Κεφάλαιο 1.3-1.4:
Εισαγωγή Στον Προγραμματισμό

(Διάλεξη 2)

Διδάσκων: Δημήτρης Ζεϊναλιπούρ

Περιεχόμενα

- Εισαγωγικές Έννοιες - Ορισμοί
- Ο κύκλος ανάπτυξης προγράμματος
- Παραδείγματα

Πότε χρησιμοποιούμε υπολογιστή?

- Χρήση υπολογιστή αν:
  - Έχουμε πολλά στοιχεία να επεξεργαστούμε.
  - Παράγονται πολλά αποτελέσματα.
  - Μέθοδος επίλυσης εξαιρετικά πολύπλοκη για ένα άνθρωπο.
  - Χρησιμοποιούμε την ίδια μέθοδο πολλές φορές.
- **Να θυμάστε:**
  - Ο Η/Υ εκτελεί «πράξεις» γρηγορότερα από τον άνθρωπο.
  - Ο Η/Υ μπορεί να «θυμάται» περισσότερα πράγματα από τον άνθρωπο.
  - Ο Η/Υ μπορεί να εκτελέσει με «ακρίβεια» μια λογική σειρά εντολών.

Εισαγωγή - Ορισμοί

- **Αλγόριθμος:** ονομάζουμε μια ταξινομημένη ακολουθία μη διφορούμενων βημάτων που οδηγούν στη λύση ενός προβλήματος.
- **Πρόγραμμα:** ονομάζουμε την αναπαράσταση ενός η πολλών αλγορίθμων σε μορφή κατανοητή από τον υπολογιστή
- **Προγραμματισμός:** Η διαδικασία της ανάπτυξης ενός αλγορίθμου σε συνδυασμό με την συγγραφή του προγράμματος.
- **Γλώσσα Προγραμματισμού:** Το σύνολο των **γραμματικών** και **συντακτικών** κανόνων που μας επιτρέπει να δίνουμε εντολές στον Η/Υ μέσω ενός προγράμματος.
- **Κύκλος Ανάπτυξης Προγράμματος:** Η διαδικασία που ακολουθούμε για την ανάπτυξη ενός προγράμματος

Ο κύκλος ανάπτυξης προγράμματος

- Ο κύκλος ανάπτυξης προγράμματος αναλύεται σε έξι βασικά βήματα:
  1. Περιγραφή του προβλήματος, καθορισμός απαιτήσεων,
  2. Ανάλυση προβλήματος, προσδιορισμός της λύσης
  3. Σχεδίαση της λύσης του προβλήματος
     - Ανάπτυξη αλγορίθμου
     - Σχεδιασμός διαγράμματος ροής
     - Δημιουργία ψευδοκώδικα
  4. Κωδικοποίηση σε γλώσσα προγραμματισμού,
  5. Έλεγχος, διόρθωση λαθών,
  6. Συντήρηση προγράμματος, τεκμηρίωση.

1. Περιγραφή του προβλήματος & καθορισμός απαιτήσεων

- **Απομονώνουμε καταγράφουμε με απλά βήματα τις πραγματικές συνιστώσες** ενός προβλήματος, τοποθετώντας τις σε λογική σειρά μεταξύ τους.
  - Ποιο είναι το απτό αποτέλεσμα που πρέπει να προκύψει από την επίλυση του προβλήματος?

- **Αποσαφηνίζουμε τους στόχους** που επιδιώκουμε να υλοποιήσουμε με αναλυτικό τρόπο προκειμένου να καταγραφεί το πλαίσιο απαιτήσεων της όλης προσπάθειας.
  - Μήπως οι απαιτήσεις για την επίλυση του προβλήματος το καθιστούν την αυτοματοποίηση της διαδικασίας μη συμφέρουσα?

