Expressions, Conditional Statements & For loop

Overview & Purpose

In this session, participants will learn about expressions, operator precedence and associativity. We'll be introducing conditional statements, learning how to effectively use if, elif, and else to guide our code's decision-making. Advanced data structures: tuples, sets and dictionaries, learning how to effectively use and iterate through each.

Objectives

- Evaluating expressions: operator precedence and associativity
- Introduction to conditional statements: if, elif, and else
- Executing code based on conditional statements.
- Using loops to iterate through the data structures in Python: lists, tuples, sets and dictionaries etc
- Perform basic operations on data structures, such as adding, modifying, and accessing elements using loops.

1. Evaluating Expressions: Operator Precedence and Associativity

The operator precedence in Python is listed in the following table:

Data Types	**Class**
()	Parenthesis
**	Exponent
Positive *x*, negative *x*	Unary plus, Unary minus
*, /, //, %	Multiplication, Division, Floor division, Modulus
+, -	Addition, Subtraction
>, >=, <, <=	Comparison Operators
==, !=	Equality Operators
=, %= , /= , //= ,-= ,+= ,*= ,**=	Assignment Operators
not	Logical NOT
and	Logical AND
or	Logical OR

Examples:

```
 x = 2**3 + 3 * 4 / 10
print(x)
```

Output: 9.2

```
 y = 13 == 5 and (not10)> 12
print(y)
```

Output: False

```
 z = (5 % 2) + 30 <= 31 or True and False
print(z)
```

Output: True

2. Introduction to Conditional Statements: if, elif, and else

Python **if** statement is used for decision-making operations. It contains a body of code which runs only when the condition given in the **if** statement is true. If the condition is false, then the optional **else** statement runs which contains some code for the else condition.

The **else** keyword catches anything which isn't caught by the preceding conditions.

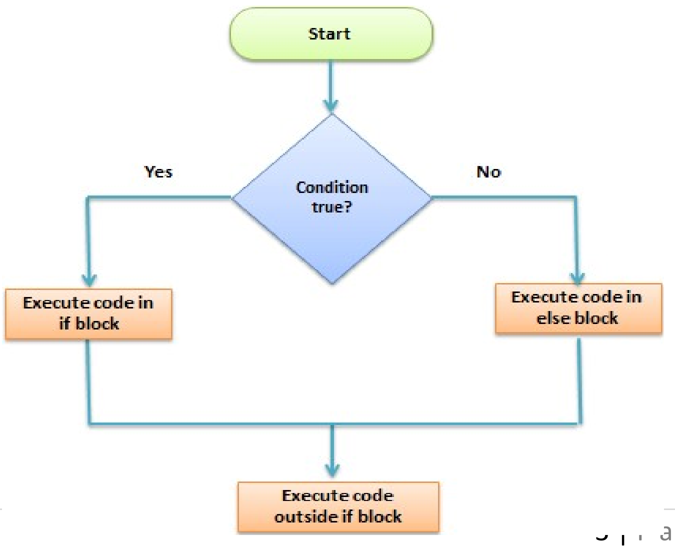
Syntax:

if expression:

Statement

Else expression:

Statement

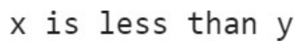


Example:

```
 x = 10
 y = 20


 if x < y:
   print("x is less than y")
else:
   print("x is greater than or equal to y")
```

Output



```
x is less than y
```

Elif Statement:

The **“elif”** keyword is Python’s way of saying “if the previous conditions were not true then try this condition”. For example:

Example:

```
    a = 33

    b = 33

    if b > a:

      print("b is greate than a")
   elif a == b:

        print("a and b are equal")
```

Output:

```
a and b are equal
```

Nested If:

You can have if statements inside if statements, this is called *nested* if statements.

```
x = 41

  if x > 10:
 print("Above ten,")
 if x > 20:

       print("and also above 20!")
 else:

         print("but not above 20.")
```

Output:

```
Above ten,
and also above 20!
```

3. Understanding the Flow of Control in Conditional Statements

It's a step-by-step process, like following a recipe. Depending on certain conditions, you might skip a step or repeat it. This flow decides how your program behaves.

3.1 Iteration using the for loop.

A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string). With the for loop, we can execute a set of statements, once for each item in a list, tuple, set etc.

Example:

```
 #iterating over a list
numbers = [8, 12, 30, 14, 6]
for i in numbers:

    print(i)
```

Output:

```
8
12
30
14
6
```

Example:

```
 # Iterating over a list

 fruit = ["apple", "mango", "orange"]
for i in fruit:

        print(i)
```

Output:

```
apple
mango
orange
```

Example:

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

Output:

```
b
a
n
a
n
a
```

Range() Function:

Python range() is a built-in function that is used to generate a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and stops before a specified number.

Syntax: range*(start, stop, step*)

Start: Optional. An integer number specifying at which position to start. Default is 0. Stop: Required. An integer number specifying at which position to stop (not included). Step: Optional. An integer number specifying the incrementation. Default is 1

Example:

```
 x = range(0, 5)
for n in x:
    print(n, end = ' ')
```

Output:

```
0 1 2 3 4
```

Example:

```
 x = range(3, 20, 2)
for n in x:

    print(n, end = ' ')
```

Output:

```
3 5 7 9 11 13 15 17 19
```

Example: The code sums up numbers from 1 to 10.

```
sum = 0

for num in range(1,11):

    sum += num

print(sum)
```

Output: 55

3.2 Indentation

For a code you write to be executed right, all the conditions must be met, including proper indentation. This is particularly important when it comes to loops and iteration. For instance, to loop five lines of code using the for loop, all the five lines must be indented by four spaces or one tab about the beginning of the loop statement.

This is vital programming practice no matter what language you use. Python requires that you properly indent your code, otherwise your program will encounter errors.

Example:

```
fruits = [“mango”, apple”, “kiwi”, “banana”]
for j in fruits:
    print(j)
```

Practice Qestions:

1. Given a list of temperatures for a week, write a program that tells you how many days were above 25°C.
2. Create a program that accepts an age from the user. If the age is below 13, print "Child". If it's between 13 and 19, print "Teenager". Otherwise, print "Adult".
3. Use a for loop and the range() function to display following sequences:
    a. [-20, -17, -14, -11, -8, -5, -2, 1, 4, 7, 10, 13, 16, 19, 22, 25, 28]
    b. [110, 112, 114, 116, 118, 120, 122, 124, 126, 128]
    c. [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
    d. [-20, -23, -26, -29]
4. A student obtains the following grades in his courses. Make a list of his scores and write a for loop to sum up his total score. Also, calculate his percentage.

Courses	**Scores out of 100**
Mathematics	78
Programming	82
Biology	65
Physics	80
Chemistry	50

Next, write the code to assign grades to the student according to the following table:

Percentage	**Grade**
90-100	A
80-89	B+
60-79	B
40-59	C
39 and below	F

5. Using a for loop, sum up even numbers from 2 to 50.