ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

Αντικειμενοστραφής Προγραμματισμός

Ενότητα 5: Συγκρίσεις και Αποφάσεις

Χ. ΑΓΓΕΛΗ

Τμήμα Ηλεκτρολόγων και Ηλεκτρονικών Μηχανικών

Μαθησιακά αποτελέσματα

- Σχεδιασμός της λογικής πίσω από τη λήψη αποφάσεων
- Λήψη αποφάσεων με τις εντολές if και if…else
- Χρήση πολλαπλών εντολών if και if…else
- Ένθεση εντολών if και if…else μέσα σε άλλες εντολές if και if…else
- Χρήση τελεστών AND και OR

Μαθησιακά αποτελέσματα (συνέχεια)

- Λήψη σαφών και αποδοτικών αποφάσεων
- Χρήση της εντολής switch
- Χρήση των τελεστών ελέγχου συνθηκών και της NOT
- Αποτίμηση προτεραιότητας τελεστών
- Προσθήκη αποφάσεων και μεθόδων κατασκευής σε μεθόδους στιγμιοτύπων

Σχεδιασμός της λογικής πίσω από τη λήψη αποφάσεων

- **Ψευδοκώδικας**
  - Χρησιμοποιήστε χαρτί και μολύβι
  - Σχεδιάστε τη λογική ενός προγράμματος σε απλή γλώσσα
  - Εκτελέστε σημαντικά βήματα μιας δεδομένης εργασίας
  - Χρησιμοποιήστε καθημερινή γλώσσα
- **Διάγραμμα ροής**
  - Βήματα σε μορφή διαγράμματος
  - Μια σειρά σχημάτων που συνδέονται με βέλη

Σχεδιασμός της λογικής πίσω από τη λήψη αποφάσεων (συνέχεια)

- **Διάγραμμα ροής** (συνέχεια)
  - Συνδυασμός σχημάτων για να αναπαραστηθούν διαφορετικές λειτουργίες
    - Τα ορθογώνια αναπαριστούν οποιοδήποτε βήμα που δεν εξαρτάται από κάποια συνθήκη
    - Οι ρόμβοι αναπαριστούν οποιαδήποτε απόφαση
- **Δομή ακολουθίας**
  - Ένα βήμα ακολουθεί ένα άλλο χωρίς καμία συνθήκη
  - Δεν επιτρέπεται διακλάδωση ή παράκαμψη ενός βήματος

Σχεδιασμός της λογικής πίσω από τη λήψη αποφάσεων (συνέχεια)

- **Δομή απόφασης**
  - Επιτρέπει την επιλογή μεταξύ εναλλακτικών διαδρομών εκτέλεσης εργασιών
  - Βασίζεται σε κάποια τιμή μέσα σε ένα πρόγραμμα
- **Λογικές τιμές**
  - Τιμές `true` και `false`
  - Χρησιμοποιούνται σε κάθε απόφαση υπολογιστών

Λήψη αποφάσεων με τις εντολές `if` και `if…else`

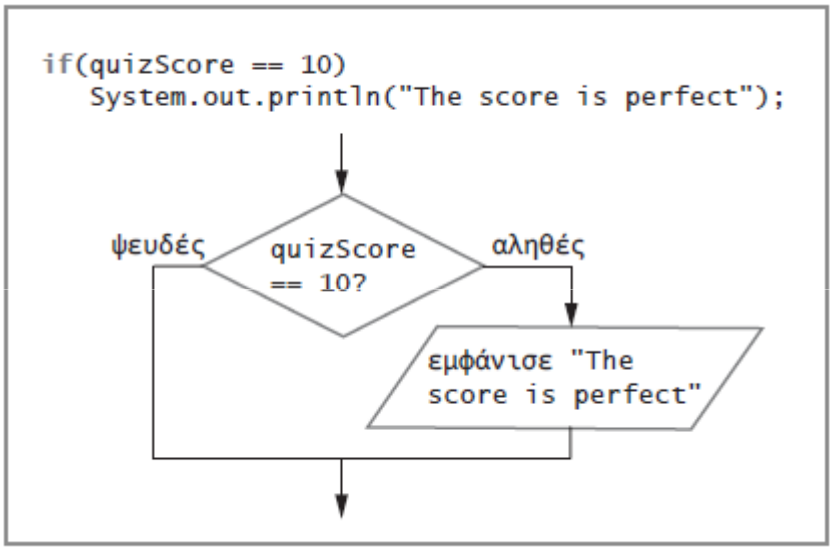
- **Εντολή `if`**
  - Η απλούστερη εντολή που μπορείτε να χρησιμοποιήσετε για να λάβετε μια απόφαση
  - Μια λογική έκφραση εμφανίζεται μέσα σε παρενθέσεις
  - Δεν υπάρχει κενό μεταξύ του `if` και την παρένθεση ανοίγματος
  - Η εκτέλεση πάντα συνεχίζεται στην επόμενη ανεξάρτητη εντολή
  - Χρησιμοποιείτε το διπλό ίσον (`==`) για έλεγχο της ισότητας

Λήψη αποφάσεων με τις εντολές if και if...else

1η Σύνταξη:

```
if (condition) {
  // block of code to be executed
if the condition is true
}
```

Λήψη αποφάσεων με τις εντολές `if` και `if...else` (συνέχεια)