2. Ανάλυση του προβλήματος & προσδιορισμός της λύσης

- Σκιαγραφούμε ένα προσχέδιο της επίλυσης του προβλήματος.
- Ελέγχουμε αν η λύση καλύπτει τους στόχους που έχουν τεθεί και αν παράγει τα επιθυμητά δεδομένα εξόδου.
- Διερευνούμε την πιθανότητα ύπαρξης περισσότερων λύσεων.
- Επιλέγουμε την βέλτιστη λύση με βάση τις προδιαγραφές που έχουν τεθεί.
- ΠΡΟΣΟΧΗ: ΛΑΘΟΣ ΑΝΑΛΥΣΗ => ΛΑΘΟΣ ΛΥΣΗ

2. Ανάλυση του προβλήματος

- Ως μέρος της ανάλυσης απαντούμε και στις πιο κάτω ερωτήσεις:
  - ποια είναι τα δεδομένα (inputs)
  - ποια είναι τα εξαγώμενα/αποτελέσματα (outputs)
  - τι χρειάζεται να γίνει (πράξεις)

3. Σχεδίαση της λύσης του προβλήματος

- Σκιαγραφούμε ένα προσχέδιο της επίλυσης του προβλήματος
- Αναπτύσσουμε τον αλγόριθμο επίλυσης του προβλήματος
  - ακολουθία αυστηρά δομημένων βημάτων προκειμένου να επιλύσουμε το πρόβλημα.
- Για την περιγραφή της λύσης ενός προγράμματος χρησιμοποιούμε
  - τον ψευδοκώδικα. ή/και
  - το λογικό διάγραμμα

Ορισμός Αλγορίθμου

Ένας αλγόριθμος είναι ένα πεπερασμένο σύνολο εκτελέσιμων και σαφών εντολών που κατευθύνει μία τερματίζουσα διαδικασία

Αλγόριθμοι

- Γιατί να καταγράψω τον αλγόριθμο;
  - **Για ιδίαν χρήση** – Δεν χρειάζεται να ξανασκεφτείτε το πρόβλημα
  - Ώστε άλλοι να μπορούν να το επιλύσουν, χωρίς να ξέρουν πολλά γύρω από αυτό
    - Δεν χρειάζεται να καταλάβουν τις αρχές πίσω από αυτό το πρόβλημα.
    - Απλώς ακολουθούν τις εντολές.
    - Η νοημοσύνη είναι «κωδικοποιημένη στον αλγόριθμο»
  - **Για να σας βοηθήσει να καταλάβατε αν επιλύει ορθά το πρόβλημα, να βρείτε αν είναι αποδοτικός**

Αλγόριθμοι

Παραδείγματα αλγορίθμων

- Οδηγίες πλυντηρίου
- Οδηγίες για συναρμολόγηση επίπλου

- Στην Πληροφορική
  - Ταξινόμηση Λίστας Αριθμών
  - Εύρεση Μέσου Όρου Μιας Λίστας Αριθμών
  - ….

Αφαιρετικότητα

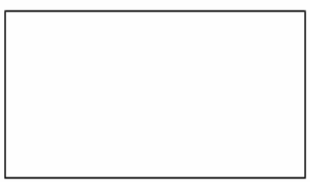
- ***Πολύ σημαντική έννοια στον Προγραμματισμό και γενικά στην Επιστήμη της Πληροφορικής.***
- ***Διακρίβωση του τι γίνεται χωρίς την γνώση του πως γίνεται***
- Η διαδικασία προγραμματισμού αποτελείται απο σχεδιασμό λύσεων σε διάφορα επίπεδα αφαιρετικότητας
- Θυμηθείτε ότι και η ίδια η λειτουργία ενός Η/Υ στηρίζεται σε αυτή την έννοια.

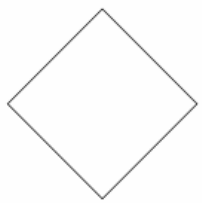
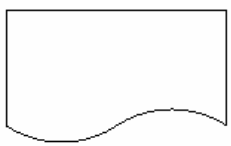
Αλγόριθμοι

- **Ένας αλγόριθμος εκφράζεται σε διάφορα επίπεδα αφαιρετικότητας.**
- Αρχικά προδιαγράφεται τι είναι το δεδομένο και τί το επιδιωκόμενο αποτέλεσμα.
- Αυτό σταδιακά εκλεπτύνεται.
- Η σταδιακή διάσπαση (εκλέπτυνση) συνεχίζεται μέχρις ότου φτάσουμε σε ατομικά υπο-προβλήματα, δηλαδή προβλήματα που δεν είναι λογικό/δυνατό να διασπαστούν περαιτέρω.
- Στο χαμηλότερο επίπεδο ο αλγόριθμος διατυπώνει με σαφήνεια την ακριβή διαδικασία παραγωγής της λύσης του προβλήματος

3. Σχεδίαση της λύσης του προβλήματος Διαγράμματα Ροής

- Σχεδιασμός διαγράμματος ροής
  - Σχηματικός τρόπος αναπαράστασης της ροής των οδηγιών που συνθέτουν έναν αλγόριθμο.

