Jared Tweed
Computing ID: jmt25
Python Programming Conditionals (Using Turtle)
https://youtu.be/fzvqQxVP_l4
March 26, 2024

Topic

Teaching Python Programming Conditionals (Using Turtle) for Middle-school students with no prior technical experience (Grades 6-8; ages 11-14). Note that I added subtitles in the youtube video that you can turn on and off.

Why I chose the topic:

I chose the topic of conditionals because it is difficult enough for the video to be necessary for people to search for. Additionally, I believed that I could make a video on this topic that is better than the videos that already exist for this given audience and topic.

Why I chose the specific audience:

I chose middle schoolers because they need to learn to code at that age. I believe that people should have access to relevant materials as early as possible. I also think this is especially true for coding because it helps children learn to think critically at an early age.

How did I develop the video:

I used Google Slides to develop the video. I also used VSCode to write and run the code. I chose to use VSCode because the way it highlights text, and darkens ignored if-statements is especially helpful for my uses. I also used https://www.online-ide.com/ when I disliked the way VSCode highlighted my code. VSCode also allowed me to highlight multiple sections of text simultaneously (e.g., when explaining the "if" keyword and ":". I also used clip-champ to make the video shorter, and edit out my stuttering. I also used my analysis of the videos I referenced in the week 9 exercise to help me make my explanation clearer.

How did I use Universal Design for Learning

(https://udlguidelines.cast.org/)

Principles (1):

- As you can see from the checkpoints below I used both engagement and representation.

Guidelines (2):

- As you can see in the checkpoints below, I used guidelines 2, 3, and 8.

Checkpoints (4):

- Activate or supply background knowledge (3.1). When explaining the ">=" and "<=" symbols, I used their knowledge of the ">" and "<" symbols as a reference point to define the

symbol. I even used the slides to communicate how it is similar to their "≥, ≤" math symbols. I used the similarity with the english word "if" to help me explain the concept of if-statements.

- Illustrate through multiple media (2.5). I did this in the "else/elif" slide when I intentionally used a code interpreter that would highlight the if-statement that would run, represented which if-statement would run via arrows, and represented which if-statement ran via showing the output. I also regularly represented the output rather than just telling them what it would be. When expressing the value of the "ourTurtle.xcor()" variable, I drew an arrow from the code where it was defined.
- Heightened the salience of goals and objectives (8.1). I did this when suggesting that they learn this to impress their friends and make their own games.
- Foster collaboration and community (8.3). I did this when suggesting they learn this to impress their friends. When the friends are impressed, they will likely ask their friends to teach them how to make a similar program as well.
- Increase mastery-oriented feedback (8.4). Between each 3rd of the video, I encouraged the viewer to check their knowledge and rewatch the previous section if they were confused.

I incorporated the feedback from the analysis of the week 9 videos from the classmates as well to help produce this video. For example, I explained the broader concept before explaining the complex details.

Appendix

Script

Let's learn python programming conditionals, so that you can impress your friends and make your own fun games.

This code creates our turtle named "ourTurtle". You do not need to understand this code in detail. All you need to know [pause] is that all the text to the right of a hashtag is a "comment" that is ignored by the computer.

This code moves our turtle 100 pixels to the right, and 100 pixels up.

This code turns our screen green IF our turtle's position is more than 50 pixels to the right. Let's go into detail on how this works.

We call this highlighted section of text, the “condition”. When this condition is true, the code beneath it that is shifted to the right, runs. When the code is shifted to the right, we say that it is “indented”. In this example, the indented code turns the screen green. The computer runs the indented code only “if” the condition is true.

Note that the “if” keyword and colon are required for the code to work as desired.

Because we moved our turtle 100 pixels to the right, it caused this highlighted text here to represent the number “100”. We will refer to this highlighted text as a variable. This **variable** represents how far we moved our turtle to the right.

You might recognize this symbol. This symbol makes the condition true only if the value on the left side is larger than the value on the right side. Remember that the **variable** on the left represents 100. Because 100 is larger than 50, this condition is true.

To make sure we understand everything clearly, pause the video and look at this slide again. Are you able to understand why the condition is true? It's related to the value represented by “turtle dot xcor”.

Comparison Symbols

This symbol means "greater than". If the value on the left side is more than the value on the right side, the condition is true. As you can see, the amount our turtle moved right is 100 pixels, and that is more than 50 pixels. Thus, because 100 is more than 50, this condition is true. Below it is the “less than” symbol. The less than symbol makes the condition true only if the left side is less than the right side. As you can see here, that is false.

This symbol means "greater than or equal to". If the value on the left side is “equal to” or “more than” the value on the right side, the condition is true. As you can see, this symbol is the exact same as the “greater than” symbol on the last slide, except there is one difference. This symbol makes the condition “true” if the left and right side of the symbol have an equal value. Below it is the “less than or equal to” symbol. This symbol also makes the condition true when both the left and right sides have equal value. Otherwise, it is also identical to the “less than” symbol.

This “double equals” symbol means "equal to". If the value on the left side is equal to the value on the right side, the condition is true. As you can see, the number of pixels the turtle traveled to the right is 100, which is not equal to 50, therefore, this condition is false. This condition would be true only if both sides had the same value. Below it is the “not equal to” symbol, which is true only if both sides

represent different values. As you can see here, these values are not equal, and thus the condition is true, and the indented code which turns the screen green runs.

So far, these are the comparison symbols that we have gone over. Make sure you understand each of them. If you are not ready, go back and rewatch the content until you understand these, and are ready to continue.

Other Symbols

You can include the word “not” before a condition to make the condition have the opposite value. Thus, “not true” is equal to “false”, and “not false” is equal to “true”.

This variable here represents 100, because that is the number of pixels we moved our turtle to the right. This variable here also represents 100, but that is because that is the number of pixels we moved our turtle up. Notice I put the word “and” between two conditions, each with a green underline. This “and” word makes the if-statement true ONLY IF both those conditions on each side are true. As you can see here, only one of the conditions is true, thus if-statement is false, preventing the code which turns the screen green from running.

Notice I put the word “or” between two conditions, each with a green underline. This “or” word makes the if-statement true ONLY IF either or both of those conditions on each side are true. As you can see here, one of the conditions is true, and that is enough to make the if-statement true, causing the code which turns the screen green to run.

else/elif

If the **condition** is true after an “if” or “elif” statement, like it is here, the computer runs the indented code under the statement and skips all the following “else” and “elif” statements until other code is written.

If the **condition** is false after an “if” or “elif” statement, like it is here, the computer jumps to the next “else” or “elif” statement. If the computer encounters an else-statement, the computer runs the indented code under it, and continues the running program.

Source Code

https://github.com/JaredTweed/Conditionals-Tutorial-For-Kids

Word Count: 1637

Page count: 4 (excluding the cover page).