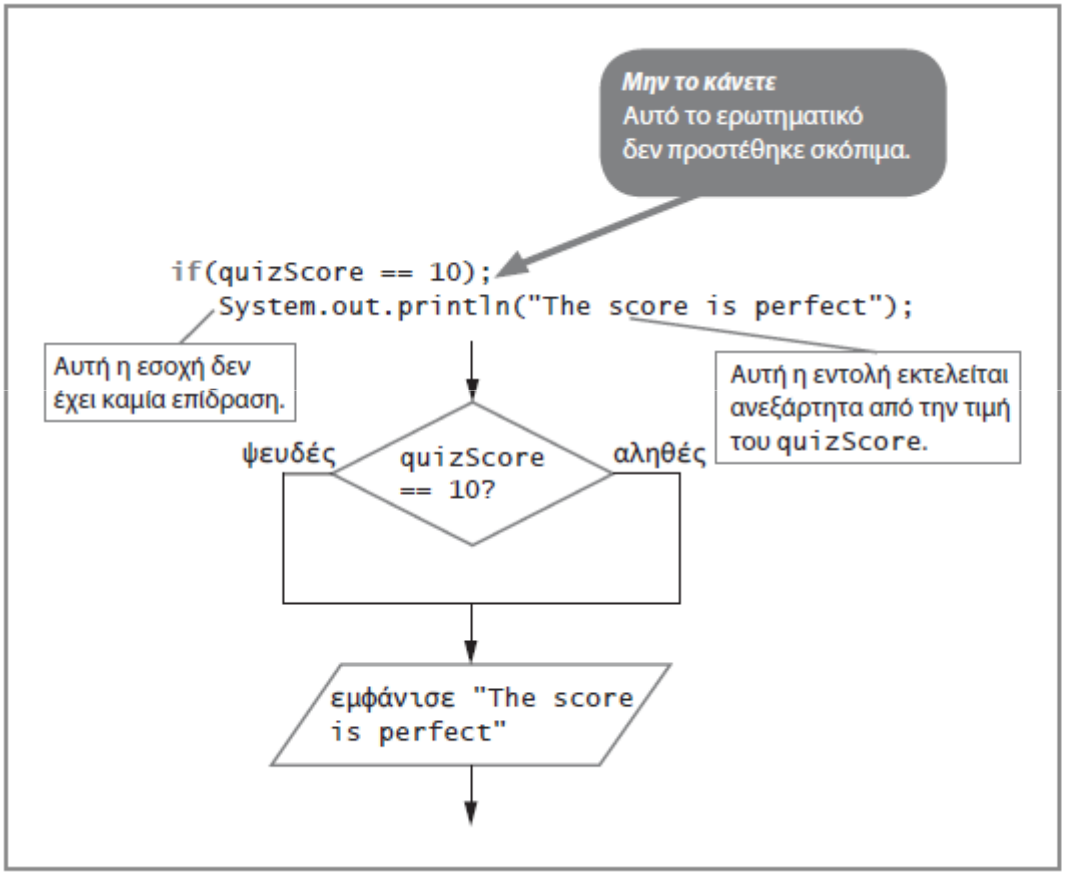


Εικόνα 5-3
Μια εντολή if της Java και η λογική της

Παγίδα: Μην τοποθετείτε σε λάθος σημείο το ερωτηματικό σε εντολές `if`

- Δεν πρέπει να υπάρχει ερωτηματικό στο τέλος της πρώτης γραμμής της εντολής `if`
  - `if(someVariable == 10)`
  - Η εντολή δεν σταματά εκεί
- Όταν ένα ερωτηματικό ακολουθεί αμέσως μετά το `if`:
  - Μια **κενή εντολή** περιέχει μόνο ένα ερωτηματικό
  - Η εκτέλεση συνεχίζεται με την επόμενη ανεξάρτητη εντολή

Παγίδα: Μην τοποθετείτε σε λάθος σημείο το ερωτηματικό σε εντολές `if` (συνέχεια)



Εικόνα 5-4
Λογική προγράμματος που εκτελείται όταν σε εντολή if εισαχθεί ένα επιπλέον ερωτηματικό

Παγίδα: Μην επιχειρείτε να συγκρίνετε αντικείμενα χρησιμοποιώντας τους σχεσιακούς τελεστές

- Χρησιμοποιείτε τους τυπικούς σχεσιακούς τελεστές για να συγκρίνετε τις τιμές των βασικών τύπων δεδομένων
  - Χωρίς αντικείμενα
- Δεν μπορείτε να χρησιμοποιείτε τις συγκρίσεις ισότητας και ανισότητας (== και !=) με αντικείμενα
  - Συγκρίνετε διευθύνσεις μνήμης αντικειμένων αντί για τιμές

Η εντολή `if…else`

- 2η Σύνταξη :

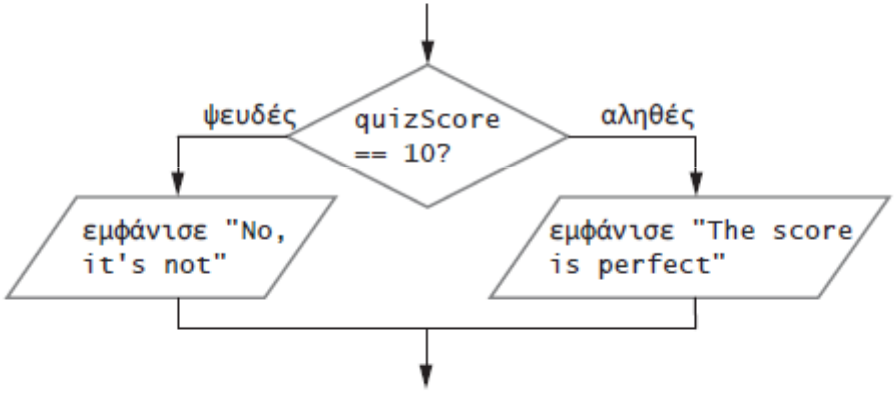
```
if (condition) {
    // block of code to be executed if the
  condition is true
  } else {
    // block of code to be executed if the
  condition is false
  }
```

Η εντολή `if…else` (συνέχεια)

- **Εντολή `if…else`** (συνέχεια)
  - Μια εντολή που εκτελείται όταν το `if` είναι `true` ή `false` και τελειώνει με ερωτηματικό
  - Κάθετη στοίχιση της εντολής `if` με την εντολή `else`
  - Δεν είναι έγκυρη η χρήση του `else` χωρίς το `if`
  - Ανάλογα με την αποτίμηση της λογικής έκφρασης μετά το `if`, μόνο μία ενέργεια λαμβάνει χώρα

