Chapter3

Flow of control: Branching:

A branching statement chooses one action from a list of two or more possible actions. A loop statement repeats an action again and again until some stopping condition is met.

If-else offers a choice of two actions;

When your program executes an if-else statement, it first checks the expression in parentheses after the keyword if . This expression must be something that is either true or false. If it is true, the statement before the else is executed. If the expression is false, the statement after the else is executed.

Indentation:

If you want to include more than one statement in each branch, simply enclose the statements in braces{ }; Several statements enclosed within braces are considered to be one larger statement.

These statements formed by enclosing a list of statements within braces are called compound statements.

Every if-else statement is of the form:

If(Boolean\_Expression)

Statement\_1

Else( optional)

Statement\_2

If (Boolean\_Expression)

Statement

Use = instead of == to test for equality

The = is the assignment operator; use == to test equality;

Floating point number can be thought of as approximations. Numbers that have a fractional part can have an infinite number of digits. And floating-point numbers often are not exact. If you compute two floating-point values, they likely will not be exactly equal.

You can form more complicated Boolean expressions from simpler ones by joining them with the logical operator &&, which is “ and “ in Java .

When you form a larger Boolean expression by connecting two smaller expressions with &&, the entire larger expression is true only if both of the smaller expressions are true. If at least one of the smaller expressions is false , the larger expression is false;

Instead of joining Boolean expressions with and, you can join them with “or” ||

(Expression\_1) || ( Expression\_2)

This expression is true if either Expression\_1 or Expression\_2 is true of both are true;

Use ! for not;

You can use ! to negate the value of a Boolean expression ! Boolean\_expression

Example

If( ! ( number <0 )

System.out.println(“OK”);

Else

System.out.println(“ Negative!”);

Compare Strings

To test whether two values of a primitive type are equal , you use the equality operator ==

To see whether two strings have equal values, you must use the method equals rather than ==

E.g: s1.equals(s2) ;

equalsIgnoreCase: without considering the lower and uppercase characters;

Eg.

If ( “ Hello “. equalsIgnoreCase(“hello”))

System.out.print;n(“Equal”);

When applied to two strings( or any two objects) , the operator == tests whether they are stored in the same meory location.

When testing strings for equality, use either of the methods equals or equalsIgnoreCase.

Strings Comparison:

The method CompareTo tests two strings to determine their lexicographic order. In lexicographic ordering, the letters and other characeters are ordered according to their Unicode sequence. If s1 and s2 are two variables of type String that have been given String values , the method call

S1.compareTo(s2)

Compares the lexicographic ordering of the two strings and returns :

A negative number if s1 comes before s2;

Zero if the two strings are equal;

A positive number if s1 comes after s2;

Lexicographic ordering :

All uppercase letters come before all lowercase letters in lexicographic order.

Tips: To see whether two strings of letters are in alphabetic order, you must ensure that all the letters have the same case before using the method compareTo to compare the strings.

Nested if-else statements:

An if-else statement can contain any sort of statements within it. You can nest one if-else statement within another if-else statement;

If(balance >=0)

If( ( )

Statements;

Else

Statements;

Else

Statemnts

Important:

In an if-else statement, each else is paired with the nearest preceding unmatched if .

Multibranch if-else statements;

If()

Statements;

Else if ()

Statements;

Else if()

Statements;

Else

Statements;

Multibranch if-else statement:

If( Boolean\_Expression\_1)

Action\_1

else if( Boolean\_Expression\_2)

Action\_2

…

else if ( Boolean\_Expression\_n)

Action\_n

else

Default\_Action

If(n1 > n2)

Max = n1;

Else

Max = n2;

Can be expressed as follows:

Max = (n1 > n2) ? n1: n2;

? : on the right side of the assignment statement are known as the conditional operator;

If ( hoursWorked <= 40)

Pay = hourWorked \* payrate;

Else

Pay = 40 \* payrate + 1.5 \* payrate\* (hoursWorked – 40);

Pay = ( hoursWroked <= 40) ? (hoursWorked \* payrate): (40 \* payrate + 1.5 \* payrate\* (hoursWorked – 40);

Sometimes your program can encounter a situation that makes continuing execution pointless. In such cases, you can end your program by calling the exit method, as follows:

System.exit(0);

If ( NumberOfWinners == 0 )

{

System.out.println(“Error: Dividing by zero.”);

System.exit(0);

}

Else

{

}

Exit is a method in the class system. The number 0 given as the argument to System.exit is the returned to the operating system. Most operating systems use 0 to indicate a normal termination of the program and 1 to indicate an abnormal termination of the program.

3.2 Type Boolean:

Type Boolean specifies only two values : true and false;

A Boolean variable can be given the value of a Boolean expression by using an assignment statement, in the same way that you user an assignment statement to set the value of any other type of variable.

Boolean isPositive = (number > 0 );

If( isPositive)

System.out.println( “ The number is positive”);

Else

System.out.println( “The number is negative or zero”);

Binary operator of equal precedence are performed in left to right order. Unary operators of equal precedence are performed in right-to-left order.

Short-circuit evaluation:

For a Boolean expression of the form Expr\_A || Expr\_B if Expr\_A is true,Java concludes that the entire expression is true without evaluating Expr\_B. Likewise, for an expression of the form Expr\_A && Expr\_B, if Expr\_A is false, Java concludes that the entire expression is false without evaluation Expr\_B.

Java also allows you to ask for complete evaluation. In complete evaluation, when two expressions are joined by an and or an or operator , both subexpressions are always evaluated, and then the truth tables are used to obtain the value of the final expression. To obtain a complete evaluation in Java , you use & rather than && for and and use | in place of || for or;

Input and output of true and false;

Each case consisting of the keyword case followed by a constant—called a case label—then a colon, and then a list of statements;

Break statement:

When execution reaches a break statement , the switch statement’s execution ends.

If no break statement is found within the statements for a particular case, execution continues on with the next case until either a break statement is encountered or the end of the switch statement is reached.

If no case matches the value of the controlling expression, the default case – which begins with the keyword default and a colon is executed.

Enumerated data type or enunmeration.

An enumeration lists the values that a variable can have.

Enum MovieRating {E,A,B}

Notice that no semicolon follows an enumeration’s definition. ?