Chapter 3 - Conditional Statements And Loops

IF Else Statement

- IF Statement:

-
  ```
  a = 33
  b = 200
  if b > a:
    print("b is greater than a")
  ```
- Else if (elif):

-
  ```
  a = 33
  b = 33
  if b > a:
    print("b is greater than a")
  elif a == b:
    print("a and b are equal")
  ```

Else

-
  ```
  a = 200
  b = 33
  if b > a:
    print("b is greater than a")
  elif a == b:
    print("a and b are equal")
  else:
    print("a is greater than b")
  ```

```
#Short Hand If:
    if a > b: print("a is greater than b")
#Short hand if else:
    a = 2
    b = 330
    print("A") if a > b else print("B")
#Short hand else-if:
    a = 330
    b = 330
    print("A") if a > b else print("=") if a == b else print("B")
```

#This technique is known as **Ternary Operators**,
or **Conditional Expressions**.

Nested If Else Statement:

```
x = 41
if x > 10:
  print("Above ten,")
  if x > 20:
    print("and also above 20!")
  else:
    print("but not above 20.")
```

And Keyword:

```
a = 200
b = 33
c = 500
if a > b and c > a:
    print("Both conditions are True")
```

OR Keyword:

```
a = 200
b = 33
c = 500
if a > b or a > c:
    print("At least one of the conditions is True")
```

Exercise:

- Make a program to check that number is positive or negative.
- Write a program to check whether a number is divisible by 5 and 11 or not.
- Write a program to check whether number is even or odd.
- Write a program to check whether a year is leap or not.
- Write a program to check whether a character is alphabet or not.
- Write a program to check whether alphabet is vowel or constant.
- Write a program to find all the roots of quadratic equation.
- Write a program to calculate profit and loss.

- Write a program to input marks of five subjects physics, chemistry, maths, biology, computer.Calculate percentage and grade according to following:
  Percentage >= 90% : Grade A
  Percentage >= 80% : Grade B
  Percentage >= 70% : Grade C
  Percentage >= 60% : Grade D
  Percentage >= 40% : Grade E
  Percentage < 40% : Grade F
- Write a program to input basic salary of an employee and calculate its gross salary according to following:
  Basic Salary <= 10000 : HRA = 20%, DA = 80%
  Basic Salary <= 20000 : HRA = 25%, DA = 90%
  Basic Salary > 20000 : HRA = 30%, DA = 95%

Functions:

- A function is a block of code which only runs when it is called.
- You can pass data, known as parameters, into a function.
- A function can return data as a result.
- Creating a Function:
  - In Python a function is defined using the def keyword:
    - def my_function():
      print(“Hello Students!")

- Calling a Function:
  - To call a function, use the function name followed by parenthesis:

    -
      ```
      def my_function():
        print("Hello from a function")
      my_function()
      ```
- Arguments
  - Information can be passed into functions as arguments.Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.

  -
    ```
    def my_function(fname):
      print(fname + " Refsnes“)
    my_function("Emil")
    my_function("Tobias")
    my_function("Linus")
    ```

- Number of Arguments
  - def my_function(fname, lname):
      print(fname + " " + lname)
    my_function("Emil", "Refsnes")
- Keyword Arguments
  - def my_function(child3, child2, child1):
      print("The youngest child is " + child3)
    my_function(child1 = "Emil", child2 = "Tobias", child3 = "Linus")
- Default parameter value
  - def my_function(**country = "Norway"**):
      print("I am from " + country)
    my_function("Sweden")
    my_function("India")
    my_function()
    my_function("Brazil")

- Passing a List as an Argument
  - You can send any data types of argument to a function (string, number, list, dictionary etc.), and it will be treated as the same data type inside the function.

  -
    ```
    def my_function(food):
      for x in food:
        print(x)
    fruits = ["apple", "banana", "cherry"]
    my_function(fruits)
    ```
- Returning the value

  -
    ```
    def my_function(x):
      return 5 * x
    print(my_function(3))
    print(my_function(5))
    print(my_function(9))
    ```

Recursion:

- Python also accepts function recursion, which means a defined function can call itself.
- Recursion is a common mathematical and programming concept. It means that a function calls itself. This has the benefit of meaning that you can loop through data to reach a result.
- The developer should be very careful with recursion as it can be quite easy to slip into writing a function which never terminates, or one that uses excess amounts of memory or processor power.

```
def tri_recursion(k):
  if(k > 0):
    result = k + tri_recursion(k - 1)
    print(result)
  else:
    result = 0
  return result
print("\n\nRecursion Example Results")
tri_recursion(6)
```

For Loops:

- A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).
- This is less like the for keyword in other programming languages, and works more like an iterator method as found in other object-orientated programming languages.

-
  ```
  fruits = ["apple", "banana", "cherry"]
  for x in fruits:
    print(x)
  ```
- Looping through a string:

  ```
  for x in "banana":
    print(x)
  ```

Break :

-
  ```
  fruits = ["apple", "banana", "cherry"]
  for x in fruits:
    print(x)
    if x == "banana":
      break
  ```

-
  ```
  fruits = ["apple", "banana", "cherry"]
  for x in fruits:
    if x == "banana":
      break
    print(x)
  ```

Continue:

-
  ```
  fruits = ["apple", "banana", "cherry"]
  for x in fruits:
    if x == "banana":
      continue
    print(x)
  ```
- The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and ends at a specified number.

  -
    ```
    for x in range(6):
      print(x)
    ```

  Note that range(6) is not the values of 0 to 6, but the values 0 to 5.

  -
    ```
    for x in range(2, 6):
      print(x)
    ```

```
#Else in for loop:
    for x in range(6):
        print(x)
    else:
        print("Finally finished!")
#If else in for loop:
    for x in range(6):
        if x == 3: break
        print(x)
    else:
        print("Finally finished!")
```

Exercise :

-
  ```
  *
  **
  ***
  ****
  *****
  ```

-
  ```
      *
     **
    ***
   ****
  *****
  ```

Nested Loops:

-
  ```
  adj = ["red", "big", "tasty"]
  fruits = ["apple", "banana", "cherry"]
  for x in adj:
    for y in fruits:
      print(x, y)
  ```

While Loop:

- With the while loop we can execute a set of statements as long as a condition is true.

-
  ```
  i = 1
  while i < 6:
    print(i)
    i += 1
  ```
- **Note:** remember to increment i, otherwise else the loop will continue forever.

```
#With Break statement:
    i = 1
    while i < 6:
        print(i)
        if i == 3:
            break
        i += 1
#With Continue Statement:
    i = 0
    while i < 6:
        i += 1
        if i == 3:
            continue
        print(i)
```

While loop using else statement:

-
  ```
  i = 1
  while i < 6:
    print(i)
    i += 1
  else:
   print("i is no longer less than 6")
  ```