Fundamentals of Programming

Laboratory 3

Selection structures, designing selections, nested selection

DECISION STRUCTURES

- allow programs to follow different courses of action or execute different tasks depending on the data values

CONDITIONALS

- cause something to happen in a program only if a specific condition is met
- **If** a condition is true **then** *do something*, **else** *do something else*
- usually a relational operator is used to compare the two properties

RELATIONAL OPERATORS

<	less than
>	greater than
==	equal to
<=	less than or equal to
>=	greater than or equal to
!=	not equal to

IF STATEMENTS

- The most basic way to test a condition in a Java program
- works along the same lines, testing to see whether a condition is true or false, and taking action only if the condition is true
- boolean variable type is used to store only two possible values: *true* or *false*

IF STATEMENTS

Single - option structure:

```
if (expression)
{
    statement(s);
}
```

IF STATEMENTS

Example:

```
int a = 1;
if(a==1)
{
  System.out.print(”a equals 1”);
}
/*The body of an if statement
(printing “a equals to 1”) will be
run only if the condition is true
(a is equal to 1). */
```

IF – ELSE STATEMENTS

- used when you want to do something if a condition is true and do something else if the condition is false

IF – ELSE STATEMENTS

Double - option structure:

```
if (expression)
{
  // code to perform if true
}
else
{
  // code to perform if false
}
```

IF – ELSE STATEMENTS

Example:

```
int a = 1;
if (a == 1)
{
   System.out.print("a equals 1");
}
else
{
   System.out.print("a is not equal to 1");
}
```

IF – ELSE STATEMENTS

Multi – alternative decision:

We can put together several different if and else statements to handle a variety of conditions

IF – ELSE IF - STATEMENTS

```
if (b==0)
{
  System.out.print("b is 0");
}
else if (b==1)
  System.out.print("b is 1");
else if (b=>2)
{
  System.out.print("b is greater than 2");
}
else
{
  System.out.print(“b value is negative”);
}
```

SMALL REVIEW

What's wrong with the code below?

```
if (i = 1)
{
/* do something */
}
```

ORAL
EXERCISE

SWITCH STATEMENTS

Another way to create in Java multi – alternative decision is to use a **switch** structure, very useful when you wish to avoid a lot of IF statements

SWITCH STATEMENTS

```
switch (integerVariable)
{
   case 1:
      System.out.println(“1");
       //break;
   case 2:
       System.out.println(“2");
       break;
   case 3:
       System.out.println(“3");
       break;
   default: //optional
      System.out.println(„Wrong number!");
}
```

SWITCH STATEMENTS

```
switch (intVariable)
{
   case 1: //no break, so continues to next case
   case 2:
        System.out.println("1 or 2");
        break;
   case 3:
        System.out.println("3");
        break;
   default: //optional
      System.out.println(„Wrong number!");
}
```

SWITCH STATEMENTS

- The first line of the switch statement specifies the variable that will be tested.
- Then the switch statement uses the brace brackets to form a block statement.
- Each of the case statements checks the test variable in the switch statement against a specific value.
- It is **impossible** to test in switch structure other conditions than “= =”

SWITCH STATEMENTS

- break statement causes that no other part of the switch statement will be considered. The break statement tells the computer to break out of the switch statement.

- The default statement is used as a catch-all if **none** of the preceding case statements contained a **break**.

TERNARY OPERATOR

You can use the ternary operator when you just want to assign a value or display a value based on a simple conditional test.

```
if (expression)
        a = someValue;
else
        a = anotherValue;
```

Instead you can write:

```
a  = (expression) ? someValue : otherValue ;
```

TERNARY OPERATOR STRUCTURE

The condition to test, surrounded by parentheses

(expression)

followed by a question mark ?

The value use if the condition is true, then a colon :

The value to use if condition is false

```
variable  =  (expression)  ? someValue
: otherValue ;
```

TERNARY OPERATOR STRUCTURE

Example 1:

```
int result = 10;
int numberOfPoints;
numberOfPoints = (result > 5) ? 1
: 0;
```

If result value is greater than 5 number of given points is 1 else there is no given points.

