Set 8: Conditionals Part 2

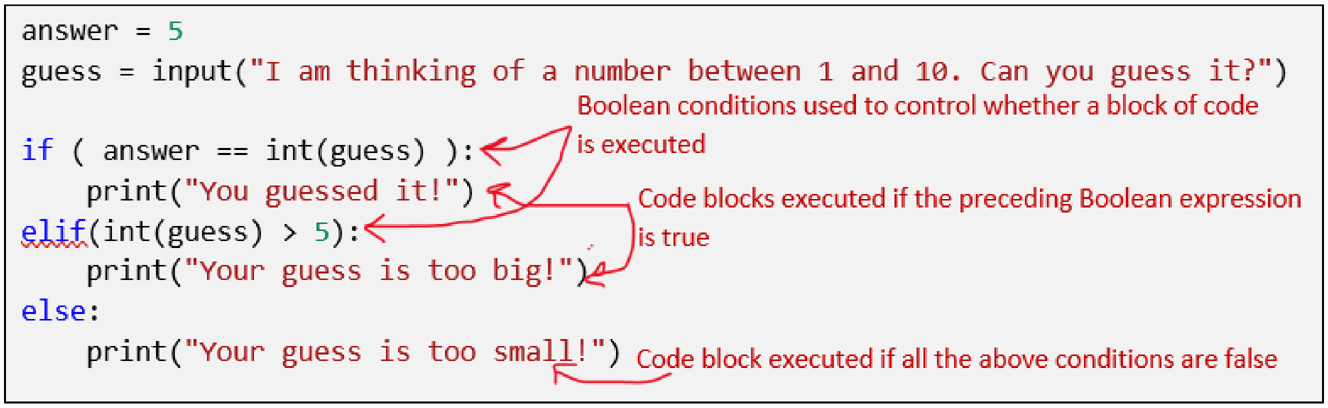
Skill 8.1: Predict the output of an *elif* statement
Skill 8.2: Write a Python *elif* statement
Skill 8.3: Predict the output of nested *if-else* statement
Skill 8.4: Write a nested *if-else* statement

Skill 8.1: Predict the output of an *elif* statement

Skill 8.1 Concepts

In addition to the *if* and *else-if* statements we learned previously, in Python, we have one more conditional statement called *elif* statements. The *elif* statement is used to check multiple conditions only if the given condition is false. It's similar to an *if-else* statement – the only difference is that *elif* statements can be applied to evaluate multiple conditions.

Below is an example,



```
answer = 5
guess = input("I am thinking of a number between 1 and 10. Can you guess it?")

if ( answer == int(guess) ):
    print("You guessed it!")
elif(int(guess) > 5):
    print("Your guess is too big!")
else:
    print("Your guess is too small!")
```

Boolean conditions used to control whether a block of code is executed
Code blocks executed if the preceding Boolean expression is true
Code block executed if all the above conditions are false

Skill 8.1 Exercise 1

Skill 8.2: Write a Python *elif* stateemnt

Skill 8.2 Concepts

Notice each logical statement above is followed by a colon (:) and all the code that is part of the logical statement is indented. But, also notice that in the example that evaluates the time variable we used two and statements for every conditional. While the code works, there is a simpler way to write this code!

```
if(time >= 22):
    print("go to bed")
elif(time > 21):
    print("hang out")
elif(time > 18.5):
    print("do homework")
elif(time > 17.5):
    print("eat dinner")
elif(time > 15.25):
    print("go to band practice")
elif(time > 7.5):
    print("wake up and get ready")
else:
    print("stay in bed")
```

In the example above, we wrote the code with the most restrictive option first. This eliminated the need to check for two boundary conditions. This is a good rule of thumb for both simplifying the logic reducing unexpected outcomes.

Skill 8.2 Exercise 1

Skill 8.3: Predict the output of nested *if-else* statement

Skill 8.3 Concepts

Nested "if-else" statements mean that an "if" statement or "if-else" statement is present inside another *if* or *if-else* block. This feature allows us to evaluate complex logical conditions in a given program.

```
grade = 92

if(grade > 90):
    if(grade > 100):
        print("you got an A+")
    if(grade < 93):
        print("you got an A-")
```

This block of code is only evaluated if the grade is greater than 90. It is said to be "nested".

Skill 8.3 Exercise 1

Skill 8.4: Write nested *if-else* statement

Skill 8.4 Concepts

In the examples above we see that when writing nested if-else statements, each nested statement must be indented and follow the same format as the parent. You can write as many nested statements as you like, just remember to use proper indention for each nested block.

Code

```
age = 25
has_driver_license = True


if age >= 18:
    print("You are old enough to drive.")
    if has_driver_license:
        print("You also have a driver's license, so you can legally drive.")
    else:
        print("You are old enough, but you need a driver's license to legally drive.")
else:
    print("You are not old enough to drive.")
```

Output

```
You are old enough to drive
You also have a driver’s license, so you can legally drive
```

Skill 8.4 Exercise 1