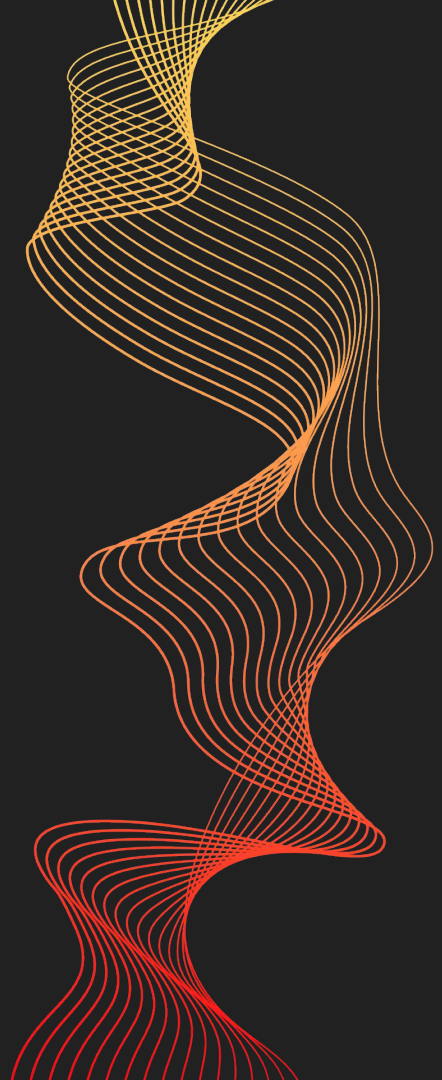
Control Statements

In Python

- 01 ALSTON ALVARES

Introduction to Control Statements in Python

This presentation provides an overview of control statements in Python and how they can be used to control the flow of a program.



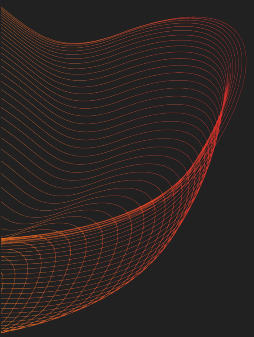
- 01 ALSTON ALVARES

What are Control Statements?

Control statements are used to alter the flow of a program based on certain conditions.

They allow for better decision making and looping in Python.

There are three main types of control statements in Python: if-else, for, and while.



- 01 ALSTON ALVARES

If-Else Statements

If-else statements allow you to execute different code blocks based on certain conditions.

The if statement checks a condition and executes a block of code if the condition is true.

The else statement provides an alternate block of code to execute if the condition is false.

- 01 ALSTON ALVARES

If else example

Code:

```
number = 10

if number > 10:
    print('The number is greater than 10.')
else:
    print('The number is less than or equal to 10.')
```

Output:

```
= RESTART: C:\Users\alsto\OneDrive\Documents\python'\5_7.py
The number is less than or equal to 10.
```

For Loops

- For loops are used to iterate over a sequence of elements.
- They allow you to perform a set of actions for each element in the sequence.
- You can use the range() function to define the number of iterations.

For Loop example

Code:

```
for i in range(10):
    print(i)
```

Output:

```
= RESTART: C:\Users\alsto\OneDrive\Documents\python'\5_7.py
0
1
2
3
4
5
6
7
8
9
```

While Loops

01 Be careful to avoid infinite loops by ensuring the condition eventually becomes false.

02 While loops continue to execute a block of code as long as a certain condition is true.

03 They are useful when you don't know the number of iterations in advance.

- 01 ALSTON ALVARES

While Loop example

Code:

```
i = 0
while i < 10:
    print(i)
    i += 1
```

Output:

```
= RESTART: C:\Users\alsto\OneDrive\Documents\python'\5_7.py
0
1
2
3
4
5
6
7
8
9
```

Break Statement

The `break` statement in Python is used to terminate the current loop and resume execution at the next statement immediately after the end of that loop. It can be used in both while and for loops, and if it is inside a nested loop, it will terminate the innermost loop.

