Python Programming

Flow of control

Agenda :

- Flow of control
- Selection
- Indentation
- Repetition
- Break and continue
- Nested loops
- Exception Handling

Flow of control :

The order of execution of the statements in a program is known as flow of control.

The flow of control can be implemented using control structures. Python supports two types of control structures—selection and repetition.

Selection :

In Programming concept of decision making or selection is implemented with the help of if..else statement.

The syntax of if statement is:

```
if (condition):
        statement(s)
Eg - if age >= 18:
                print("Eligible to vote")
```

A variant of if statement called if..else statement allows us to write two alternative paths and the control condition determines which path gets executed.

The syntax for if..else statement is as follows.

```
if condition:
        statement(s)
else:
        statement(s)
```

If there is a situations that require multiple conditions to be checked and it may lead to many alternatives. In such cases we can chain the conditions using if..elif (elif means else..if).

The syntax for a selection structure using elif is as shown below.

```
if condition:
        statement(s)
elif condition:
        statement(s)
elif condition:
        statement(s)
else:
        statement(s)
```

Indentation :

Python uses indentation for block as well as for nested block structures. Leading whitespace (spaces and tabs) at the beginning of a statement is called indentation.

In Python, the same level of indentation associates statements into a single block of code.

The interpreter checks indentation levels very strictly and throws up syntax errors if indentation is not correct.

It is a common practice to use a single tab for each level of indentation.

Repetition :

Repetition of a set of statements in a program is made possible using looping constructs.

The For loop :

The for statement is used to iterate over a range of values or a sequence.

The for loop is executed for each of the items in the range.

These values can be either numeric, or, they can be elements of a data type like a string, list, or tuple.

Syntax of For loop :

for <control variable> in <sequence/items in range>:

    <statements inside the body of loop>

The Range() Function :

It is used to create a list containing a sequence of integers from the given start value upto stop value (excluding stop value), with a difference of the given step value.

range([start], stop[, step])

The ‘While’ Loop :

The while statement executes a block of code repeatedly as long as the control condition of the loop is true.

After each iteration, the control condition is tested again and the loop continues as long as the condition remains true

Syntax : while test_condition:

body of while

Break and Continue Statements:

Break Statement :

The break statement alters the normal flow of execution as it terminates the current loop and resumes execution of the statement following that loop.

Continue Statement:

When a continue statement is encountered, the control skips the execution of remaining statements inside the body of the loop for the current iteration and jumps to the beginning of the loop for the next iteration.

Nested Loops:

A loop may contain another loop inside it. A loop inside another loop is called a nested loop.

Any type of loop (for/while) may be nested within another loop (for/while).

No restriction on how many loops can be nested inside a loop.

Eg-

```
for i in range(1,num + 1):
    for j in range(1,i + 1):
        print(j, end = " ")
```

Thank You