CSCI 141 - Fall 2019
Lab 3: Conditionals and Boolean Logic
Due Date: Friday, October 18th at 9:59pm

Introduction

This lab gives you practice with `if` statments (also sometimes called *conditional*, or *selection* statements. In the process, you'll also get some more experience with Boolean operators. The idea is the following: your goal is to write a program that recommends what clothing items to wear based on the weather conditions indicated by a user.

If you have questions, be sure to ask the TA: your TA is there to help you! By now you've seen how to use Thonny on both your Windows and Linux accounts. You are free to select whichever operating system you want to use.

1 Setup

We recommend creating a new directory/folder called *lab3* on your N drive (Windows) or in your home directory (Linux). In your lab3 directory, create a new Python file `clothing_picker.py`.

2 Unary Selection

Unary selection is a fancy name for a simple `if` statement. You've seen these already in lecture: the `if` statement allows you to execute a sequence of statements (or a *code block*) if a given **boolean expression** evaluates to `True`, or skip over the code block if the expression evaluates to `False`.

The code block inside an `if` statement must contain one or more statements. In Python, the code block associated with an `if` statement is distinguished by indenting lines of code immediately underneath the line containing the `if` keyword of the selection statement. The syntax and structure of a unary selection statement are shown below:

```
if boolean_expression:
    statement_1
    statement_2
    statement_3
```

For this first version, write a single unary selection statement that checks whether the user has specified whether it is windy or not. If it is windy, the program should tell the user not to bring an umbrella. Pseudocode and sample input and output for this first version of your program are given below.

- Ask the user if it is windy

```
>>> %Run clothing_picker.py
  Is it windy? yes
  Don't bring an umbrella because it's windy.
>>> %Run clothing_picker.py
  Is it windy? no
>>>
```

Figure 1: Sample output for the initial version.

- Save user input into a variable
- If it is windy, print "Don't bring an umbrella because it's windy."
- If it is not windy, do nothing

3 Binary Selection

We've also discussed binary selection, which is a fancy name for an `if/else` statement. It has an `if` clause and an indented code block just as in unary selection, but it also has an `else` clause and code block that is executed whenever the Boolean expression in the if clause evaluates to `False`. The syntax and structure of a binary selection statement are shown below below, where statements 1 through 3 are the code block for the `if` clause, and statements 4 and 5 constitute the code block for the `else` clause.

```
if boolean_expression:
    statement_1
    statement_2
    statement_3
else:
    statement_4
    statement_5
```

Next, modify your code so that it still prompts the user to answer whether it is windy, but this time if the answer is "no", then have the program output, "It is not windy. Bring an umbrella if you wish." If it is windy, the program should output the same as before. Sample input and output for this second version of your program is shown below.

```
>>> %Run clothing_picker.py
  Is it windy? no
  It is not windy. Bring an umbrella if you wish.
>>>
```

Figure 2: Sample output, Binary Selection

4 Boolean expression with logical operators

We've discussed in lecture how to use more complicated boolean expressions; specifically, the logical operators `or`, `and`, and `not` were presented. Modify your code to also prompt the user to for whether it is sunny or cloudy. Retain the `if, else` code as you've already written, except change the Boolean expression to check if it is windy and sunny. If the user specifies yes, then the output should be "It's windy and sunny, so bundle up and don't bring an umbrella." If it is not both windy and sunny, have the program output, "It is not both windy and sunny."

Pseudocode for the revised version of your program is shown below. Sample output is shown in Figure 3.

- Ask user if it is windy, save input into a variable
- Ask user if it is sunny or cloudy, save input into a second variable
- If it is windy and sunny, output "It's windy and sunny, so bundle up and don't bring an umbrella."
- Otherwise, output "It is not both windy and sunny."

Note: Here and in all further parts of the lab, you may assume that the user responds to the sunny/cloudy prompt with the exact input `"sunny"` or `"cloudy"`; you do not need to handle other inputs. Also notice that the instructions above are phrased only in terms of `sunny` and `not sunny`.

```
>>> %Run clothing_picker.py

  Is it windy? yes
  Is it sunny or cloudy? sunny
  It's windy and sunny, so bundle up and don't bring an umbrella.

>>> %Run clothing_picker.py

  Is it windy? yes
  Is it sunny or cloudy? cloudy
  It is not both windy and sunny.

>>>
```

Figure 3: Sample output for boolean expression with logical operators

5 Nested if statements

As shown in lecture, it is possible to nest an entire selection statement (`if` statement with an `else` clause) inside of a code block of an existing `if` statement. The syntax is shown in below. To make it easier to see, a box has been drawn around the outer-most and inner most if statements.

```
if boolean_expression_1:
    if boolean_expression_2:
        statement_1
    else:
        statement_2
else:
    statement_4
    statement_5
```

Modify your code so that the outer condition (*boolean_expression1*) checks if it is windy, and the inner condition (*boolean_expression2*) checks whether it is sunny. If it is windy and sunny, print "It is windy and sunny,"; if it is windy and not sunny, print "It is windy and not sunny"; if it is not windy, print "It is not windy." Sample output is shown In Figure 4.

```
>>> %Run clothing_picker.py
  Is it windy? yes
  Is it sunny or cloudy? sunny
  It is windy and sunny.
>>> %Run clothing_picker.py
  Is it windy? yes
  Is it sunny or cloudy? cloudy
  It is windy and not sunny.
>>> %Run clothing_picker.py
  Is it windy? no
  Is it sunny or cloudy? sunny
  It is not windy.
>>>
```