Η εντολή `if...else` (συνέχεια)

```
if(quizScore == 10)
   System.out.println("The score is perfect");
else
   System.out.println("No, it's not");
```



Εικόνα 5-5
Μια εντολή if...else και η λογική της

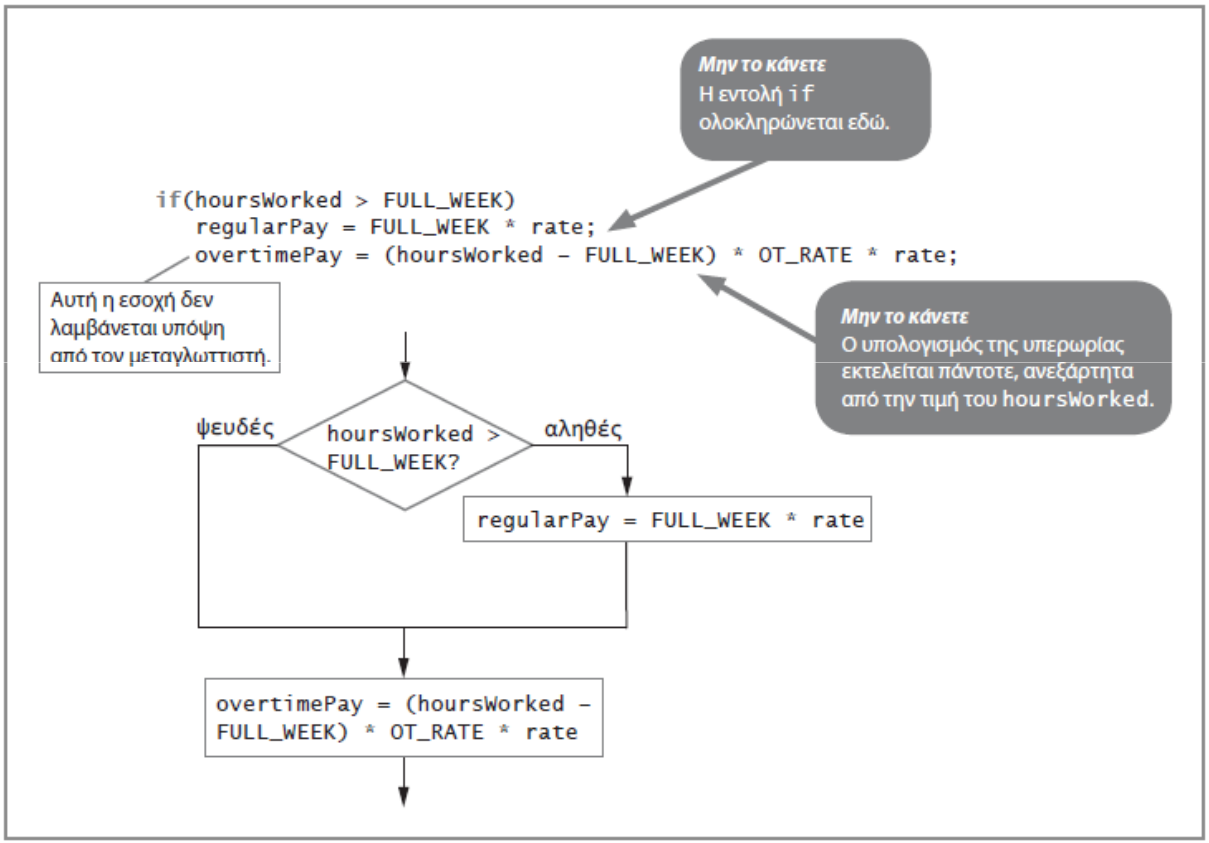
Χρήση πολλαπλών εντολών `if` και `if…else`

- Για εκτέλεση περισσότερων από μίας εντολών, χρησιμοποιείτε ένα ζεύγος αγκίστρων
  - Τοποθετείτε εξαρτημένες εντολές μέσα σε μια ενότητα
  - Είναι κρίσιμη η ορθή τοποθέτηση των αγκίστρων
- Οποιαδήποτε μεταβλητή που δηλώνεται μέσα σε μια ενότητα είναι τοπική για τη συγκεκριμένη ενότητα

- 3η Σύνταξη

```
if (condition1) {
    // block of code to be executed if
  condition1 is true
  } else if (condition2) {
    // block of code to be executed if the
  condition1 is false and condition2 is true
  } else {
    // block of code to be executed if the
  condition1 is false and condition2 is
  false
  }
```

Χρήση πολλαπλών εντολών `if` και `if...else` (συνέχεια)



