Python Assignment- 3

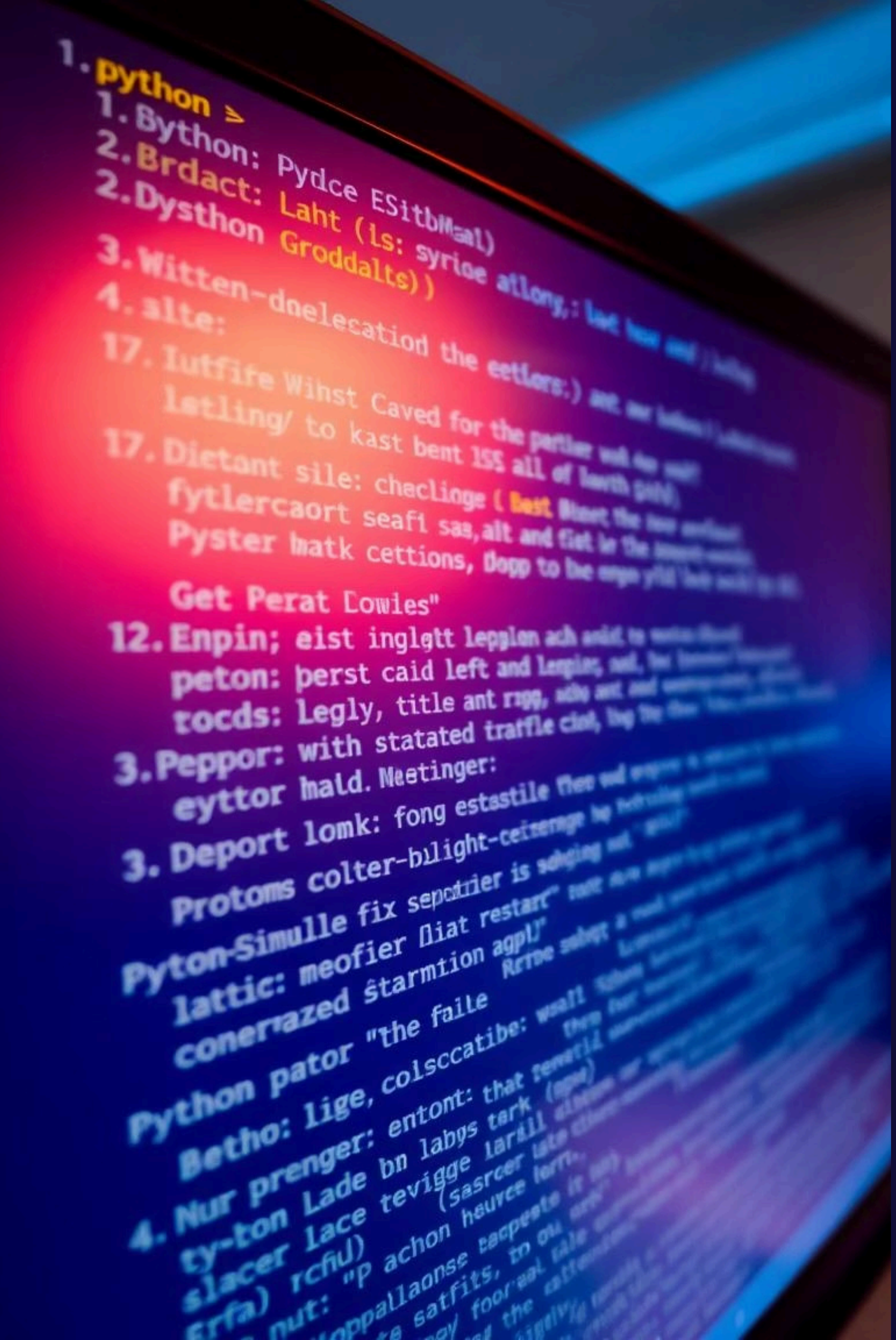
conditional statement

NAME-KIRTAN DESAI VIPULBHAI

ROLL NO- 025

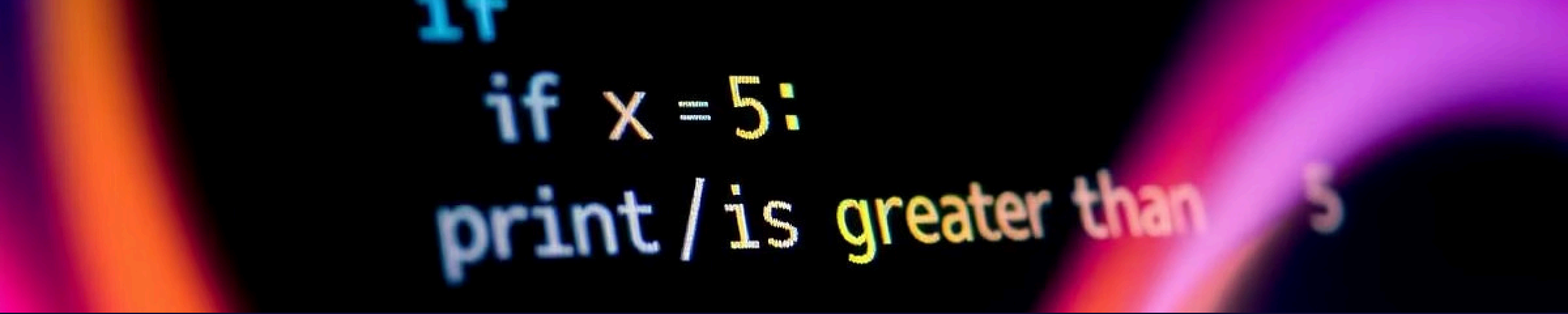
ENROLLMENT NO- 2402030430065

CLASS- 4CE-C



Conditional Statements in Python

This presentation covers conditional statements in Python. These statements allow programs to make decisions. We'll explore **if**, **if-else**, and **if-elif-else** structures. Learn how to control program flow with conditional logic.



The Basics of the **if** Statement

The **if** statement executes code based on a condition. The syntax is **if condition:** Code runs only if the condition is true. Indentation is crucial in Python. It defines the code block.

Syntax

if condition:

Execution

Code executes if the condition is **true**.

Example

Checking if a number is positive.

The **if-else** Statement

The **if-else** statement provides an alternative block. This block runs when the condition is false. It handles both true and false scenarios. Practical uses include determining even or odd numbers.

Syntax

if condition: else:

Alternative Block

Runs when the condition is **false**.

The **if-elif-else** Structure

This structure checks multiple conditions sequentially. The first true condition's block executes. If none are true, the **else** block runs. Simplifies complex decision-making. A grading system is a good example.

