ΤΕΙ ΚΕΝΤΡΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΑΝΟΙΚΤΑ ΑΚΑΔΗΜΑΙΚΑ ΜΑΘΗΜΑΤΑ

ΤΕΧΝΟΛΟΓΙΑ ΛΟΓΙΣΜΙΚΟΥ Ι

κ. ΠΕΤΑΛΙΔΗΣ

ΤΜΗΜΑ **ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΤΕ**

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

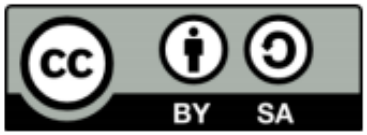
ΕΣΠΑ 2007-2013
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.

- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.



Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «Ανοικτά Ακαδημαϊκά Μαθήματα στο ΤΕΙ Κεντρικής Μακεδονίας» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

13η Διάλεξη

Σχεδίαση συστημάτων/Συνεκτικότητα

Υποέργο 1 «Αυτεπιστασία του τμήματος Πληροφορικής και Επικοινωνιών»,
«Ενίσχυση Σπουδών Πληροφορικής στο ΤΕΙ Σερρών» Μέτρο 2.2, Γ' ΚΠΣ

Συνεκτικότητα

- Αναφέρεται στην εσωτερική συνοχή ενός συστατικού
- Όσο πιο συνεκτικό είναι ένα συστατικό τόσο περισσότερο σχετίζονται μεταξύ τους τα εσωτερικά του μέρη και εξυπηρετούν καλύτερα το συνολικό σκοπό του
- Ένα συστατικό με μεγάλη συνοχή δεν έχει παραπανίσια κομμάτια!

Είδη συνεκτικότητας

Λειτουργική

Σειριακή

Επικοινωνιακή

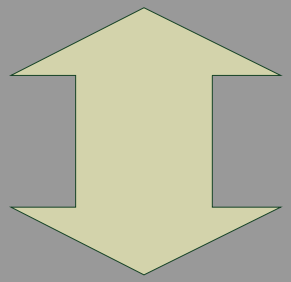
Διαδικασιακή

Χρονική

Λογική

Συμπτωματική

Υψηλή συνεκτικότητα (καλή)



Χαμηλή συνεκτικότητα (κακή)

Υποέργο 1 «Αυτεπιστασία του τμήματος Πληροφορικής και Επικοινωνιών»,
«Ενίσχυση Σπουδών Πληροφορικής στο ΤΕΙ Σερρών» Μέτρο 2.2, Γ' ΚΠΣ

Συμπτωματική συνεκτικότητα

- Όταν τα μέρη δε σχετίζονται μεταξύ τους
- Παράδειγμα:

```
function doItAll(char *par1,int par2)
printNextLine();
ReverseString(par1)
par2 += 7;
return par2
```

- Συνήθως προκύπτει από εντολές του τύπου
  - "Γράψτε μια συνάρτηση που αποτελείται από 10-20 εντολές"

Υποέργο 1 «Αυτεπιστασία του τμήματος Πληροφορικής και Επικοινωνιών», «Ενίσχυση Σπουδών Πληροφορικής στο ΤΕΙ Σερρών» Μέτρο 2.2, Γ' ΚΠΣ

Γιατί είναι κακή;

- Το συστατικό δε συντηρείται εύκολα γιατί είναι δυσνόητο
- Δε μπορεί να επαναχρησιμοποιηθεί

- Διορθώνεται όμως εύκολα!
  - Φτιάξτε τόσα χωριστά συστατικά όσες και οι ενέργειες που έχετε

Λογική συνεκτικότητα

- Όταν διάφορα λογικά σχετιζόμενες συναρτήσεις ή δεδομένα τοποθετούνται στο ίδιο συστατικό
- Για παράδειγμα έχετε μια συνάρτηση η οποία γράφει δεδομένα σε συσκευές. Επιλέγετε τη συσκευή στην οποία θα γράψει περνώντας μια παράμετρο: Αν περαστεί το 1 γράφει σε δίσκο, αν περαστεί το 2 γράφει στη μνήμη κτλ
- Συνήθως συνάρτηση αυτή θα παίρνει και άλλες παραμέτρους που θα τους χρησιμοποιεί ανάλογα με τον κωδικό που περάσατε

