Hi, welcome to Basic Program Structure.

☼Part C1: Boolean Data Type, Relational Operators, and Selection Basics.

☼The ability to make billions of decisions per second and make them repeatedly is the source of computing's power. In this lesson, we will introduce control in the form of selection. At the end of this lesson, you should be able to

Explain the usage of Boolean data type

Use relational operators to form conditional expressions

Discuss the basic concepts of program execution and flow control/ control flow

Explain selection basics

Describe the IF statement used for selection (branching)

☼Our plan is to begin this lesson with a quick introduction to selection with a scenario you are familiar with. In this lesson, we will go through

Boolean Data Type

Relational Operators

Selection Basics

one by one.

☼ In the previous lesson, scenario 3 illustrates the distance travelled between two points. Now, let's add one more feature to our scenario 3. After calculating the distance between two points, if the distance is greater than 3 kilometer, suggest to take a taxi

☼A decision box and a process box are inserted to the flowchart of scenario 3 to implement the newly added feature.

☼The newly added boxes form a **conditional statement, which performs different actions depending on whether the condition evaluates to true or false. Each decision box, which is a conditional expression, has two outgoing lines, true or**

false. If the conditional expression, " Is distance > 3km", is true, suggest to book a taxi. Or else, do nothing.

The simple decision presents a choice of doing one of two things: of doing one thing or the other. In programming, selection is the process of applying a decision to control: the choice of executing one part of a program or another.

More examples are given here.

4+5 is 9, 3+6 is 9, are they equal, yes, so execute true path, take action A.

99-55 = 44, 100-60 = 40, is 44 less than 40? no, 44 is not less than 40, so execute false path, take action B.

Are you late for today's class? No. That is good. Execute false path, take action B.

☼Representing the results of such a decision, to do one thing or another, lends itself to a straightforward representation in computer hardware. That is, binary decision, whether (True), or (False) in hardware is either one or zero. In most computer programming languages, a Boolean data type is a data type with only two possible values, either True or False.

Considering executing instructions one after another until a selection is reached. It is like a fork in the road. Associated with statements like a selection or repetition is a conditional expression, which is also known as Boolean expression and **may be composed of a combination of the Boolean constants True or False, Boolean-typed variables, Boolean-valued operators, and Boolean-valued functions.**

Can you think of any application of Boolean-valued operators? Let's look at those examples in the previous slide.

Click arrow button to go back to the previous slide (arrow button 7)

☼Equal to or not, less than or not, "late for today's class or not" means the time is greater than a specified time, so all these operators are numerical Relational Operators.

Click arrow button to jump to slide 9 (arrow button 9)

☼ A **relational operator** compares two numbers (**float** or **int**) and returns a **Boolean** value of either **True** or **False**.

An expression such as "f > 5.0" is a Boolean expression. The expression evaluates whether the value associated with f is greater than 5.0. The result of that evaluation is a Boolean. It is either True or False. In most programming language, we write the expression in a familiar way as what we have learnt in our maths class.

Expressing equality in a Boolean expression is a bit of a challenge . We are used to using the = sign for equality in mathematics but as you have seen, the equal sign represents assignment, not equality. To deal with this problem, many programming languages use a pair of equal signs, ==, to represent equality. For example, a == 1 is a measure of equality: Is the value represented by variable a equal to 1?

It is fairly easy to guess the remaining Boolean operators. This table lists them for you.

☼A simple recap before we move on to selection statement.

Control Flow controls which instruction should be executed next. By default, it is defined by the "sequence" concept, i.e., one after another. However, some structure can alter the flow, e.g., selection.

The flow control in a program is, in essence, logic.

When writing or reading a program, ensure that you could understand the flow, i.e., what should be executed next for every step.

☼To understand the flow, we need to learn those basic programming structures. IF statement is the most common programming "statement" used for branching.

The IF statement expresses selective execution. It is our first control statement and, simple as it is, it is quite powerful. Selective execution is one of the primary control mechanisms in general programming. It is a critical part of enabling computers to perform complex tasks.

The basic IF statement does the following:

1. Evaluate the conditional expression, which is a Boolean expression, yielding either True or False
2. If the conditional expression yields True, do this. To emphasize again, the choice of whether to execute this or not depends on the condition during program runtime. We usually **indent** this statement(s) to improve code readability that this part becomes to the true condition.
3. If the conditional expression yields False, ignore any code under IF (that is, do not execute it)
4. Continue the sequence by executing the next instruction.

☼Alternatively, when visualized as a flowchart, Evaluate conditional expression, if true, execute the statement along the true path, or else, skip it.

Please note that, there might be multiple statements in if block to be executed as a group. Details will be given in detailed programming lesson.

☼A quick summary:

In most computer programming languages, a Boolean data type is a data type with only two possible values, either True or False.

A **relational operator** compares two numbers (**float** or **int**) and returns a **Boolean** value of either **True** or **False**.

Control Flow controls which instruction should be executed next. By default, it is defined by the "sequence" concept i.e. one after another. However, some structure can alter the flow, e.g., selection.

IF statement is the most common programming "statement" used for branching.