- 01 ALSTON ALVARES

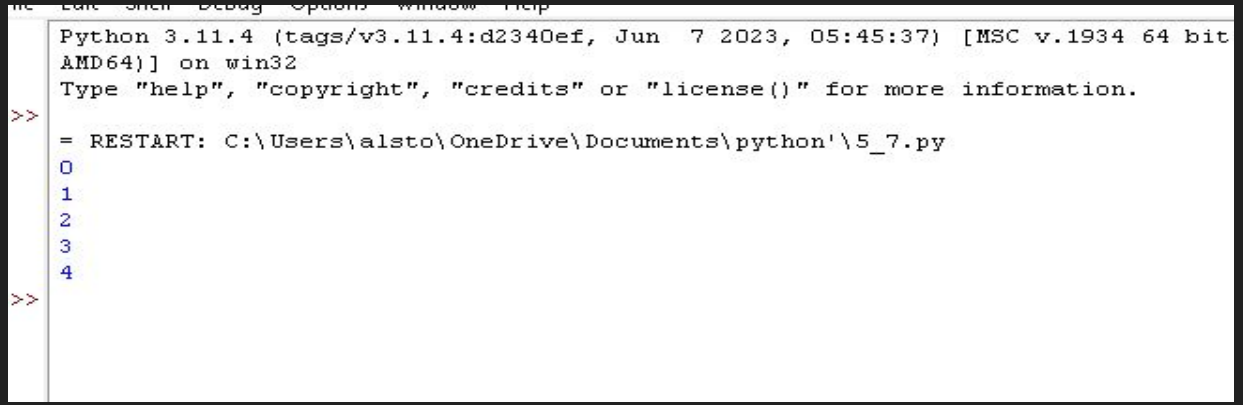
Break statement example

- 01 ALSTON ALVARES

Code:

```
for i in range(10):
    if i == 5:
        break
    print(i)
```

Output:



Pass Statement

01 The pass statement can be a useful tool for writing incomplete code

02 It allows you to create a working program even if you haven't yet implemented all of the functionality.

03 The pass statement can also be used as a placeholder for code that you plan to implement later.

- 01 ALSTON ALVARES

Pass statement example

Code:

```
def factorial(n):
    if n == 0:
        return 1
    else:
        # This is a pass statement. It does nothing.
        pass

print(factorial(5))
```

Output:

```
= RESTART: C:\Users\alsto\OneDrive\Documents\python'\5_7.py
None
```

Continue Statement

- The `continue` statement is used to skip the rest of the current iteration of a loop

- This can be useful for cases where you want to process all of the items in a sequence, but you want to skip some of them.

- 01 ALSTON ALVARES

Continue statement example

Code:

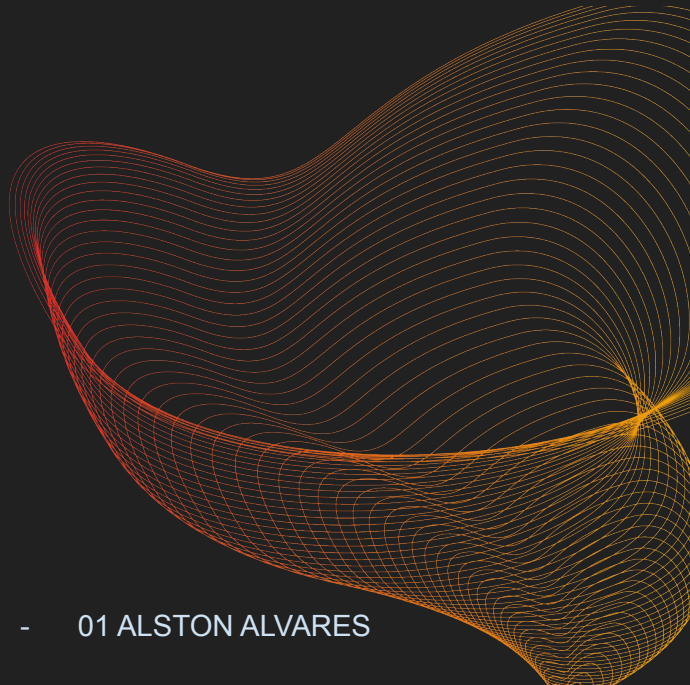
```
for i in range(10):
    if i % 2 == 0:
        continue
    print(i)
```

Output:

```
>>
= RESTART: C:\Users\alsto\OneDrive\Documents\python'\5_7.py
1
3
5
7
9
>>
```

Topics covered in this presentation.

- Control Statements
- If-else Statements
- For Loops
- While Loops
- Break Statements
- Pass Statements
- Continue Statements



- 01 ALSTON ALVARES

Thank you. Please feel free to ask any questions.

😄

- 01 ALSTON ALVARES