	**Τερματισμός ή Αρχή** BEGIN - END
	Κατεύθυνση της **ροής** του προγράμματος
	Γενικό σύμβολο που δηλώνει **διεργασίες** (μαθηματικές πράξεις, αρχικοποίηση μεταβλητών)

	Σύμβολο **απόφασης**, χρησιμοποιείται όταν υπάρχουν δύο εναλλακτικές διαδρομές
	Σύμβολο **εισόδου, εξόδου.** Χρησιμοποιείται όταν διαβάζονται αποτελέσματα
	Σύμβολο **εκτύπωσης** (οθόνη η εκτυπωτή)

Note: Βοηθητικές Ασκήσεις Επανάληψης στην Ιστοσελίδα!

3. Σχεδίαση της λύσης του προβλήματος Ψευδοκώδικας

- Είναι ένα μίγμα αγγλικών και κοινών σε διάφορες γλώσσες προγραμματισμού όρων (εντολών), που χρησιμοποιούνται για να εκφράσουμε τη λύση.

```
open STUDENT.DAT                    /* Open the file that contains the
                                                student list */
read student_name                   /* Read the first student name */
while not EOF do                    /* Begin the main loop */
    read student_info               /* Read the remaining student info */
    if YEAR=1 then print student_name        /* conditional structure */
    read student_name               /* Read the next student name */
end while
close STUDENT.DAT                   /* Close the input file */
end                                 /* End of program */
```

4. Κωδικοποίηση

- Αξιοποιείται η διαδικασία του σχεδιασμού
- Πραγματοποιείται η συγγραφή του προγράμματος σε μια γλώσσα προγραμματισμού.
- Μέσω **μεταγλωττιστή** (compiler) ή **μεταφραστή** (interpreter) το πρόγραμμα μετατρέπεται σε **γλώσσα μηχανής** η οποία είναι αναγνωρίσιμη από τον υπολογιστή.
- Στο στάδιο αυτό γίνεται ο έλεγχος **συντακτικών** λαθών

5. Έλεγχος λαθών

- Έλεγχος λαθών και διόρθωση προγράμματος:
  - Διορθώνονται πιθανά **λογικά σφάλματα** (σφάλματα που σχετίζονται με τον σχεδιασμό της λύσης).
    - Λύσε με το χέρι το πρόβλημα με ένα σύνολο δεδομένων και σύγκρινε το με τις εξόδους του προγράμματος
  - **Σφάλματα σύνταξης**
    - Σφάλματα που σχετίζονται με το αν χρησιμοποιήσαμε σωστά τη γλώσσα προγραμματισμού στη διάρκεια της υλοποίησης
  - **Σφάλματα Run-time**
    - Σφάλματα κατά τη διάρκεια εκτέλεσης του προγράμματος

6. Συντήρηση προγράμματος

- Συντήρηση προγράμματος:
  - Συγγραφή τεκμηρίωσης
  - Το λογισμικό εγκαθίσταται και ξεκινά η λειτουργία του.
- Το περιβάλλον αλλάζει ➔ αλλάζει και το πρόγραμμα
- Οι χρήστες επιθυμούν (ή χρειάζονται) περισσότερα από το πρόγραμμα

Τεκμηρίωση

- Συνιστούν τεκμηρίωση τα παρακάτω:
  - Συνοπτική περιγραφή των απαιτήσεων
  - Περιγραφή εισόδων, εξόδων, περιορισμών και τύπων
  - Ψευδοκώδικας ή διάγραμμα ροής του αλγορίθμου
  - Ο ίδιος ο πηγαίος κώδικας (source code)
  - Οδηγός για τη χρήση του προγράμματος

Αξιολόγηση Λύσεων

- **Ορθότητα Λύσεων**
  - Αναλυτικές Μέθοδοι – Αποδείξεις
  - Εμπειρικές Μέθοδοι - Δοκιμές
- **Τεκμηρίωση Λύσεων**
  - Σχόλια στο κώδικα
  - Ευκολία κατανοήσεως
- **Εκτίμηση Απόδοσης**
  - Ταχύτητα, ανάγκη σε μνήμη
  - Αναλυτικές/Εμπειρικές Μέθοδοι
- **Επεκτασιμότητα**

Παράδειγμα Προγραμματισμού

- Γράψετε ένα πρόγραμμα το οποίο να μετατρέπει μίλια σε χιλιόμετρα.

