.4.1) βάλε το καπάκι στο δοχείο
(2.4.2) βάλε το δοχείο στο ράφι

{στο τέλος πρόσθεσε νερό}
(3.1) βάλε νερό στο φλυτζάνι από τη χύτρα, μέχρι να γεμίσει

Αλγοριθμικές δομές ελέγχου : ακολουθία

- Ακολουθία (βημάτων): μια αλγοριθμική δομή ελέγχου η οποία διευθετεί 2 ή περισσότερα βήματα σε μια γραμμική διάταξη

- Συμβολισμός μιας δομής ακολουθίας :

```
start
        βήμα 1
        . . .
        βήμα N
end
```

- Παράθεση με στηλοθέτηση
- Χρήση δεσμευμένων λέξεων

Σημασιολογία εκτέλεσης μιας ακολουθίας

- Η εκτέλεση αρχίζει με το πρώτο βήμα της ακολουθίας
- Όλα τα βήματα εκτελούνται με τη σειρά που γράφονται
- Κάθε βήμα εκτελείται ακριβώς μία φορά (δεν υπάρχουν παραλείψεις ή επαναλήψεις βημάτων)
- Η εκτέλεση ολοκληρώνεται με το τελευταίο βήμα της ακολουθίας

Αξιολόγηση της δομής ακολουθίας

- Η ακολουθία είναι η πλέον απλή δομή ελέγχου αλγορίθμων
- Παρουσιάζει «δυσκαμψία» με την έννοια της αδυναμίας προσαρμογής σε συγκεκριμένες συνθήκες των εισόδων ή του περιβάλλοντος
- Δεν μπορεί να αποτυπώσει εναλλακτικές συμπεριφορές, όπως
  α) παράλειψη κάποιου βήματος
  β) πραγματοποίηση κάποιου εναλλακτικού βήματος
  γ) επανάληψη κάποιου βήματος

- π.χ., Υπολογισμός μέσου όρου (ενδεχόμενο πλήθος = 0), υπολογισμός αθροίσματος χωρίς να είναι γνωστό εκ των προτέρων το πλήθος των προσθετέων

Αλγοριθμικές δομές ελέγχου : επιλογή

- απλή επιλογή : μια αλγοριθμική δομή ελέγχου η οποία συσχετίζει ένα σύνολο αλγοριθμικών βημάτων (κλάδος επιλογής) με μια συνθήκη (συνθήκη επιλογής)

- συμβολισμός μιας δομής απλής επιλογής :
  if συνθήκη επιλογής
  then κλάδος επιλογής

- παράθεση με στηλοθέτηση
- χρήση δεσμευμένων λέξεων

Η δομή της διπλής επιλογής

- διπλή επιλογή : μια αλγοριθμική δομή ελέγχου η οποία συσχετίζει δύο κλάδους επιλογής με μια συνθήκη (συνθήκη επιλογής) και την αντίθετη της συνθήκης αυτής

- συμβολισμός μιας δομής διπλής επιλογής :
  if συνθήκη επιλογής
  then κλάδος επιλογής 1
  else κλάδος επιλογής 2

- παράθεση με στηλοθέτηση
- χρήση δεσμευμένων λέξεων

Η δομή της πολλαπλής επιλογής

- πολλαπλή επιλογή : μια αλγοριθμική δομή ελέγχου η οποία συσχετίζει δύο ή περισσότερους κλάδους επιλογής με ισάριθμες τιμές (τιμές επιλογής) μιας μεταβλητής (μεταβλητή επιλογής)

- συμβολισμός μιας δομής πολλαπλής επιλογής :

```
switch μεταβλητή επιλογής
        τιμή 1 : κλάδος επιλογής 1
        . . .
        τιμή N : κλάδος επιλογής N
end
```

