Tasks Assigned today

18-02-2025
Suneetha.V

Control Flow Statements:

These are used to control the flow of execution in a program, based on certain conditions or repeatedly executing a block of code. This category includes:

- Conditional statements: `if`, `elif`, `else`, `nested if`.
- Looping statements: `for`, `while`.
- Loop control statements: `break`, `continue`.

1. `if` Statement

The `if` statement allows you to execute a block of code only if a condition is `True`.

```
if condition:
    # Code to execute if condition is True

example:
age = 18
if age >= 18:
print("You are an adult.")
```

2. `else` Statement

The else statement is used to execute a block of code when the condition in the if statement is False.

```
if condition:
    # Code if condition is True
else:
    # Code if condition is False
```

Tasks Assigned today

18-02-2025
Suneetha.V

Example:

```
age = 16
if age >= 18:
print("You are an adult.")
else:
print("You are a minor.")
```

3. elif (else if) Statement

The elif statement allows you to check multiple conditions. It stands for "else if" and is used when you have more than two possibilities.

```
if condition1:
    # Code if condition1 is True
elif condition2:
    # Code if condition 2 is True
else:
    # Code if none of the above conditions are True
```

Example:

```
age = 20
if age < 13:
print("Child")
elif age < 18:
print("Teenager")
 else:
print("Adult")
```

4. Nested Conditional Statements

Tasks Assigned today

18-02-2025
Suneetha.V

You can also nest if, elif, and else statements inside each other.

```
if condition1:
    if condition2:
        # Code if both conditions are True
    else:
        # Code if condition1 is True but condition2 is False
else:
    # Code if condition1 is False
```

Example:

```
age = 25
is_student = True

if age >= 18:
    if is_student:
        print("Adult and a student.")
    else:
        print("Adult but not a student.")
else:
    print("Minor.")
```

1. for Loop

The for loop is used to iterate over a sequence (like a list, tuple, dictionary, string, or range). It will execute the block of code once for each item in the sequence.

Syntax:

```
for variable in sequence:
    # Code to execute for each item in sequence
```

Example:

Tasks Assigned today

18-02-2025
Suneetha.V

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

for i in range(5): # Loop from 0 to 4
print(i)
```

2. while Loop

The while loop repeatedly executes a block of code as long as a condition is True. It is typically used when the number of iterations is not known beforehand and depends on the condition.

Syntax:

```
while condition:
    # Code to execute while condition is True
```

Example:

```
python
Copy
count = 0
while count < 5:
    print(count)
    count += 1  # Increment the count to avoid infinite loop
```

Tasks Assigned today

18-02-2025
Suneetha.V

1. break Statement

The break statement is used to terminate the loop prematurely, regardless of whether the loop condition is true or not.

Example:

```
for i in range(10):
    if i == 5:
        break  # Exit the loop when i is 5
    print(i)
```

2. continue Statement

The continue statement skips the current iteration of the loop and proceeds with the next iteration.

Example:

```
for i in range(5):
    if i == 3:
        continue  # Skip the iteration when i is 3
    print(i)
```