TERNARY OPERATOR STRUCTURE

Example 2:

```
String gender = "female";
System.out.print(
(gender.equals("female")) ? "Ms." : "Mr." );

  /* text variables cannot be compared using ==,
  instead the .equals() method must be used */
```

NESTED DECISION STATEMENTS

Example [Sanchez]:

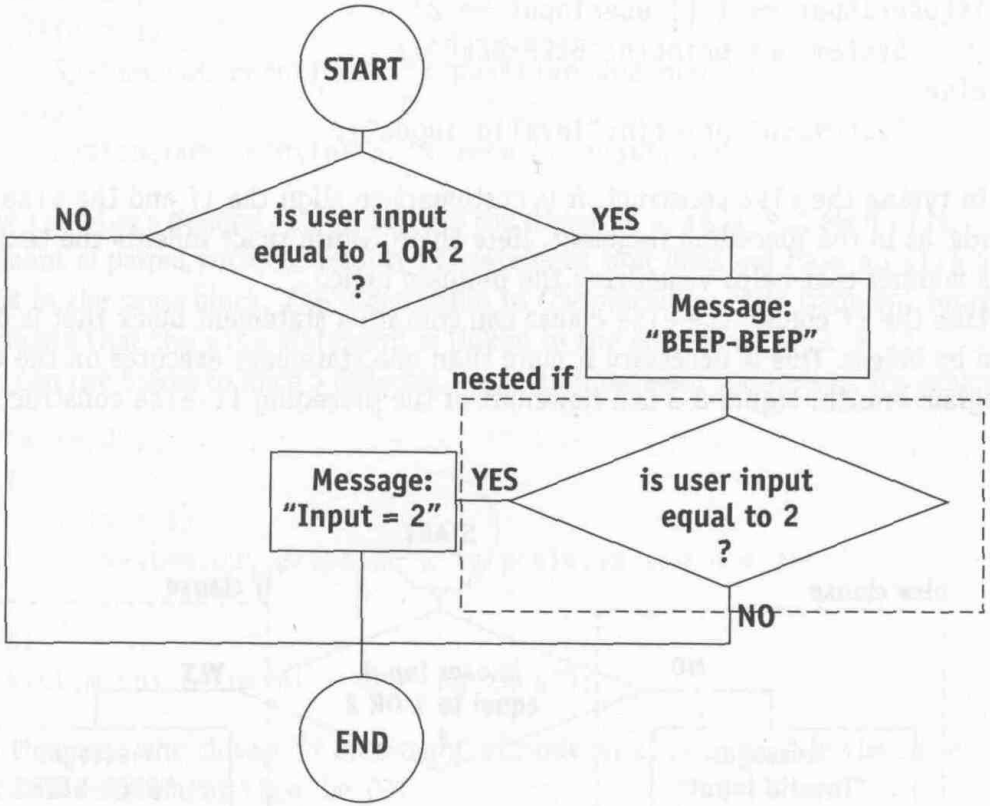


Figure 8-2
Flowchart of a nested if construct

Java Code:

```
Scanner input = new Scanner(System.in);
int userInputValue;
userInputValue = input.nextInt();
if ((userInputValue = = 1) || (userInputValue = = 2))
{
   System.out.println("BEEP - BEEP");
   if(userInputValue = = 2)
   {
        System.out.print("Input = 2");
   }
}
```

NOTE:
*|| **means OR,***
*&& **means AND.***

The Dangling else Problem

Because the `else` statement is optional, we can have several nested `if` constructs, some of which without a corresponding `else` clause. This situation, sometimes referred to as a *dangling else*, can give rise to uncertainty about the pairing of `if` and `else` clauses. For example, the `else` clause in the following fragment is paired with the inner `if` statement:

```
if(a != 0 )
   if(a > 1)
     System.out.println("x is positive and non-zero);
   else
     System.out.println("x is zero or negative);
```

The compiler's general rule in solving the dangling `else` is that each `else` statement is paired with the closest `if` statement that does not have an `else` and that is in the same block. The indentation in the preceding code fragment serves to indicate that the `else` statement is linked to the statement `if(a > 1)`.

We can use braces to force a different association between statements. For example:

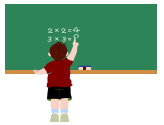
```
if(a != 0 )
  {
     if(a > 1)
        System.out.println("x is positive and non-zero);
  }
else
  System.out.println("x must be zero");
```

In this case, the closest `if` statement without an `else` located in the same block is the statement `if(a != 0)`.

Here again we have used indentation of the text lines to indicate the `if-else` pairing.

EXERCISE 1a

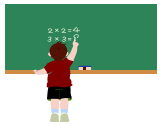
If the variable *angle* is equal to 90 degrees, print the message "*The angle is a right angle*"; otherwise, print the message "*The angle is not a right angle*".