Υποέργο 1 «Αυτεπιστασία του τμήματος Πληροφορικής και Επικοινωνιών», «Ενίσχυση Σπουδών Πληροφορικής στο ΤΕΙ Σερρών» Μέτρο 2.2, Γ' ΚΠΣ

Λογική συνεκτικότητα

- Παράδειγμα

```
function code = 7;
writeData(op code, dummy 1, dummy 2, dummy 3);
// dummy 1, dummy 2, and dummy 3 are dummy variables,
// not used if function code is equal to 7
```

- Η `writeData` καλείται με 4 παραμέτρους αλλά μόνο 3 χρησιμοποιούνται όταν ο κωδικός είναι 7

Παράδειγμα

- Το συστατικό στα δεξιά περιέχει κώδικα που διαχειρίζεται όλες τις εγγραφές/αναγνώσεις.
- Βάζετε μια καινούργια συσκευή. (πχ. Ένα νέο δίσκο). Πρέπει να αλλάξετε τα κομμάτια 1,2,3,4,7,8.
- Τι θα γίνει όμως με τις συσκευές που ήδη χρησιμοποιούσαν αυτόν τον κώδικα;

1. Code for all input and output
2. Code for input only
3. Code for output only
4. Code for disk and tape I/O
5. Code for disk I/O
6. Code for tape I/O
7. Code for disk input
8. Code for disk output
9. Code for tape input
10. Code for tape output
⋮ ⋮ ⋮
37. Code for keyboard input

Υποέργο 1 «Αυτεπιστασία του τμήματος Πληροφορικής και Επικοινωνιών»,
«Ενίσχυση Σπουδών Πληροφορικής στο ΤΕΙ Σερρών» Μέτρο 2.2, Γ' ΚΠΣ

Γιατί είναι κακή;

- Οι διεπαφές δεν είναι ξεκάθαρες
- Δεν είναι ξεκάθαρο που βρίσκεται και τι κάνει κάθε κομμάτι
- Η επαναχρησιμοποίηση είναι δύσκολη

Χρονική συνεκτικότητα

- Όταν τα μέρη σχετίζονται με βάση το χρονισμό τους
- Παράδειγμα.
  - Ένα συστατικό που κάνει αρχικοποίηση

class initialize {
Αρχικοποίησε μεταβλητές προγράμματος
Άδειασε τον πίνακα Sales
Διάβασε την πρώτη εγγραφή από τον πίνακα Transactions
Εμφάνισε μήνυμα στην οθόνη

Υποέργο 1 «Αυτεπιστασία του τμήματος Πληροφορικής και Επικοινωνιών»,
«Ενίσχυση Σπουδών Πληροφορικής στο ΤΕΙ Σερρών» Μέτρο 2.2, Γ' ΚΠΣ

Γιατί είναι τόσο κακή;

- Γιατί οι ενέργειες που εκτελούνται μέσα στο συστατικό έχουν μόνο μικρή σχέση μεταξύ τους αλλά συνήθως μεγάλη σχέση με ενέργειες που γίνονται σε άλλα συστατικά.
  - Αρχικοποιούμε τον πίνακα Sales αλλά άλλες ενέργειες όπως «ενημέρωσε τον Sales» «τύπωσε τον Sales» βρίσκονται σε άλλα συστατικά
  - Αν θέλουμε να αλλάξουμε τη δομή του πίνακα σε ποιά συστατικά πρέπει να αλλάξουμε τον κώδικα;
- Δεν επιτρέπει την επαναχρησιμοποίηση