Εικόνα 5-8

Εσφαλμένος υπολογισμός υπερωριακής αμοιβής όταν λείπουν τα άγκιστρα

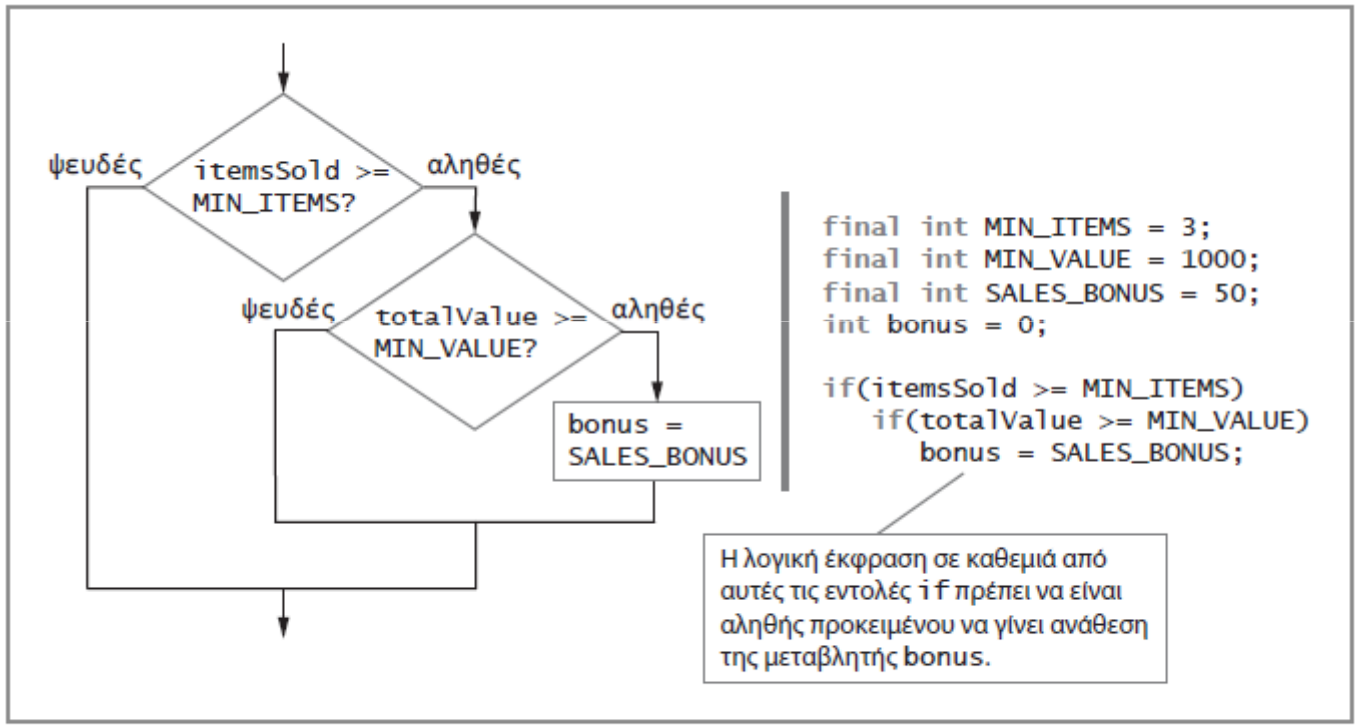
Χρήση εντολών `if` και `if…else`

```
import java.util.Scanner;
public class Payroll
{
    public static void main(String[] args)
    {
        double rate;
        double hoursWorked;
        double regularPay;
        double overtimePay;
        final int FULL_WEEK = 40;
        final double OT_RATE = 1.5;
        Scanner keyboard = new Scanner(System.in);
        System.out.print("How many hours did you work this week? ");
        hoursWorked = keyboard.nextDouble();
        System.out.print("What is your regular pay rate? ");
        rate = keyboard.nextDouble();
        if(hoursWorked > FULL_WEEK)
        {
            regularPay = FULL_WEEK * rate;
            overtimePay = (hoursWorked - FULL_WEEK) * OT_RATE * rate;
        }
        else
        {
            regularPay = hoursWorked * rate;
            overtimePay = 0.0;
        }
        System.out.println("Regular pay is " +
            regularPay + "\nOvertime pay is " + overtimePay);
    }
```

Ένθεση εντολών `if` και `if…else`

- **Ένθετες εντολές `if`**
  - Εντολές στις οποίες μια δομή `if` περιέχεται μέσα σε άλλη δομή `if`
  - Τουλάχιστον δύο συνθήκες πρέπει να πληρούνται πριν εκτελεστεί κάποια ενέργεια
- Προσέχετε ιδιαίτερα την τοποθέτηση των εντολών `else`
- Οι εντολές `else` σχετίζονται πάντα με το `if` ακολουθώντας μια πορεία “first in-last out” (το πρώτο που εισέρχεται είναι το τελευταίο που εξέρχεται)

Ένθεση εντολών `if` και `if...else` (συνέχεια)



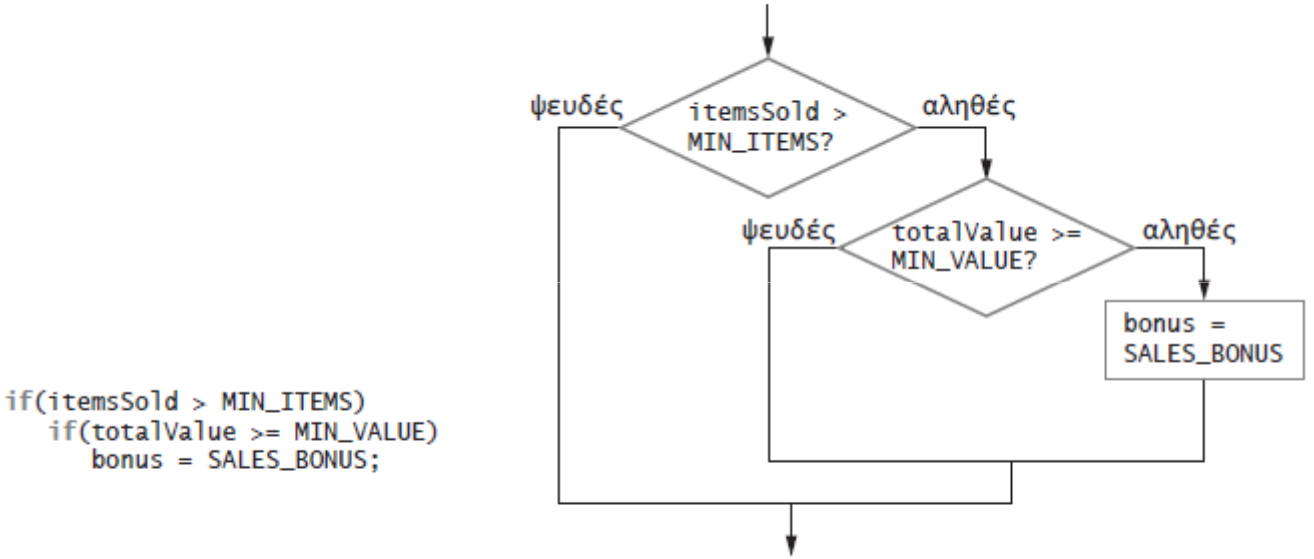
Εικόνα 5-12

Λήψη απόφασης, με τη βοήθεια ένθετων εντολών if, σχετικά με το αν θα δοθεί μπόνους ή όχι

