Python

TECHNOLOGY

Conditions and Loops

Learning Objectives

By the end of this lesson, you will be able to:

- Define if Conditions
- Explain for loop, while loop, do-while loop

TECHNOLOGY

Conditional Statements

if Conditions

In Python, **if** is a conditional statement that is used to execute a block of code only when a certain condition is met. Here is the basic syntax of an if statement in Python:

```
if condition:
    # code to be executed when the condition is True
```

Here is an example of using an if statement to check if a number is positive:

```
num = 5
if num > 0:
   print("The number is positive")
```

If-else Conditions

We can use the **if** statement with an **else** block to execute different blocks of code depending on whether the condition is True or False.

Here is an example of using an if-else statement to check if a number is even or odd:

```
num = 5
if num % 2 == 0:
   print("The number is even")
else:
   print("The number is odd")
```

if-elif else Conditions

We can use the **if** statement with an **elif** block to execute different blocks of code depending on multiple conditions.

Here is an example of using an **if-elif-else** statement to check the grade of a student:

```
score = 80
if score >= 90:
     print("A")
elif score >= 80:
     print("B")
elif score >= 70:
     print("C")
else:
     print("F")
```

Nested if Conditions

You can use **nested if** statements to test multiple conditions within a single if statement. A **nested if** statement is an if statement inside another if statement. Here's the basic syntax of a **nested if** statement:

```
if condition1:
        # code to execute if condition1 is true
        if condition2:
                # code to execute if condition2 is true
```

Nested if Conditions

Here's an example of a nested if statement in Python:

```
x = 10
y = 5
if x > 0:
    if y > 0:
        print("both x and y are positive")
    else:
        print("x is positive but y is not")
else:
    print("x is not positive")
```

For Loop

For Loop

For loop is used to iterate over a sequence of elements. The sequence can be a list, tuple, string, or any other iterable object.

Here's an example of a **for** loop in Python:

```
fruits = ["apple", "banana", "cherry"]
for fruit in fruits:
    print(fruit)
```

TECHNOLOGY

While Loop

While Loop

A while loop is used to repeatedly execute a block of code as long as a condition is satisfied.

Here's an example of a while loop in Python:

```
x = 0
while x < 5:
    print(x)
    x += 1
```

While Loop

You can also use the break statement to exit a while loop prematurely, or the continue statement to skip over certain iterations of the loop.

Here's an example that uses both statements:

```
x = 0
while x < 10:
    if x == 5:
        x += 1
        continue
    if x == 8:
        break
    print(x)
    x += 1
```

do-while Loop

do-while Loop

Python does not have a built-in do-while loop like some other programming languages do. However, you can simulate a do-while loop using a while loop with a conditional statement at the end.

Here's an example:

```
x = 0
while True:
    print(x)
    x += 1
    if x >= 5:
        break
```

Key Takeaways

- if statement is used for conditional execution.
- Nested if statements to test multiple conditions within a single if statement.
- Python does not have a built-in do-while loop like some other programming languages do.