Διαδικασιακή συνεκτικότητα

- Διαφορετικές λειτουργίες ομαδοποιούνται σε ένα συστατικό επειδή πρέπει να εκτελεστούν με συγκεκριμένη σειρά
- Συνήθως οι λειτουργίες δεν έχουν κάποιο κοινό δεδομένο, αλλά απλά όταν ολοκληρώνεται η μία περνάει τον έλεγχο στην άλλη
  - Για παράδειγμα:
  - function makeRepairs() {

    read part number

    update repair record on master file

    }

Γιατί είναι κακή;

- Δεν επιτρέπει την επαναχρησιμοποίηση
- Οι ενέργειες μέσα στο συστατικό είναι μόνο χαλαρά συνδεδεμένες μεταξύ τους.

Πότε είναι καλή;

- Όταν φτιάχνεται ένα συστατικό που ελέγχει (βρίσκεται πάνω από) όλα τα άλλα

Επικοινωνιακή συνεκτικότητα

- Ομαδοποιούνται συναρτήσεις που δρουν στο ίδιο σύνολο δεδομένων. Μοιάζει με τη χρονική αλλά εδώ κάθε λειτουργία δρα στα ίδια δεδομένα και η σειρά με την οποία γίνονται οι λειτουργίες δεν είναι τόσο σημαντική
- Παράδειγμα
  - Ενημέρωσε μια εγγραφή στη βάση ΚΑΙ γράψε τα στοιχεία της εγγραφής σε ένα αρχείο
  - Υπολόγισε νέες συντεταγμένες ΚΑΙ τύπωσέ τες στην οθόνη

Γιατί δεν είναι καλή;

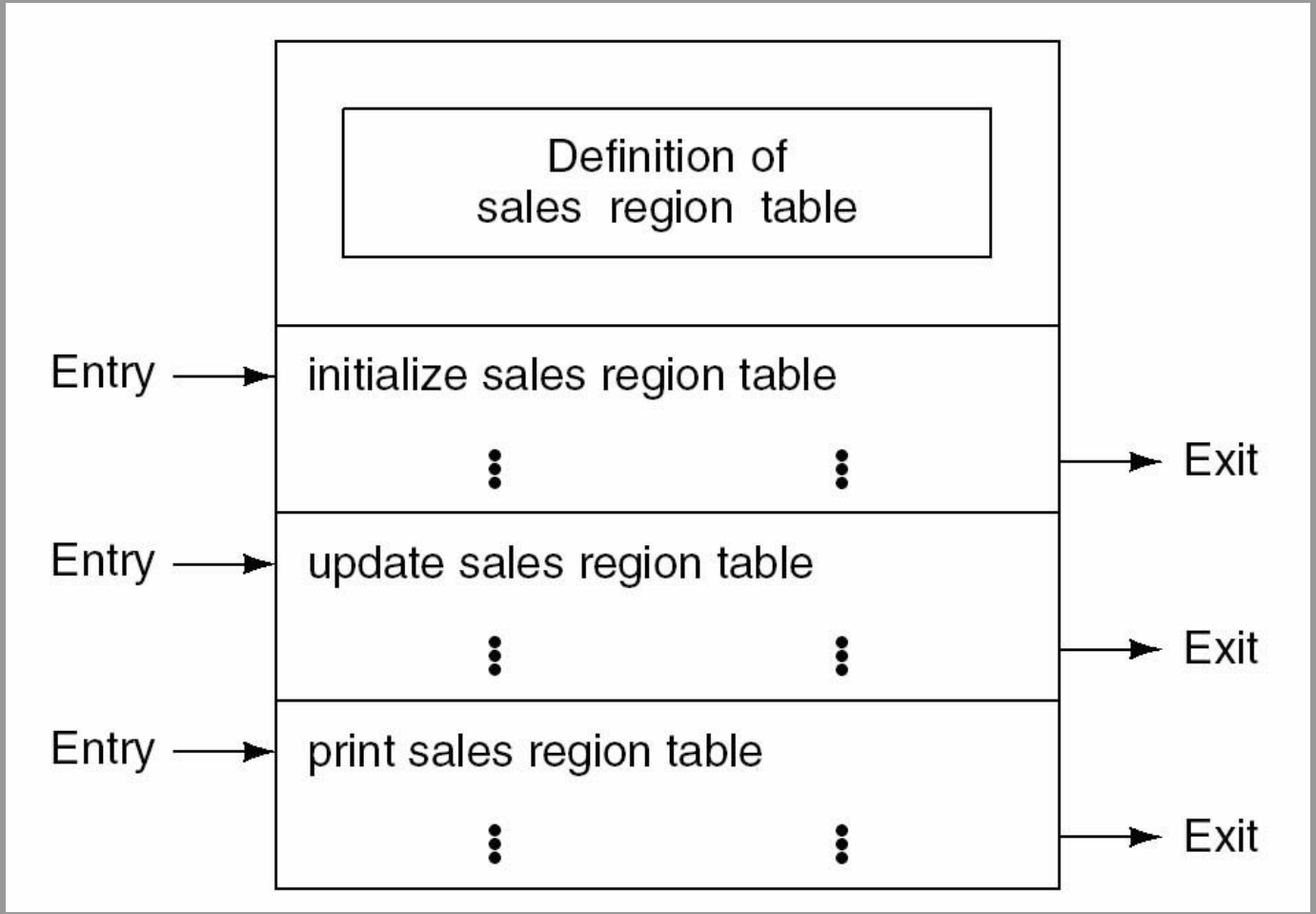
- Οι ενέργειες είναι πιο στενά συνδεδεμένες αλλά όχι με το βέλτιστο τρόπο
- Δεν είναι εύκολη η επαναχρησιμοποίηση γιατί κάθε συστατικό τελικά εκτελεί πάνω από μια ενέργειες
- Καλό θα ήταν να χωριστεί σε περισσότερα συστατικά, ένα για κάθε ενέργεια