Χρήση τελεστών AND και OR

- Ο **λογικής τελεστής AND**
  - Εναλλακτική για ορισμένες ένθετες εντολές `if`
  - Χρησιμοποιείται ανάμεσα σε δύο λογικές εκφράσεις για να διαπιστωθεί αν αμφότερες είναι `true`
  - Γράφεται με δύο σύμβολα & (`&&`)
    - Συμπεριλαμβάνεται μια πλήρης λογική έκφραση σε κάθε πλευρά
  - Αμφότερες οι λογικές εκφράσεις που περιβάλλουν τον τελεστή πρέπει να είναι αληθείς προτού πραγματοποιηθεί η ενέργεια στην εντολή

Χρήση τελεστών AND και OR (συνέχεια)




```
if(itemsSold > MIN_ITEMS)
   if(totalValue >= MIN_VALUE)
      bonus = SALES_BONUS;
```

```
if(itemsSold > MIN_ITEMS && totalValue >= MIN_VALUE)
   bonus = SALES_BONUS;
```

Εικόνα 5-15

Κώδικας και λογική για την απόφαση χορήγησης μπόνους, που χρησιμοποιούν ένθετα if και τον τελεστή &&

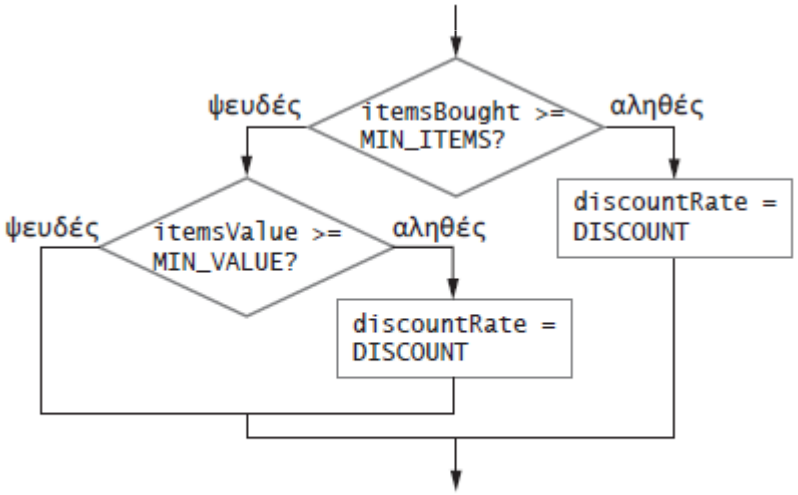
Χρήση τελεστών AND και OR (συνέχεια)

- Ο **τελεστής OR**
  - Μια ενέργεια που πρέπει να συμβεί αν μόνο μία από τις δύο συνθήκες είναι αληθής
  - Γράφεται ως ||
    - Αναφέρονται μερικές φορές ως μπάρες

Χρήση τελεστών AND και OR (συνέχεια)

```
if(itemsBought >= MIN_ITEMS)
   discountRate = DISCOUNT;
else
    if(itemsValue >= MIN_VALUE)
       discountRate = DISCOUNT;

if(itemsBought >= MIN_ITEMS || itemsValue >= MIN_VALUE)
   discountRate = DISCOUNT;
```