1

Condition 1

Evaluate the first condition

2

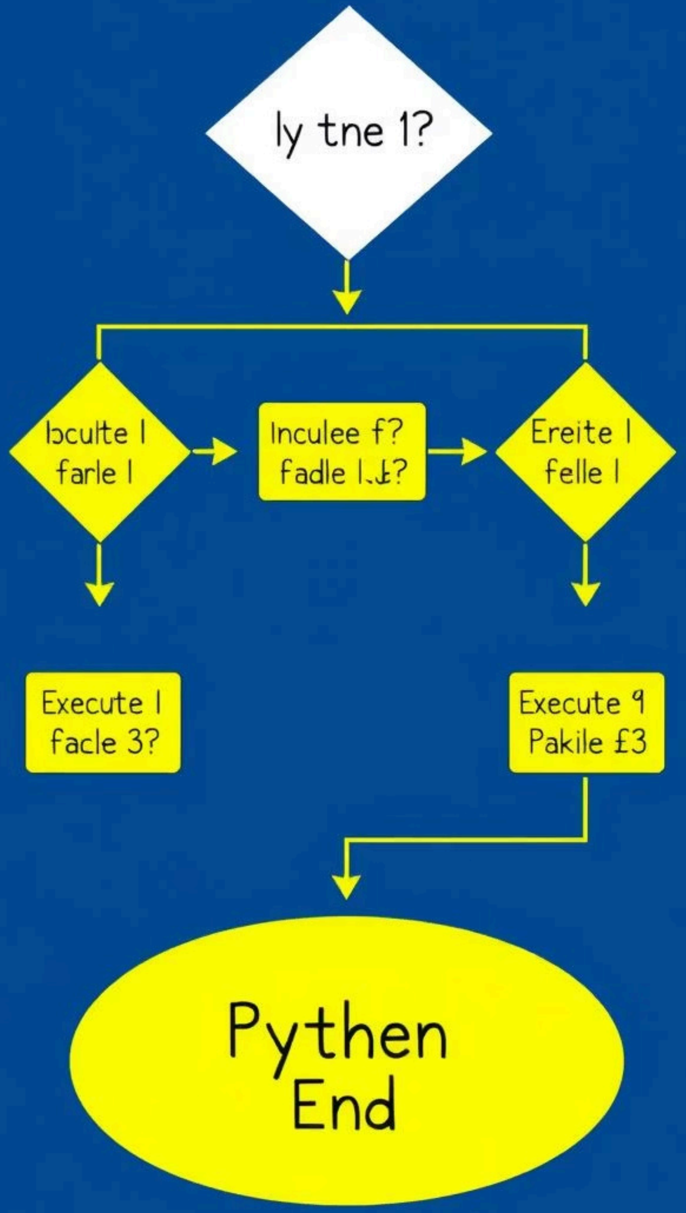
Condition 2

Check the next condition if the first is false.

3

Else

Execute this if all conditions are false.



Nested Conditional Statements