Βήμα 1. Κατανόηση

- Τι μας ζητά;
  - Τι μίλια;
    - αγγλικά μίλια, ναυτικά μίλια;
  - Από πού παίρνουμε τις πληροφορίες;
    - Από τον χρήστη, από αρχείο;

Βήμα 2. Ανάλυση

- Δεδομένα (εισόδου): μίλια
- Δεδομένα (εξόδου): χιλιόμετρα
- Άλλα δεδομένα: σχέση 1mile:1.609Km
- Υπολογισμός:

χιλιόμετρα = μίλια * 1.609

- Παράδειγμα: Πόσα Km είναι 10 μίλια?

1.609*10 =

>>>>>> 16.09 χιλιόμετρα

Βήμα 3. Σχεδιασμός Λύσης

- Αλγόριθμος

1. Πάρε τα δεδομένα εισόδου
2. Κάνε τη μετατροπή
3. Παρουσίασε το αποτέλεσμα

- 1η Εκλέπτυνση

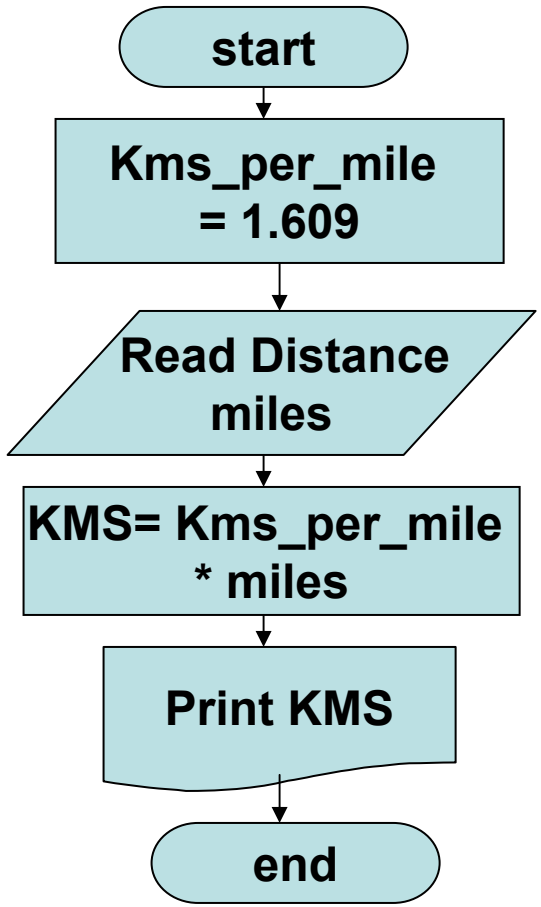
1. Διάβασε τα Μίλια
2. Κάνε τον υπολογισμό: Χλμ.=Μίλια * 1.609
3. Δείξε το αποτέλεσμα στην οθόνη

Βήμα 3. Σχεδιασμός Λύσης

- 2η Εκλέπτυνση

1. Διάβασε τα Μίλια και αποθήκευσε τα στη μεταβλητή miles
2. Κάνε τον υπολογισμό: Kms.=Miles * 1.609
3. Δείξε το αποτέλεσμα στην οθόνη

- Διάγραμμα Ροής



Βήμα 4. Υλοποίηση

```
1.  /*
2.   * Converts distance in miles to kilometers.
3.   */
4.  #include <stdio.h>              /* printf, scanf definitions */
5.  #define KMS_PER_MILE 1.609      /* conversion constant       */
6.
7.  int
8.  main(void)
9.  {
10.       double miles,  /* input - distance in miles.       */
11.              kms;    /* output - distance in kilometers  */
12.
13.       /* Get the distance in miles. */
14.       printf("Enter the distance in miles> ");
15.       scanf("%lf", &miles);
16.
17.       /* Convert the distance to kilometers. */
18.       kms = KMS_PER_MILE * miles;
19.
20.       /* Display the distance in kilometers. */
21.       printf("That equals %f kilometers.\n", kms);
22.
23.       return (0);
24. }
```

6. Σχόλια / Τεκμηρίωση

5. Έλεγχος

Sample Run

```
Enter the distance in miles> 10.00
That equals 16.090000 kilometers.
```

C Language Elements in Miles-to-Kilometers Conversion Program