Lesson 5

Introduction to IF Statements

As you continue to work through the lessons and expand your knowledge of Python (and coding in general!), you will be using if statements very regularly. Basically, if statements test if a condition is true or false. Let's go back to when we were first introduced to if statements in the Number Guessing Game.

```
randomNumber = random.randint(1, 10)  # generate random number between 1 and 10

guessedNum = (int)(input(" "))  # user inputs an INT that will be compared with the random value

if guessedNum == randomNumber:  # if the guessed number is equal to the randomly generated number
    print("You Won!") # code block inside the if statement
```

In the above example, the if statement is used to test if the guessed number that the user has inputted is equal to the randomly generated number.

Note: in programming "==" means "is equal to"

The code block inside the if statement ("print("You Won!")") will only run if the condition in the if statement is met.

How To Use "if" Statements

```
if what is in here is true:
    do what is in the code block after the if

otherwise just continue after the end of the code block
```

- If what is after if is true, then DO run the code in the code block (code with spaces).
- If what is before if is false, then DO NOT run the code in the code block.

How To Use “else” Statements

else statements can also be used along with if statements. They are usually used after an if statement and will only run the code in their code block if the condition in the previous if statement was not true.

```
if what is in here is true:
    do what is in the code block after the if
else:
    otherwise do what is in this code block after
```

- If what is after if is true, then run the code in that code block.
- If what is before if is false, then run the code in the else code block.

Below is an example of how to use if/else statements. Try it out for yourself!

Hint: Increase the value that is stored in the myNumber variable and see what happens!

```
myNumber = 5

if myNumber > 8: # if myNumber is GREATER THAN 8
    print("Number is greater than 8")
else: # if myNumber is NOT GREATER THAN 8
    print("Number is less than or equal to 8")
```

How To Use “elif” Statements

elif statements (short for “else if”) can also be used when more conditions are needed. They are usually used after an if statement and will only run the code in their code block if the condition in the previous if/elif statement was not true.

```
colour = input("What is your favourite colour? ") # user inputs their favourite colour

if colour == "red": # if the user said red
    print("Your favourite colour is red")
elif colour == "green": # or else if the user said green
    print("Your favourite colour is green")
elif colour == "blue": # or else if the user said blue
    print("Your favourite colour is blue")
else: # or else if the user said none of these
    print("Your favourite colour is something else...")
```

- If the user inputted the colour “red”, the program will print “Your favourite colour is red”. If the user inputted something else, the program will move onto the first elif statement.
- If the user inputted “green”, the program will print “Your favourite colour is green”. If the user inputted something else, the program will move onto the second elif statement. This step is repeated for if the user inputted “blue”.
- If the user inputted any colour other than “red”, “green” or “blue”, then the program will print “Your favourite colour is something else...”.

Tasks:

1. Ask the user if they are right-handed or left handed.

```
userInput = input("Are you right-handed or left-handed? (Type right or left) ")
```

2. If the user is right-handed, then print “You are right-handed”.
3. Or else if the user is left, then print “You are left-handed”.
4. Or else if the user inputs something completely different, then print something like “Unknown hand type...”.

If Item Is In/Not In List

If we have a list of objects, we can test to see if an objects is already in that list using if statements. For example:

```
numbers = [0, 2, 4, 6, 8, 10] # list of int numbers

if 4 in numbers:
    print("Number 4 is in the list")
else:
    print("Number 4 is not in the list")
```

You can also do this the opposite way where you test if an object is **not** in the list.

```
numbers = [1, 3, 5, 7, 9, 11] # list of int numbers

if 8 not in numbers:
    print("Number 8 is not in the list")
else:
    print("Number 8 is in the list")
```

Tasks:

1. Declare a list of strings that contain the names of a few different types of animals.
2. Test whether the following are in the list and print out whether they were found in the list or not:
    a. Bird
    b. Snake
3. Test whether the following are **not** in the list and print out whether they were found in the list or not:
    a. Cow
    b. Dog
4. Update your code so that if one of the animals given above is **not** already in the list, add it to the end of the list.

Castle Dragonsnax Project

You have snuck into a scary castle late at night and are presented with three, giant, wooden doors. Write a program that will ask the user to pick one of the three doors to open. Each door has a different consequence for the player when it is opened.

- One of the doors results in the player finding a giant room full of treasure - player wins.
- Another one of the doors results in the player being hit by a giant ogre's club - player loses.
- The final door will result in the player finding a sleeping dragon that awakes when it hears the player come in the room and eats the player.

Tasks:

1. Create the setting for the player by printing out an introduction to the story. For example:

```
print("You are in a dark mysterious castle.")
print("In front of you are three doors. You must choose one.")
```

2. Ask the user to input the door they wish to choose (1, 2 or 3).
3. If the user chooses door 1, print out that the player has found the treasure and the player has won.
4. Or else if the user chooses door 2, print out that the player has been hit by the ogre's club and the player has lost.
5. Or else if the user chooses door 3, print out that the player has been has found the dragon, has been eaten and the player has lost.
6. Or else if the player has chosen something else, print out that they must choose either door 1, 2 or 3.
7. Print out that the user must run the program again to have another go (at the end, outside of the if/elif/else statements).