Figure 4: Output of program with nested conditionals

6 Clothing picker: chained conditionals

One more possible clause in a conditional statement an `elif` clause. An elif, which is short of else if, contains a Boolean expression that is checked ONLY if the first `if`'s condition, and all preceding `elif` conditions all evaluate to False. Unlike an `else`, whose code block is ALWAYS executed if the if condition evaluates to False, the code block of an elif is executed only if the conditional of the elif evaluates to True. Note that including an `else` is never syntactically required; an `if` statement can have zero or more `elif` clauses and zero or one `else` clause. In the program you submit, you should make sure you're demonstrating at least one use of an `else` clause.

This time, have your program prompt the user to ask if it is sunny, and also ask for the **temperature**. Modify your program so that the `else` code block has a nested if statement as

Table 1: Sample input/output combinations

Sunny	Temperature	Output
Yes	Less than 60 degrees	Wear a sweater
Yes	60 degrees exactly	Woo hoo, it is 60 degrees. Wear what you want
Yes	More than 60 degrees	Wear a t-shirt and flip flops
No	Less than 40 degrees	Wear a coat and hat
No	Between 40 and 50 degrees	Not quite freezing, but close. Bundle up
No	50 degrees exactly	A jacket is best
No	More than 50 degrees	Wear a long sleeved shirt

well. Both of the nested if statements will now have `if`, `elif`, and `else` clauses (see sample below). The outer condition should check whether it is sunny, and inner `if/elif/else` should make recommendations according to the temperature and whether or not it is sunny. Don't forget to convert the temperature input to an `int` and recall that you can use comparison operators like `<=` and `>=` to get the boolean result of numerical comparisons.

Write appropriate conditions that rely on the user's input (whether it is sunny and the temperature), and write appropriate print statements that produce the clothing recommendations indicated in Figure 5.

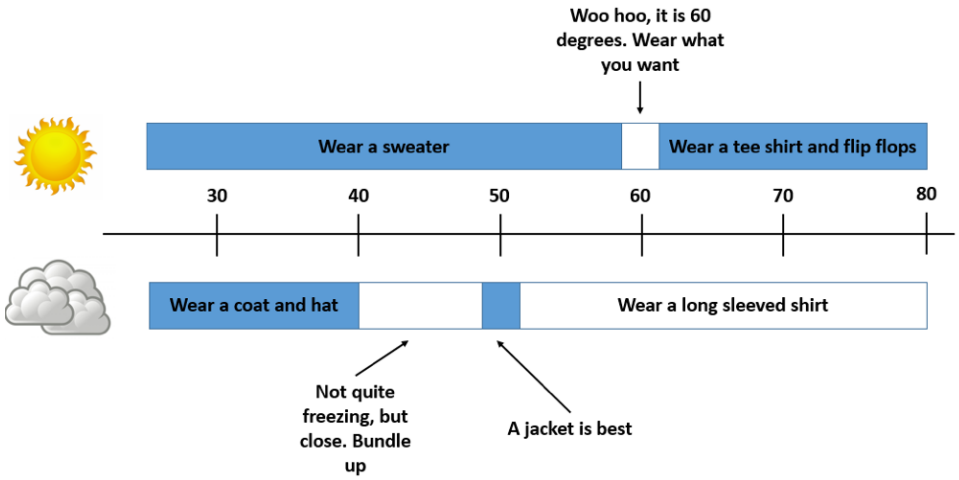

Figure 5: Schematic of logic for clothing picker

Table 1 shows the sunny/temperature combinations and their corresponding output value (clothing recommendation) that your program should print. Sample output is shown in Figure 6.

Submission

Upload `clothing_picker.py` to Canvas for grading.

```
>>> %Run clothing_picker.py

  Is it sunny or cloudy? sunny
  What is the temperature? 80
  Wear a t-shirt and flip flops.

>>> %Run clothing_picker.py

  Is it sunny or cloudy? sunny
  What is the temperature? 40
  Wear a sweater

>>> %Run clothing_picker.py

  Is it sunny or cloudy? cloudy
  What is the temperature? -24
  Wear a coat and hat

>>> %Run clothing_picker.py

  Is it sunny or cloudy? cloudy
  What is the temperature? 68
  Wear a long sleeved shirt

>>>
```

Figure 6: Sample clothing picker program output. Note that this does not display all possible cases in the table above, but your program must work for all of them.

Rubric

Your file is called `clothing_picker.py`	1 point
The top of `clothing_picker.py` has comments including your name, date, and a short description of the program's purpose. Comments placed throughout the code explain what the code is doing.	3
Your program makes use of `if`, `elif`, and `else`.	6
Your program correctly prompts the user and stores the user's input.	3
Your code provides unique clothing combinations for each of the sunny/temperature combinations in the table in this lab handout.	7
Total	20 points