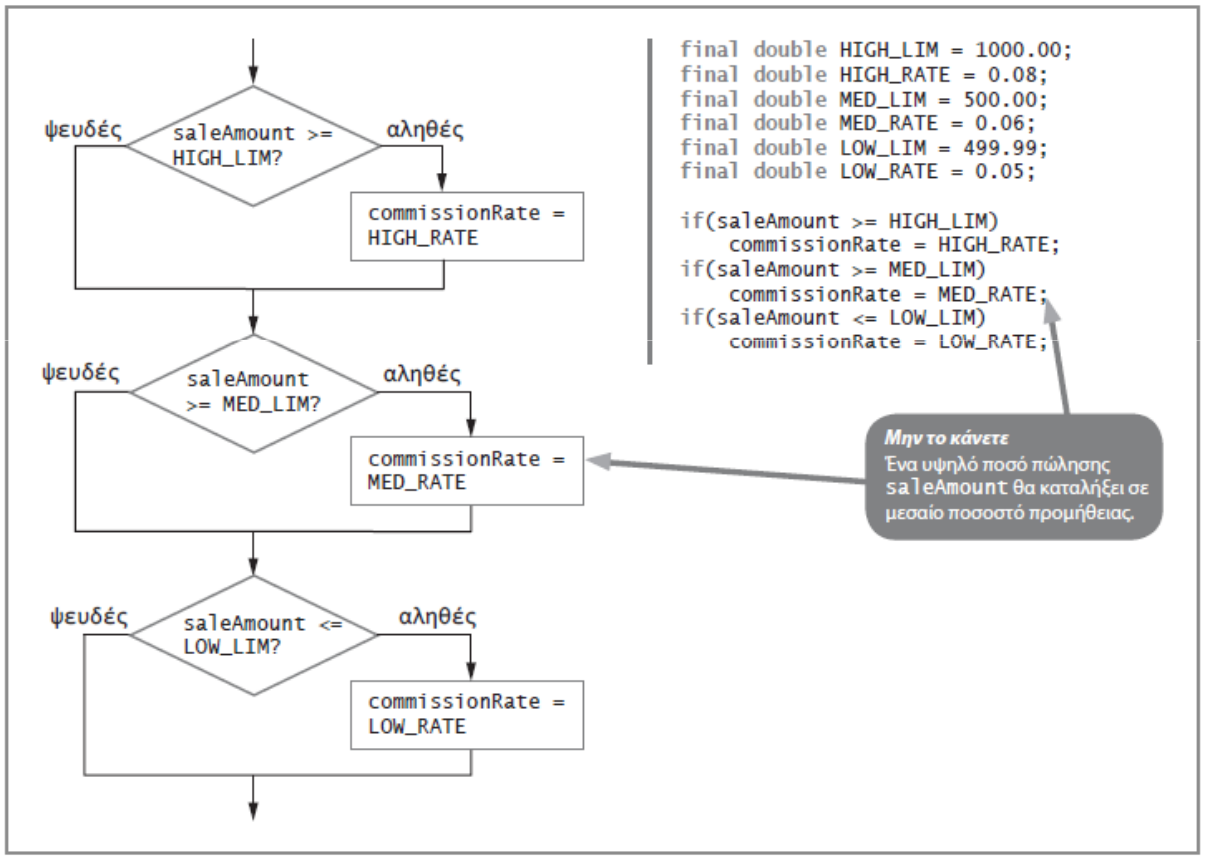


Εικόνα 5-16
Η απόφαση για έκπτωση σε έναν πελάτη, όταν ο πελάτης χρειάζεται να ικανοποιήσει μόνο ένα από δύο καθορισμένα κριτήρια

Λήψη σαφών και αποδοτικών αποφάσεων

- Πραγματοποίηση σαφών ελέγχων διαστήματος
  - **Έλεγχος διαστήματος**: Μια σειρά από εντολές που καθορίζουν το εύρος των διαδοχικών τιμών στο οποίο μπορεί να ανήκει μια άλλη τιμή
  - Οι προγραμματιστές Java συνήθως τοποθετούν κάθε `else` ενός διαδοχικού `if` στην ίδια γραμμή
  - Μέσα σε μια ένθετη εντολή `if…else`:
    - Είναι πιο αποδοτικό να τίθεται πρώτα το πιο πιθανό ερώτημα
    - Αποφεύγετε τα πολλά ερωτήματα

Λήψη σαφών και αποδοτικών αποφάσεων (συνέχεια)

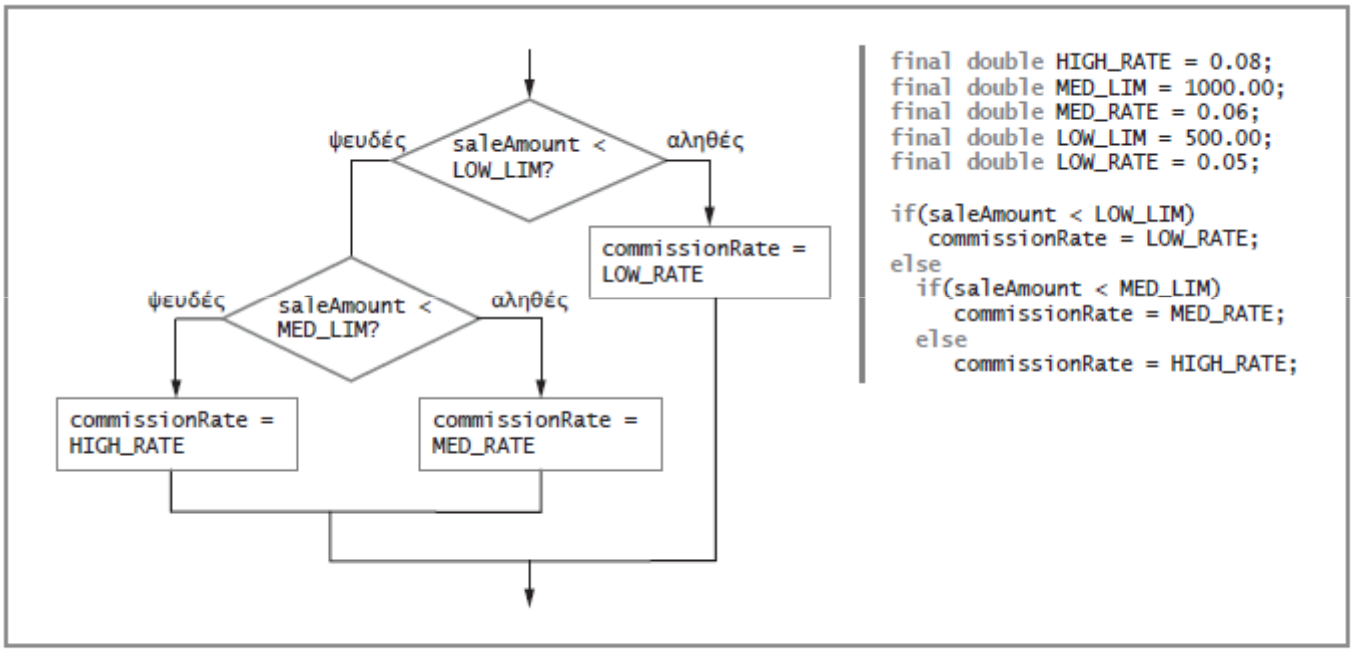


Εικόνα 5-19
Εσφαλμένος κώδικας ορισμού προμήθειας και η λογική του

Λήψη σαφών και αποδοτικών αποφάσεων (συνέχεια)

- Είναι πιο αποδοτικό να τίθεται πρώτα ένα ερώτημα που μάλλον θα είναι `true`
  - Αποφύγετε τα πολλά ερωτήματα
  - Γίνεται πιο αποδοτική η λήψη μιας σειράς αποφάσεων

Λήψη σαφών και αποδοτικών αποφάσεων (συνέχεια)




```
final double HIGH_RATE = 0.08;
final double MED_LIM = 1000.00;
final double MED_RATE = 0.06;
final double LOW_LIM = 500.00;
final double LOW_RATE = 0.05;

if(saleAmount < LOW_LIM)
   commissionRate = LOW_RATE;
else
  if(saleAmount < MED_LIM)
     commissionRate = MED_RATE;
  else
     commissionRate = HIGH_RATE;
```