Ακολουθιακή συνεκτικότητα

- Όταν ένα συστατικό εκτελεί μια σειρά από ενέργειες, αλλά κάθε μία είναι ορισμένη ανεξάρτητα από την άλλη και όλες ενεργούν πάνω στο ίδιο σύνολο δεδομένων

Υποέργο 1 «Αυτεπιστασία του τμήματος Πληροφορικής και Επικοινωνιών»,
«Ενίσχυση Σπουδών Πληροφορικής στο ΤΕΙ Σερρών» Μέτρο 2.2, Γ' ΚΠΣ

Παράδειγμα


Definition of
sales region table
Entry
initialize sales region table
Exit
Entry
update sales region table
Exit
Entry
print sales region table
Exit

Παράδειγμα (σε κώδικα C)

- Ένα αρχείο `salesRegion.c` που περιέχει

```
struct SalesRegion {}; //definition
initSalesRegion() //function
updateSalesRegion() //function
printSalesRegion() //function
```

Γιατί είναι καλή

- Γιατί ο κώδικας για κάθε ενέργεια είναι ανεξάρτητος και η επαναχρησιμοποίηση είναι πιο εύκολη

Λειτουργική συνεκτικότητα

- Κάθε τμήμα επεξεργασίας είναι απαραίτητο για την απόδοση μιας μόνο λειτουργίας
  - Παράδειγμα
    - Υπολόγισε τη θερμοκρασία του δωματίου
  - Παράδειγμα
    - Υπολόγισε το εμβαδόν ενός τετραγώνου

Υποέργο 1 «Αυτεπιστασία του τμήματος Πληροφορικής και Επικοινωνιών»,
«Ενίσχυση Σπουδών Πληροφορικής στο ΤΕΙ Σερρών» Μέτρο 2.2, Γ' ΚΠΣ

Γιατί είναι καλή

- Γιατί η επαναχρησιμοποίηση είναι εύκολη
- Γιατί η εύρεση και διόρθωση λαθών είναι εύκολη
- Η λειτουργική συνεκτικότητα είναι ιδεατή και όχι πάντα δυνατή