Lesson 3: Creating a Smart Calculator

In this lesson, we will create a program to make a calculator that performs actions based on the user input. For example, if the user wants the calculator to add numbers, it will do so. If he/she wants the calculator to subtract numbers, it will perform that operation.

We will use conditional statements to direct the flow of actions in our program. In programming, conditional statements are used to perform different actions or computations depending on whether a condition is true or false.

Think of conditional statements in this format:

```
If this is true:
    then do this
```

This is called an if-then statement.

Here's an example from real life:

```
If it is raining outside:
    then wear rain boots
```

We can also add another action if the condition is not met. This is an if-else conditional statement.

Here's an example from real life:

```
If it is raining outside:
    wear rain boots
Else:
    wear tennis shoes
```

We can also have several actions depending on which condition is met. This is called an elif statement (short for else-if).

```
If it is raining outside:
    wear rain boots
Else-if it is snowing outside:
    wear snow boots
Else-if it is dry weather outside:
    wear tennis shoes
```

PART 1: Starting the Interactive Calculator Program

To begin, we ask the user for the desired operation (addition, subtraction, etc.). We will create a variable and name it operation to keep track of what the user inputs.
Next, we ask for them to input 2 numbers for the calculation.

```
print("Welcome to Calc-It-All!")

operation = input("Enter your operation: +, -, *, or /. ")

x = int(input("Enter the first number: "))

y = int(input("Enter the second number: "))
```

PART 2: Coding an If Statement

In Python, an `if` statement follows this formatting:

```
if condition:

    do something here
```

Note two things about the statement:

- There needs to be a colon at the end of the if statement
- The "then" code is indented (Python automatically indents the lines of code following an if statement.)

Type in the following below your current code. Here we have the program check for the user inputting a "+" symbol indicating an addition problem. If a "+" is input, the computer will add the two numbers (variables x and y) and then print out the sum.

```
y = int(input("Enter the second number: "))

if operation == "+":

    answer = x+y

    print("The answer is", answer)
```

PART 3: Coding the Elif Statements

In Python, an `elif` statement allows the computer to check multiple expressions to see which is true. It will then execute code for the true expression.

In this way, the elif statement is different from the `else` statement, which only has one possible action.

Now code the elif statements which represents and actions if the user types in a subtraction, multiplication, or division symbol.

```
if operation == "+":
    answer = x+y
    print("The answer is", answer)

elif operation == "-":
    answer = x-y
    print("The answer is", answer)

elif operation == "*":
    answer = x*y
    print("The answer is", answer)

elif operation == "/":
    answer = x/y
    print("The answer is", answer)
```

Run your program. Test to make sure that the calculator works with all four operations.

You will discover that if you wish to re-run the program , you will need to close the shell and re-run the module. In Lesson 4, we will use a loop to enable the user to easily re-run a Python program.

PART 4: Comparison Operators = Determining If Two Values are Equal

In Python and almost all coding languages, you will use a *comparison operator* to compare the values on either side of the operator. == is a comparison operator.

In our program, we used:

```
if operation == "+":
```

to see whether the left operand (the variable called operation) was equal to the right operand (+). If so, then the condition is true and the computer will perform the action which follows.

== is called a comparison or relational operator.

Here are some additionalcomparison operators that you will likely use:

Operator	Description	Code Example
!=	Checks if the value of two operands are equal or not. If the values are not equal then the condition is true.	a != b
>	Checks if the value of the left operand is greater than the value of the right one. If yes, then the condition is true.	a > b
>=	Checks if the value of the left operand is greater thanor equal to the value of the right one. If yes, then the condition is true.	a >= b
<	Checks if the value of the left operand is less than the value of the right one. If yes, then the condition is true.	a < b
<=	Checks if the value of the left operand is less than or equal to the value of the right one. If yes, then the condition is true.	a <= b