Nest conditionals within other conditionals. This allows for complex logic. Ensure readability with proper indentation. Common use cases include validating multiple criteria. For example, checking age and location.

Concept

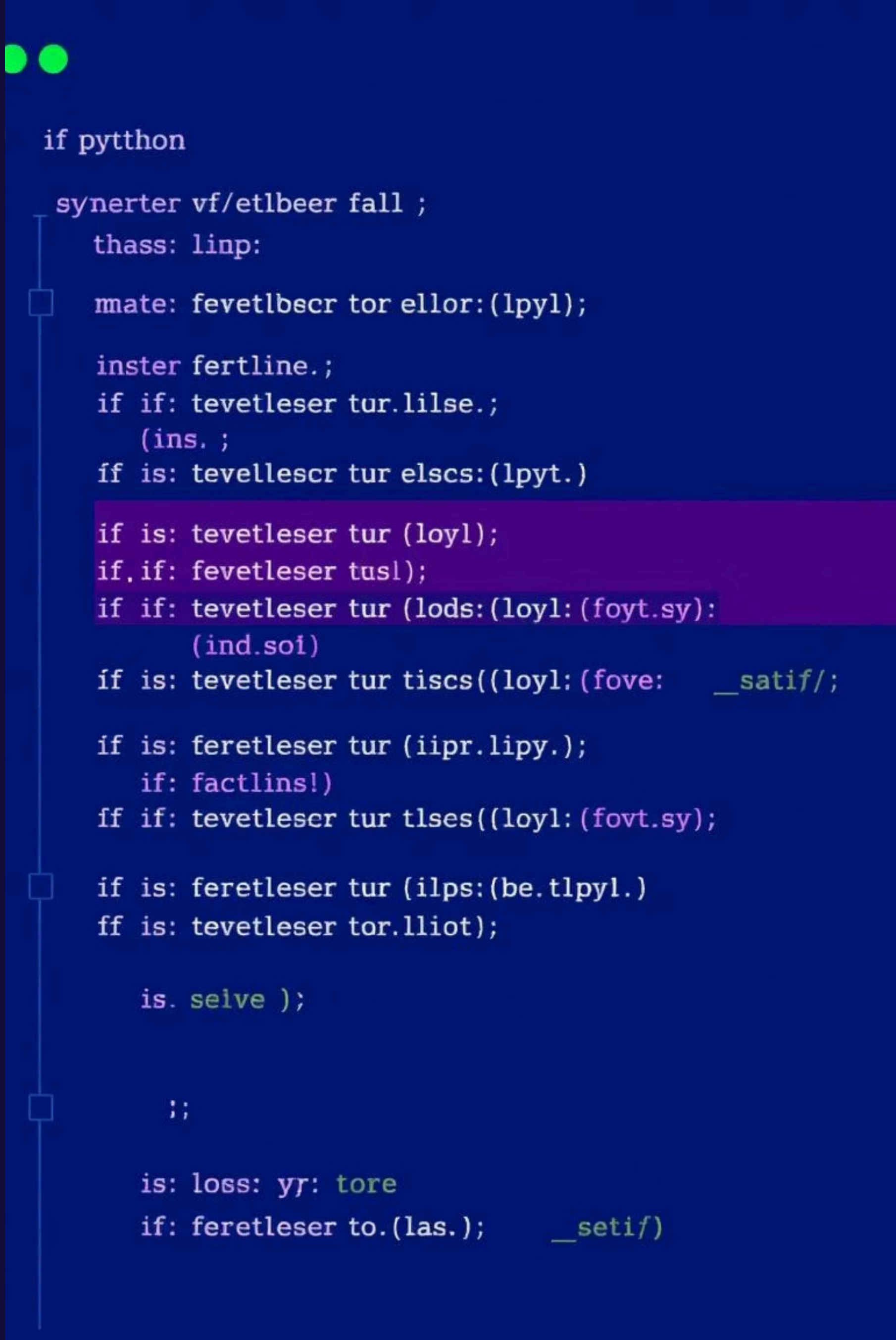
Conditionals within conditionals.