Η δομή της πλήρους πολλαπλής επιλογής

- πλήρης πολλαπλή επιλογή : μια δομή πολλαπλής επιλογής η οποία συσχετίζει έναν επιπλέον κλάδο επιλογής με τις υπόλοιπες τιμές της μεταβλητή επιλογής

- συμβολισμός μιας δομής πλήρους πολλαπλής επιλογής :

```
switch μεταβλητή επιλογής
        τιμή 1 : κλάδος επιλογής 1
        . . .
        τιμή N : κλάδος επιλογής N
        else : κλάδος επιλογής N+1
end
```

Εμφωλευμένες (nested) δομές επιλογής

- οι κλάδοι μιας δομής επιλογής μπορούν να είναι και οι ίδιοι ή να περιέχουν δομές επιλογής

```
{υπολογισμός του μεγαλύτερου από τρεις αριθμούς α, β, γ}
if x >= y
    then if x >= z
                then μεγαλύτερος := x
                else μεγαλύτερος := z
    else if y >= z
                then μεγαλύτερος := y
                else μεγαλύτερος := z
```

Ισοδυναμία των δομών επιλογής

- όλες οι δομές επιλογής μπορούν να εκφραστούν ως δομές διπλής επιλογής

δομή απλής επιλογής :
if συνθήκη επιλογής
    then βήμα
    else null

- δομή πολλαπλής επιλογής :
  if μεταβλητή επιλογής = τιμή 1
      then βήμα 1
      else if μεταβλητή επιλογής = τιμή 2
          then βήμα 2
          else if ...

- δομή πλήρους πολλαπλής επιλογής : ομοίως

Διαδρομή και ίχνος εκτέλεσης

- Διαδρομή εκτέλεσης : μια δυνατή σειρά εκτέλεσης για τα βήματα ενός αλγορίθμου
- Συνώνυμα : μονοπάτι εκτέλεσης, λογικό μονοπάτι

- Ίχνος εκτέλεσης : η συγκεκριμένη διαδρομή εκτέλεσης που ακολουθήθηκε για ένα συγκεκριμένο συνδυασμό τιμών εισόδου

- Η διαδρομή εκτέλεσης ορίζεται σε χρόνο σχεδίασης
- Το ίχνος εκτέλεσης δημιουργείται σε χρόνο εκτέλεσης

Αλγοριθμικές δομές ελέγχου : επανάληψη

{βρες ένα τηλέφωνο από τον τηλεφωνικό κατάλογο}
start
πάρε το πρώτο όνομα
if το όνομα αυτό είναι το όνομα που ψάχνουμε
then πάρε το τηλέφωνο
else βρες το επόμενο όνομα
if το όνομα αυτό είναι το όνομα που ψάχνουμε
then πάρε το τηλέφωνο
else . . .

- Είμαστε σίγουροι ότι το όνομα υπάρχει στον κατάλογο ;
- Γνωρίζουμε πόσες δομές επιλογής πρέπει να γράψουμε ;

Η ανάγκη της επανάληψης (συνέχ.)

```
{βρες το μικρότερο πρώτο φυσικό αριθμό μετά το Ν}
start
δοκιμαστικός αριθμός := Ν
αύξησε το δοκιμαστικό αριθμό κατά 1
if ο δοκιμαστικός αριθμός είναι πρώτος
        then τύπωσε το δοκιμαστικό αριθμό
        else αύξησε το δοκιμαστικό αριθμό κατά 1
                if ο δοκιμαστικός αριθμός είναι πρώτος
                  then τύπωσε το δοκιμαστικό αριθμό
                  else . . .
```

- γνωρίζουμε πόσες δομές επιλογής πρέπει να γράψουμε ;

Η ανάγκη της επανάληψης (συνέχ.)

```
{βρες εάν ο φυσικός αριθμός Ν είναι πρώτος}
start
if το 2 δεν διαιρεί το Ν ακριβώς
        then if το 3 δεν διαιρεί το Ν ακριβώς
                then if . . .
                        then if το Ν-1 δεν διαιρεί το Ν ακριβώς
                                then ο Ν είναι πρώτος
                                else ο Ν δεν είναι πρώτος
                        else
                else...
        else ο Ν δεν είναι πρώτος
τέλος
```

- γνωρίζουμε πόσες δομές επιλογής πρέπει να γράψουμε
- γιατί να γράψουμε Ν-2 φορές την ίδια δομή επιλογής;

Η ανάγκη της επανάληψης (συνέχ.)