Εικόνα 5-22

Κώδικας ορισμού προμήθειας και η λογική του, η οποία αποτιμά πρώτα το μικρότερο saleAmount

Κατάλληλη χρήση των && και ||

- Σφάλματα αρχάριων προγραμματιστών:
  - Χρήση του τελεστή AND όταν θέλουν να χρησιμοποιήσουν το OR
    - Παράδειγμα: καμία τιμή `payRate` δεν μπορεί να είναι ποτέ μικρότερη από 5,65 και μεγαλύτερη από το 60 ταυτόχρονα

```
if(payRate < LOW && payRate > HIGH)
    System.out.println("Error in pay rate");
```

    - Χρησιμοποιείτε τον τελεστή "||" αντίθετα
  - Χρησιμοποιείτε ένα μόνο & ή μία μόνο | για να δηλώσετε λογικό AND ή OR

Χρήση της εντολής `switch`

- Εντολή `switch`
  - Εναλλακτική λύση αντί της χρήσης μιας ακολουθίας ένθετων εντολών `if`
  - Έλεγχος μίας μεμονωμένης μεταβλητής ως προς κάποια σειρά συγκεκριμένων ακεραίων, χαρακτήρων ή συμβολοσειρών

Χρήση της εντολής `switch` (συνέχεια)

- Δεσμευμένες λέξεις
  - `switch`
    - Ξεκινά τη δομή της εντολής
    - Ακολουθείται από μια έκφραση ελέγχου μέσα σε παρενθέσεις
  - `case`
    - Ακολουθείται από μία από τις πιθανές περιπτώσεις τιμών προς έλεγχο και μια άνω και κάτω τελεία

Χρήση της εντολής `switch` (συνέχεια)

- Δεσμευμένες λέξεις (συνέχεια)
  - `break`
    - Τερματίζει προαιρετικά μια εντολή `switch` στο τέλος κάθε περίπτωσης
  - `default`
    - Χρησιμοποιείται προαιρετικά πριν από οποιαδήποτε ενέργεια που θα πρέπει να πραγματοποιηθεί σε περίπτωση που η μεταβλητή ελέγχου δεν αντιστοιχεί σε καμία από τις περιπτώσεις οι οποίες ελέγχονται

Χρήση της εντολής `switch` (συνέχεια)

```
switch(year)
{
    case 1:
        System.out.println("Freshman");
        break;
    case 2:
        System.out.println("Sophomore");
        break;
    case 3:
        System.out.println("Junior");
        break;
    case 4:
        System.out.println("Senior");
        break;
    default:
        System.out.println("Invalid year");
}
```

Εικόνα 5-24

Εύρεση του χαρακτηρισμού του χρόνου σπουδών με χρήση εντολής switch

Χρήση της εντολής `switch` (συνέχεια)

```
int department;
String supervisor;
// Οι εντολές για την εύρεση του τμήματος τοποθετούνται εδώ
switch(department)
{
    case 1:
    case 2:
    case 3:
        supervisor = "Jones";
        break;
    case 4:
        supervisor = "Staples";
        break;
    case 5:
        supervisor = "Tejano";
        break;
    default:
        System.out.println("Invalid department code");
}
```

Χρήση της εντολής `switch` (συνέχεια)

- Εντολές `break` στη δομή `switch`
  - Αν παραλείπεται μια εντολή `break`:
    - Το πρόγραμμα βρίσκει μια αντιστοιχία για τη μεταβλητή ελέγχου
    - Όλες οι εντολές μέσα στην εντολή `switch` εκτελούνται απ' αυτό το σημείο και μετά
- Εντολή `case`
  - Δεν είναι υποχρεωτικό να γράφετε κώδικα για κάθε περίπτωση
  - Αποτιμάτε μεταβλητές `char`
    - Παραβλέπετε αν χρησιμοποιούνται πεζοί ή κεφαλαίοι χαρακτήρες

Χρήση της εντολής `switch` (συνέχεια)

- Γιατί χρησιμοποιούμε εντολές `switch`;
  - Προσφέρονται όταν πολλές εναλλακτικές σειρές ενεργειών εξαρτώνται από έναν μεμονωμένο ακέραιο, χαρακτήρα ή συμβολοσειρά
  - Συνιστώνται μόνο όταν υπάρχει εύλογος αριθμός συγκεκριμένων αντιστοιχιών που πρέπει να ελεγχθούν

Χρήση των τελεστών ελέγχου συνθηκών και της NOT

- **Τελεστής συνθήκης**
  - απαιτεί τρεις εκφράσεις, οι οποίες διαχωρίζονται με ένα αγγλικό ερωτηματικό και μια άνω και κάτω τελεία
  - Χρησιμοποιείται ως σύντομη εκδοχή της εντολής `if…else`
  - Δεν είστε υποχρεωμένοι σε καμία περίπτωση να τον χρησιμοποιήσετε
- Σύνταξη τελεστή συνθήκης:

```
testExpression ? trueResult :
falseResult;
```

Χρήση των τελεστών ελέγχου συνθηκών και της NOT (συνέχεια)

- Μια λογική έκφραση αποτιμάται ως `true` ή `false`
  - Αν η τιμή του `testExpression` είναι `true`:
    - Ολόκληρη η έκφραση συνθήκης αποκτά την τιμή της έκφρασης που ακολουθεί το αγγλικό ερωτηματικό
  - Αν η τιμή είναι `false`:
    - Ολόκληρη η έκφραση αποκτά την τιμή της έκφρασης `falseResult`
- Ένα πλεονέκτημα από τη χρήση του τελεστή συνθήκης είναι η συμπυκνωμένη γραφή της εντολής