Extra Tasks:

1. Add an extra layer of doors that the player will come upon **before** meeting the previous doors in the castle. Perhaps the first door that the player chooses will lead them to the great hall and another door will lead them to the grand staircase etc. (be creative!!!) where they will find the other three doors that we had previously encountered.

   Hint: This task will require the use of nested if statements (if statement within another if statement code block). See the pseudo-code block below as an example.

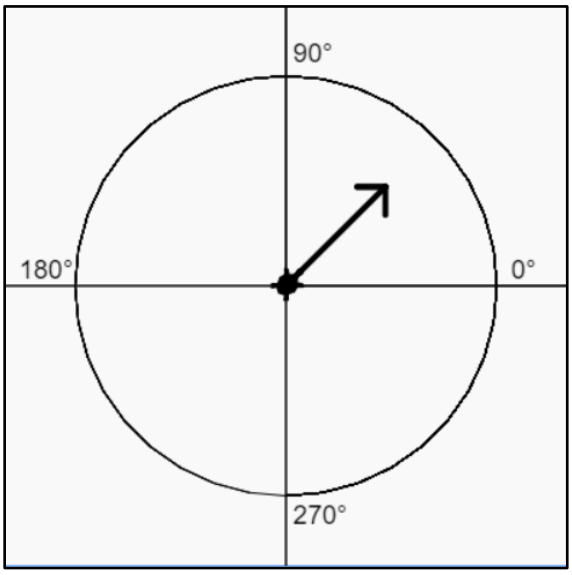
```
if player chooses door 1:
    print something
    player chooses another door

    if player chooses door 4:
        print something only if door 1 was chosen originally
    elif player chooses door 5:
        print something else only if door 1 was chosen originally
    elif player chooses door 6:
        print something else only if door 1 was chosen originally
    else:
        print something only if door 1 was chosen originally
```

Giving Directions Project

In the project, the user will be able to tell Tina the Turtle where to go on the screen using conditional statements to give her directions. Initially, the user will be able to tell Tina to go either right or left on the screen (we will add up and down later on also).

An important thing to know when doing this project is to understand what direction Tina the Turtle is facing. Tina can be facing anywhere from 0 degrees to 359 degrees (a full, circular rotation). See the diagram below for a representation of this.



From the diagram we can see the following:

- 0° means Tina is facing to the right.
- 90° means Tina is facing up.
- 180° means Tina is facing to the left.
- 270° means Tina is facing down.

To find out what direction Tina is facing we can use the heading() method which will tell us the number of degrees Tina is facing.

Tasks:

1. Import the Turtle module and initialize Tina the Turtle how we always do.
2. Ask the user what direction they want Tina to go in (“left” or “right).
3. Add in the conditional statements for each case. You will need an if statement for when the user chooses to go left, an elif statement for when the user chooses to go right and an else statement for when the user puts in something else entirely. Don’t worry about putting in the code block just yet, let’s worry about the condition first!

```
directionInput = input("What direction should Tina go?") # Ask user what direction to go
```

```
if directionInput == "left": # user chooses to go left
    #  code block

elif directionInput == "right": # user chooses to go right
    #  code block

else: # user chooses something else
    #  code block
```

4. Fill in the code blocks under each condition.
    a. If the user chooses “left”, check to see what direction Tina is facing, turn her around to face the opposite way if necessary and then move her 50 pixels to the left.

```
if tina.heading() == 0:  # if Tina is facing to the right
  tina.left(180)         # turn left 180 degrees
  tina.forward(50)       # go forward 50 pixels
else:                    # else Tina is facing to the left
  tina.forward(50)       # go forward 50 pixels
```

    Note: The lines of code above are nested within another if statement (just like in the extra tasks of the Castle Dragonsnax Project).

    b. Or else if the user chooses “right”, check to see what direction Tina is facing, turn her around to face the opposite way if necessary and then move her 50 pixels to the right. Use the code above to help you with this one (it’s very similar!).
    c. Or else the user chooses something else that is incorrect, print out some sort of error message, explaining that the user must input either “left” or “right”.

5. Allow the program to run continuously so the user can keep inputting directions for Tina to follow. This is done using an **infinite while loop** (we will meet this in the next lesson, don’t worry too much about it for now!).

```
while True:
  # entire code block
```

Basically, put all of the code we have done up to now (except importing the Turtle module and initializing Tina the Turtle) inside the while loop code block so it will repeatedly ask the user to input another direction after the previous one has taken place.

Extra Tasks:

1. Allow the user to input “up” and “down” directions that Tina will follow. In the previous tasks, to make Tina go left or right, we simply had to check if she was facing left or right. Now we must check if she is facing up, down, left or right every time the user inputs a new direction.

```
if user chooses “up”:

    if Tina is facing right:
        turn left 90 degrees
        go forward
    elif Tina is facing left:
        turn right 90 degrees
        go forward
    elif Tina is facing down:
        turn right 270 degrees
        go forward
    else:
        go forward
```