- μια δομή ελέγχου που να υπονοεί (ως σημασιολογία εκτέλεσης) την επανάληψη κάποιων βημάτων είναι αναγκαία για λόγους :

- τερματισμού του αλγορίθμου υπό κάποια συνθήκη (παράδειγμα 1)
- αναπαράστασης του αλγορίθμου με πεπερασμένο πλήθος δηλώσεων (παραδείγματα 1 και 2)
- οικονομίας ως προς την αναπαράσταση (παράδειγμα 3)

Η δομή της μετα-ελεγχόμενης επανάληψης

- μετα-ελεγχόμενη επανάληψη : μια δομή ελέγχου η οποία συσχετίζει ένα σύνολο αλγοριθμικών βημάτων (σώμα επανάληψης) με μια συνθήκη (συνθήκη τερματισμού)

- συμβολισμός μιας δομής μετα-ελεγχόμενης επανάληψης :
- **repeat**
- σώμα επανάληψης
- **until** συνθήκη τερματισμού

- Ονομάζεται ανακύκλωση ή βρόχος (loop)

Παράδειγμα μετα-ελεγχόμενης επανάληψης

```
{βρες ένα τηλέφωνο από τον τηλεφωνικό κατάλογο}
start
πάρε το πρώτο όνομα
repeat
        if το όνομα αυτό είναι το όνομα που ψάχνουμε
                then πάρε το τηλέφωνο
                else βρες το επόμενο όνομα
until  (το όνομα αυτό είναι το όνομα που ψάχνουμε
        ή ο τηλεφωνικός κατάλογος εξαντληθεί)
end
```

Παράδειγμα μετα-ελεγχόμενης επανάληψης

```
{Έλεγξε αν ο αριθμός είναι πρώτος }
start
   θέσε πιθανό διαιρέτη ίσο με 2
   repeat
        διαίρεσε υποψήφιο πρώτο με πιθανό διαιρέτη
        πρόσθεσε 1 στον πιθανό διαιρέτη
   until  (η διαίρεση είναι ακριβής ή διαιρέτης > √υποψήφιου πρώτου)
   if καμία διαίρεση δεν είναι ακριβής
        then ο υποψήφιος είναι πρώτος
        else ο υποψήφιος δεν είναι πρώτος
end
```

Αντι-παράδειγμα μετα-ελεγχόμενης επανάληψης

```
{βρες τον μεγαλύτερο αριθμό μιας λίστας}
start
μεγαλύτερος := ο πρώτος αριθμός της λίστας
repeat
    πάρε τον επόμενο αριθμό της λίστας
    if επόμενος αριθμός > μεγαλύτερος
        then μεγαλύτερος := επόμενος αριθμός
until η λίστα εξαντληθεί
end
```

- πιθανότητα σφάλματος ;

Αξιολόγηση της δομής μετα-ελεγχόμενης επανάληψης

- επιτρέπει την αναπαράσταση με πεπερασμένο πλήθος δηλώσεων ενός ακαθόριστου πλήθους αλγοριθμικών βημάτων
- η συνθήκη τερματισμού θα πρέπει να είναι καλά μελετημένη ορθή ανάλυση γεγονότων που μπορεί να προκαλούν τερματισμό της επανάληψης (π.χ. εξάντληση τηλεφωνικού καταλόγου)
- η δομή αυτή δεν εξυπηρετεί στην περίπτωση που η πρώτη εκτέλεση του σώματος μπορεί να προκαλέσει σφάλμα