Χρήση του τελεστή NOT

- **Τελεστής NOT**
  - Συμβολίζεται με το θαυμαστικό (`!`)
  - Δίνει την άρνηση του αποτελέσματος οποιασδήποτε λογικής έκφρασης
  - Όταν ακολουθεί τον τελεστή NOT, οποιαδήποτε έκφραση που αποτιμάται ως:
    - `true` γίνεται `false`
    - `false` γίνεται `true`
- Εντολές με τον τελεστή NOT:
  - Δεν διαβάζονται πολύ εύκολα
  - Απαιτούν δύο ζεύγη παρενθέσεων

Αποτίμηση προτεραιότητας τελεστών

- Συνδυάζετε όσους τελεστές AND ή OR χρειάζεστε
- Προτεραιότητα ενός τελεστή
  - Πώς αποτιμάται μια έκφραση
  - Η σειρά συμφωνεί με την κοινή αλγεβρική χρήση
    - Πρώτα γίνονται οι αριθμητικές πράξεις
    - Στο τέλος γίνεται η εκχώρηση
    - Ο τελεστής AND αποτιμάται πριν τον τελεστή OR
    - Οι εντολές μέσα σε παρενθέσεις αποτιμώνται πρώτα

Αποτίμηση προτεραιότητας τελεστών (συνέχεια)

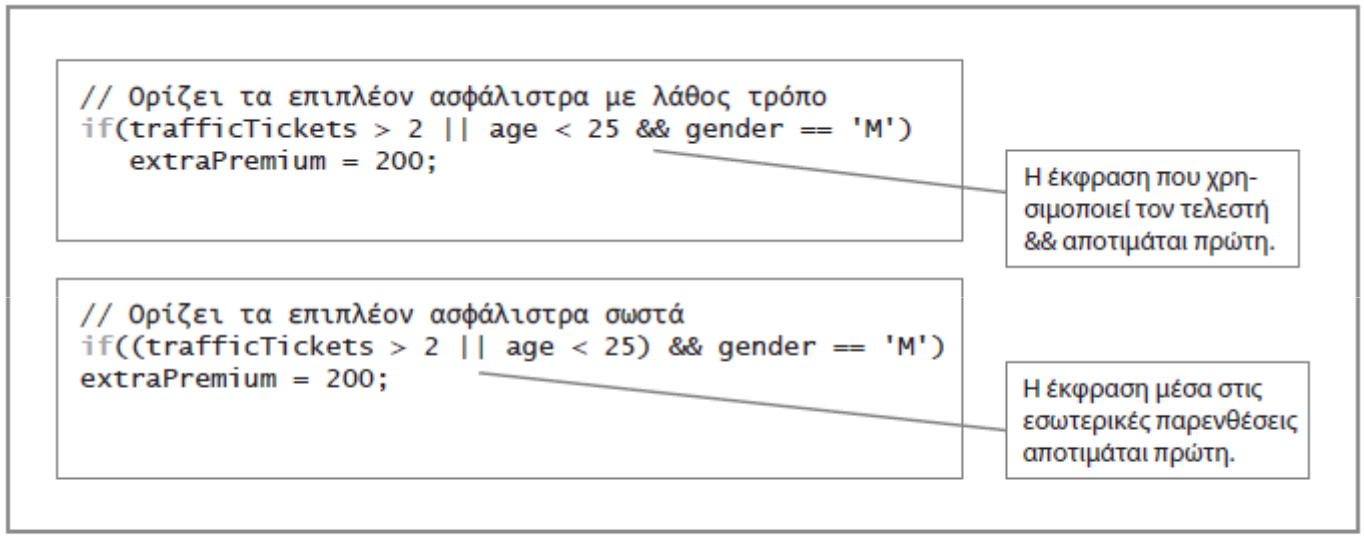
Προτεραιότητα	Τελεστής (τελεστές)	Σύμβολο (σύμβολα)
Υψηλότερη	Λογικό NOT	!
Ενδιάμεση	Πολλαπλασιασμός, διαίρεση, modulus	* / %
	Πρόσθεση, αφαίρεση	+ -
	Σχεσιακός	> < >= <=
	Ισότητα	== !=
	Λογικό AND	&&
	Λογικό OR	||
	Συνθήκη	?:
Χαμηλότερη	Ανάθεση	=

Πίνακας 5-1 Προτεραιότητα τελεστών για τελεστές που έχετε χρησιμοποιήσει ήδη

Αποτίμηση προτεραιότητας τελεστών (συνέχεια)

- Δύο σημαντικές συμβάσεις
  - Η σειρά με την οποία χρησιμοποιούνται οι τελεστές έχει σημασία
  - Πάντα χρησιμοποιείτε παρενθέσεις όταν θέλετε να αλλάξετε την προτεραιότητα ή να αποσαφηνίσετε τις προθέσεις σας

Αποτίμηση προτεραιότητας τελεστών (συνέχεια)



Εικόνα 5-31

Δύο συγκρίσεις που χρησιμοποιούν τα && και ||

Προσθήκη αποφάσεων σε μεθόδους κατασκευής αντικειμένων

- Εξασφαλίζει ότι τα πεδία έχουν αποδεκτές τιμές
- Προσδιορίζει αν οι τιμές βρίσκονται εντός των επιτρεπόμενων ορίων για τα πεδία

Προσθήκη αποφάσεων σε μεθόδους κατασκευής αντικειμένων (συνέχεια)

```
public class Employee
{
    private int empNum;
    private double payRate;
    public int MAX_EMP_NUM = 9999;
    public double MAX_RATE = 60.00;
    Employee(int num, double rate)
    {
        if(num <= MAX_EMP_NUM)
            empNum = num;
        else
            empNum = MAX_EMP_NUM;
        if(payRate <= MAX_RATE)
            payRate = rate;
        else
            payRate = 0;
    }
    public int getEmpNum()
    {
        return empNum;
    }
    public double getPayRate()
    {
        return payRate;
    }
}
```

Εικόνα 5-32
Η κλάση Employee που περιέχει μέθοδο κατασκευής η οποία λαμβάνει αποφάσεις