Example

Checking conditions within a block.

Readability

Maintain using indentation.



```
if pytthon

 synerter vf/etlbeer fall ;
    thass: linp:

    mate: fevetlbecr tor ellor:(lpyl);

    inster fertline.;
    if if: tevetleser tur.lilse.;
       (ins. ;
    if is: tevellescr tur elscs:(lpyt.)

    if is: tevetleser tur (loyl);
    if, if: fevetleser tusl);
    if if: tevetleser tur (lods:(loyl:(foyt.sy):
           (ind.soi)
    if is: tevetleser tur tiscs((loyl:(fove:    _satif/;

    if is: feretleser tur (iipr.lipy.);
       if: factlins!)
    fif if: tevetlescr tur tlses((loyl:(fovt.sy);

    if is: feretleser tur (ilps:(be.tlpyl.)
    ff is: tevetleser tor.lliot);

       is. selve );

         ;;

       is: loss: yr: tore
       if: feretleser to.(las.);     _setif)
```

Logical Operators in Conditions

and, **or**, and **not** combine conditions. **and** requires both conditions to be true. **or** requires at least one condition to be true. **not** negates a condition. Enhance flexibility of conditional checks.

and

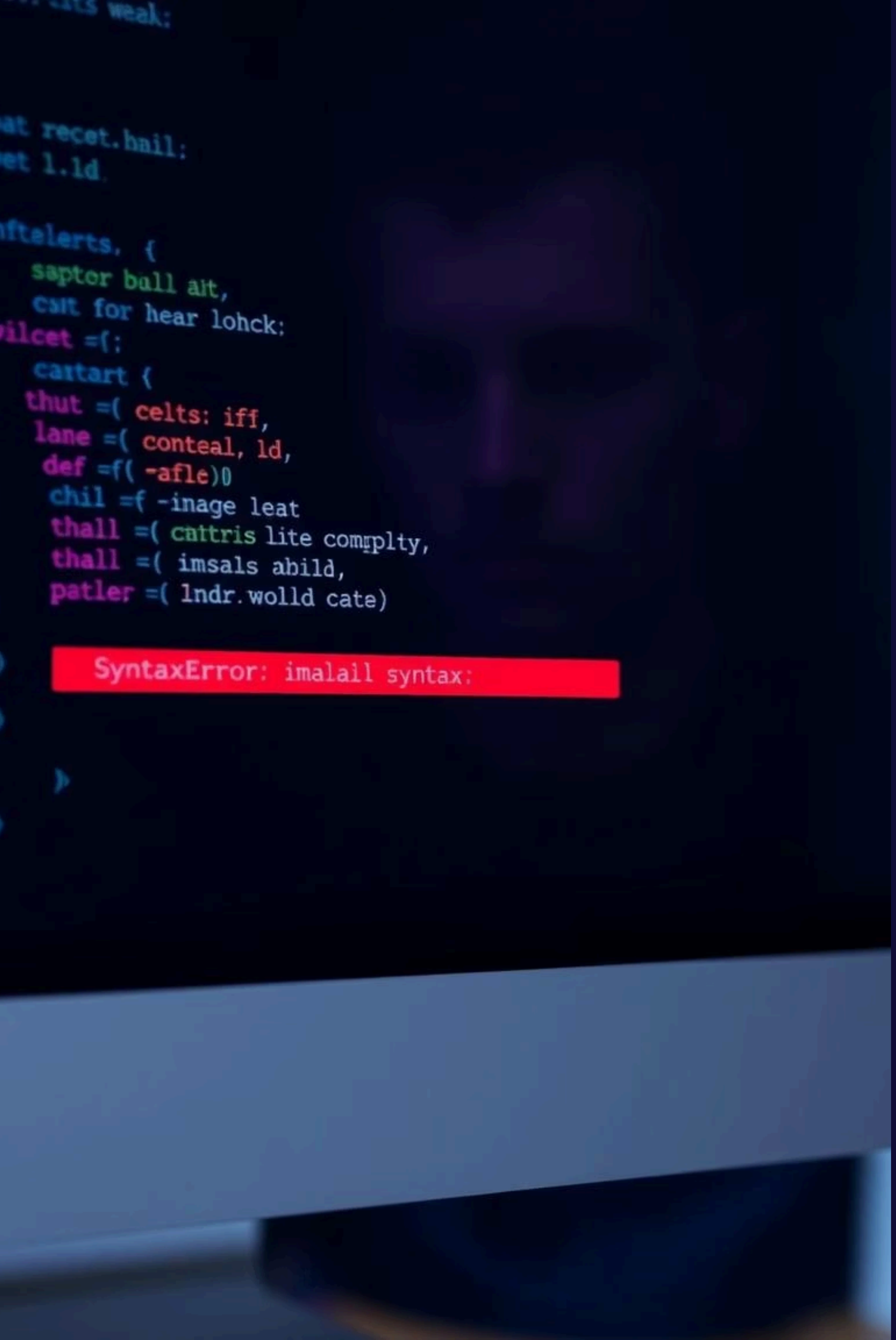
Both conditions true

or

At least one condition true

not

Negates a condition



```
its weak:

at recet.hnil:
et 1.ld

iftelerts, {
  saptor bull alt,
  cait for hear lohck:
ilcet ={:
  cartart {
 thut =( celts: iff,
  lane =( conteal, ld,
  def =f( -afle)0
   chil =f -image leat
   thall =( cattris lite complty,
   thall =( imsals abild,
   patler =( Indr.wolld cate)

  SyntaxError: imalail syntax:
```

Common Errors and Best Practices

Avoid indentation errors and infinite loops. Use comments for clarity. Don't make conditions overly complex. Thoroughly test edge cases. These practices improve code reliability. They also make debugging easier.

Comments

Add comments for code clarity.

Simplicity

Avoid complex conditions.

Testing

Test edge cases thoroughly.

Conclusion and Next Steps

Conditional logic is crucial in programming. Practice using **if**, **if-else**, and **if-elif-else** structures. Use Python documentation and tutorials. Write a program with multiple conditions for practice.

Review

if, **if-else**, **if-elif-else**

Practice

Write programs with conditions.

Resources

Use Python documentation.