

Python Flow Control - Notes

What is Flow Control?

Flow control determines the order in which individual statements, instructions or function calls are executed or evaluated.

```
Includes: - Conditional statements - Loop control statements (break, continue, pass) - if-elif-else logic - Loop else blocks
```

Conditional Statements

```
if, elif, else
```

```
x = 10

if x > 0:
    print("Positive")

elif x == 0:
    print("Zero")

else:
    print("Negative")
```

Conditions use comparison (==, !=, <, >, etc.) and logical (and, or, not) operators

Loop Control Statements

break

Immediately exits the nearest enclosing loop

```
for i in range(5):
    if i == 3:
        break
    print(i)
```

continue

Skips the rest of the current iteration and continues with the next

```
for i in range(5):
   if i == 2:
        continue
   print(i)
```

pass

Does nothing. Used when a statement is syntactically required but no action is needed.

```
for i in range(3):
    pass # Placeholder for future code
```

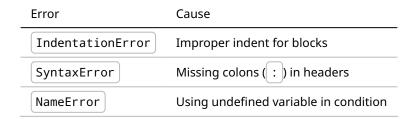
else with Loops

Executes only if the loop wasn't exited with break

```
for i in range(3):
    print(i)
else:
    print("Loop completed without break")
```

```
for i in range(3):
    if i == 1:
        break
    print(i)
else:
    print("This won't run")
```

Common Errors



Best Practices

- Always end if , elif , and else with a colon (:)
 Keep conditions readable; avoid deeply nested logic
- Use pass to indicate intentional no-op
- Combine conditions using and / or instead of nesting