BOARD
EXERCISE

EXERCISE 1b

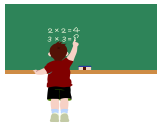
If the *temperature* is above 100 degrees, display the message "*above boiling*"; if it is equal to 100 display the message "*boiling";* otherwise display *"below boiling"*



BOARD EXERCISE

EXERCISE 1c

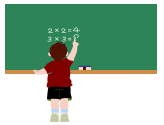
If the variable ***number*** is positive, add ***number*** to ***posSum*** *(*increment ***posSum*** by ***number****)*; otherwise, add ***number*** to ***negSum***



BOARD
EXERCISE

EXERCISE 1d

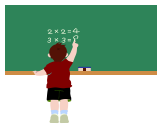
If the variable *slope* is less than 0.5 set the *flag* to zero; otherwise, set *flag* to one



BOARD
EXERCISE

EXERCISE 1e

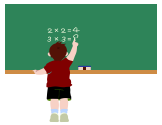
If the difference between the variables ***num1*** and ***num2*** is less than 0.001, set the variable ***approx*** to zero; otherwise, calculate ***approx*** as the quantity (***num1*** – ***num2***)/ *2.0*.



BOARD
EXERCISE

EXERCISE 1f

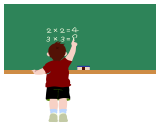
If the difference between the variables ***temp1*** and ***temp2*** exceeds 2.3 degrees, calculate the variable ***error*** as (***temp1*** – ***temp2***) * ***factor***



BOARD EXERCISE

EXERCISE 1g

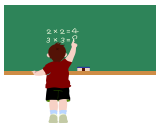
If the variable *x* is greater than the variable *y* and the variable *z* is less than 20, read in a value for the variable *p*;



BOARD
EXERCISE

EXERCISE 1h

If the variable *distance* is greater then 20 and it is less than 35, read in value for the variable *time*.



BOARD EXERCISE

EXERCISE 2

Write a Java program that displays either the message “I feel great today! Or “ I feel down today!” depending on the input. If the input is the integer 1, entered in the variable choice, the first message should be displayed; otherwise, the second message should be displayed

PROGRAMMING EXERCISE

EXERCISE 3

A senior salesperson is paid $400 a week and a junior salesperson $275 a week. Write a Java program that accepts as input a salesperson's status in a String variable. If the status equals "s", the senior person's salary should be displayed; otherwise, the junior person's salary should be output.

Note:

```
String status = input.nextLine();
if (status.equals("s"))
   …
```

PROGRAMMING EXERCISE

EXERCISE 4

Write a Java program that displays a student's status based on the following codes:

Code	Student Status
1.	Freshman
2.	Sophomore
3.	Junior
4.	Senior
5.	Master program
6.	Doctoral program

Your program should accept the code number as a user – entered input value and based on this value display the correct student status. If an incorrect code is entered, your program should display the string "An incorrect code was entered".

PROGRAMMING EXERCISE

EXERCISE 5

Each disk drive in a shipment of these devices is stamped with a code from 1 through 4, which indicates a drive manufacturer as follows

Code	Disk Drive Manufacturer
1.	3M Corporation
2.	Maxell Corporation
3.	Sony Corporation
4.	Verbatim Corporation

Write a Java program that accepts the code number as an input and, based on the value entered, displays the correct disk drive manufacturer.

Note: Use a different method than in Ex.4

PROGRAMMING EXERCISE

HOMEWORK (option 1)

Write a Java program that calculates the areas of different geometrical figures. Your program should display menu as below:

```
 ----------------------
| GEOMETRICAL FIGURE   |
|    CALCULATIONS      |
 ----------------------
| Figure:              |
| 1. Circle            |
| 2. Triangle          |
| 3. Ellipse           |
| 4. Trapezium         |
-----------------------
| Type number desired  |
|                      |
-----------------------
```

After choosing the option program should ask the user for the data required (height, width, radius etc.) to calculate the area. Program displays the result of figure field calculation based on entered data.

HOMEWORK
EXERCISE

HOMEWORK (option 2)

Write a Java program that works like a “Magic 8-ball” or “Futrzak”.

Your program should ask the user to type a question with a yes/no answer, then display a random answer, example:

```
Will I get a top mark in this class?
Not likely
```

Make sure to check if the user typed something (the string is not empty textvar.equals(""))

Hint:

Use Math.random() to generate a random number 0.0<=x<1.0 multiply the number by 10 and round it to int, then use switch structure.

HOMEWORK EXERCISE