Basic Programming Logic - if/loops

Author	josh
Date	2023-01-07

Overview

Have you ever wondered how computers think, but never programmed one before? There is a wealth of specific programming tutorials for various languages out there. This tutorial will focus more generally on programming *logic* – how programmers *think* when writing code, rather than how to write in one particular language. We will use some simple examples from the popular Python programming language while learning this logic.

Programming Logic – Decisions and Repetitions

Making Decisions – if Statements

One of the most basic things programmers need to do is making "if else" decisions in their code. What should the code do if the user inputs one thing, what to do if they input another thing, or some other corner case. Our code needs to be able to go in one direction or another depending on input. A very simple example might be a user's preference for one unit of measurement or another. Let's say we're writing a program that tracks mileage for activities. We can display this distance in miles or kilometers depending on what the user decides:

```
if units == "mi":
     calc_miles()
elif units == "km":
     calc_kilometers()
else:
     display_error()
```

This simple example shows how our program can calculate data based on miles or kilometers depending on what the user wants to see. If the user selects miles, we call the calc_miles() function. Else, if the user picks kilometers we'll call the calc_kilometers() function. We've allowed our program to *decide* the correct code path to run based on user input, or display an error if the code receives some invalid unit.

Looping Indefinitely – the while Loop

Another important part of coding logic is the loop. Computers are excellent at doing *repetitive* tasks, where people usually aren't. A computer can wait for input indefinitely, display some image on the screen in an infinite loop over time, and much more. A basic type of loop

commonly used in programming is called the “while” loop or “do while” loop. For example, let’s say we have a program that will read data from a car’s OBD port. This program will read data from the car until the user decides to quit. Our code might look like this:

```
while True:
     keep_reading = user_input()
     if keep_reading == False:
          break
read_data()
parse_data()
display_data()
```

This code will read the OBD data, parse it, and display it to the user on every loop. Every loop, we also check if the user enters the quit command. This is the “break” condition that will exit the loop. Until that break condition is met, the loop will continue indefinitely.

Looping a Few Times – the for Loop

Another useful type of loop is the for loop. This allows us to loop a *specific* number of times, rather than indefinitely until some break condition is met. For example, let’s say we have a program that calculates statistics from a list of numerical data. Sum, average, min, max, etc. Our code that calculates the sum might look like this:

```
sum = 0
for item in dataset:
     sum = sum + item
display_sum(sum)
```

This loop goes through every item in the data set, adding each individual number to the sum total. This loop will quit when all numbers in this list are exhausted, and therefore the complete sum has been calculated.

Basic Logic, Big Uses

Programming logic is built on foundational operators such as if, else, while, for, etc. These tools exist in many different programming languages from Python to C to Rust to JavaScript and many, many more. If you learn how to *think* in one programming language, you can transfer this knowledge to a new one! In fact, many programmers use multiple tools depending on the project. Coding is a fascinating exercise in logic, so if you are thinking of getting started now is the time. Go code, have fun!