Chapter 3 - Decision Structures

Chapter 3 introduces the idea of decision structures, which is a fundamental concept in the field of computer science. Decision structures work by using conditions, executing tasks when something matches its conditions, if A happens, then do B. This is the building block of chapter 3, where the topics revolve around if-else statements, logical operations, and switch statements.

If statements allow users to compare values and execute tasks in an effective manner, and relational operators are used when comparing two values in an if statement. The following are examples of relational operators.

Symbol	Meaning
>	Is greater than
<	Is less than
>=	Is greater than or equal to
<=	Is less than or equal to
==	Is equal to
!=	Is not equal to

The result of comparing two values using a relational operator is a boolean value, where if the comparison is true, the boolean value is true, and the opposite is true if the comparison is false. When placed as a condition in an if statement, if the boolean value is true, the if statement will run, and if the boolean value is false, the if statement will not run. Else-if statements work in the same way as general if statements, but will only run when the first if statement did not run and the conditions for the statement are met. An else statement works the same way as an if

statement as well, but will only run if the first if statement does not run. Else statements do not have a particular condition to be met. It is important to note that else-if statements and else statements cannot exist without an if statement written on the top, and using these three types of statements is called using and if-else-if statement. ProgramOneCThree.java provides an example of how to use if-else-if statements.

Nested if statements allow programmers to run if statements that have two or more conditions that need to be met. This can be achieved by placing an if statement inside an if statement. However, this method tends to get more and more complicated if there are many conditions to be met, and is usually less preferred when compared with an if-else-if statement which does almost the same thing.

An if statement that has two different comparisons and conditions require the use of logical operators. Logical operators consider the two boolean values that the two different comparisons produce and create one overall boolean expression. The following are the logical operators used in Java.

Operator	Meaning	Effect
&&	AND	If both expressions are true, the final value is true
||	OR	If at least one of the expressions is true, the final value is true
!	NOT	A special case that does not need 2 different comparisons, but instead converts a true boolean value into false and vice-versa when applied to an expression

There is a unique scenario when it comes to comparing string objects, as they cannot use relational operators to compare the two strings. However, the String class has an equals method to compare two strings together. The general form of the method is,

stringA.equals(stringB);

where stringA and stringB are two different String variables. ProgramTwoCThree.java shows a more detailed example of this comparison. stringB can be replaced by a literal string, such as "Helloworld".

Switch statements allow a value to be tested beforehand, and then depending on the value, selecting which set of statements are going to be executed. Switch statements have cases that the value tested can be compared to, and the value of each case must be unique. It is another alternative to an if-else-if statement. ProgramThreeCThree.java shows an example of using switch statements.