Η δομή της προ-ελεγχόμενης επανάληψης

- προ-ελεγχόμενη επανάληψη : μια δομή ελέγχου η οποία συσχετίζει ένα σύνολο αλγοριθμικών βημάτων (σώμα επανάληψης) με μια συνθήκη (συνθήκη συνέχειας)

- συμβολισμός μιας δομής προ-ελεγχόμενης επανάληψης :

```
while συνθήκη συνέχειας
        σώμα επανάληψης
end
```

Παράδειγμα προ-ελεγχόμενης επανάληψης

```
{βρες τον μεγαλύτερο αριθμό μιας λίστας}
start
    μεγαλύτερος := ο πρώτος αριθμός της λίστας
    while η λίστα δεν έχει εξαντληθεί
            πάρε τον επόμενο αριθμό της λίστας
            if επόμενος αριθμός > μεγαλύτερος
                    then μεγαλύτερος := επόμενος αριθμός
    end
end
```

- σωστή συνθήκη συνέχειας
- η πρώτη εκτέλεση δεν μπορεί να προκαλέσει σφάλμα

Παράδειγμα προ-ελεγχόμενης επανάληψης (συνέχ.)

```
{βρες τον μεγαλύτερο αριθμό μιας λίστας}
start
if η λίστα δεν είναι κενή
        then μεγαλύτερος := ο πρώτος αριθμός της λίστας
                while η λίστα δεν έχει εξαντληθεί
                        πάρε τον επόμενο αριθμό της λίστας
                        if επόμενος αριθμός > μεγαλύτερος
                        then μεγαλύτερος := επόμενος αριθμός
                end
        else μεγαλύτερος := 0
end
```

Παράδειγμα προ-ελεγχόμενης επανάληψης

```
start
    αριθμός :=1;
    while αριθμός ≠ 6
                αριθμός := αριθμός + 2
                τύπωσε το περιεχόμενο του αριθμού
    end
end
```

Αξιολόγηση της δομής προ-ελεγχόμενης επανάληψης

- επιτρέπει την αναπαράσταση με πεπερασμένο πλήθος δηλώσεων ενός ακαθόριστου πλήθους αλγοριθμικών βημάτων
- η συνθήκη συνέχειας θα πρέπει να είναι καλά μελετημένη
- ορθή ανάλυση γεγονότων που μπορεί να προκαλούν τερματισμό της επανάληψης (π.χ. εξάντληση τηλεφωνικού καταλόγου)

Παράδειγμα : Αλγόριθμος σειριακής αναζήτησης (ταξινομημένη λίστα)

```
start
    if η λίστα είναι κενή
          then ο αριθμός δεν υπάρχει
          else επέλεξε το πρώτο στοιχείο για έλεγχο
    while (ελεγχόμενο στοιχείο < επιθυμητή τιμή &
                υπάρχουν επόμενα στοιχεία)
                   πάρε το επόμενο στοιχείο για έλεγχο
     end
    if (επιθυμητή τιμή == ελεγχόμενο στοιχείο)
        then ανακοίνωσε ότι ο αριθμός υπάρχει
        else ανακοίνωσε ότι ο αριθμός δεν υπάρχει
```

Αποδοτικότητα του αλγόριθμου σειριακής αναζήτησης

- Αν η λίστα έχει Ν στοιχεία τότε κατά μέσο όρο γίνονται Ν/2 συγκρίσεις
- Π.χ., αν η λίστα περιέχει 30.000 εγγραφές τότε θα γίνονται κατά μέσο όρο 15.000 συγκρίσεις
- Αν η ανάκτηση και η σύγκριση με μία εγγραφή χρειάζεται 10 msec τότε ο μέσος χρόνος θα είναι 150 sec (ή 2.5 min). Αν η πράξη χρειάζονταν 1 msec τότε θα χρειαζόμασταν κατά μέσο όρο 15 sec