Logical Operators

Logical operators in Python are used to combine or manipulate boolean values. The three main logical operators are `and`, `or`, and `not`. They are often used to create more complex conditions by combining simpler conditions. Here are the details along with examples:

1. `and` Operator:
- The `and` operator returns `True` if both conditions on the left and right are `True`. Otherwise, it returns `False`.

   ```python
   x = 5
   y = 10

   result = (x < 10) and (y > 5)
   print(result)  # Output: True
   ```

2. `or` Operator:
- The `or` operator returns `True` if at least one of the conditions on the left or right is `True`. If both are `False`, it returns `False`.

   ```python
   x = 5
   y = 3

   result = (x > 4) or (y < 2)
   print(result)  # Output: True
   ```

3. `not` Operator:
- The `not` operator is a unary operator that returns the opposite boolean value. If the condition is `True`, `not` returns `False`, and vice versa.

   ```python
   x = True

   result = not x
   print(result)  # Output: False
   ```

Examples with Conditionals:

Logical operators are often used in conjunction with conditional statements to control the flow of the program based on multiple conditions:

```python
age = 25
income = 50000

if age > 18 and income > 30000:
    print("You qualify for the loan.")
else:
    print("You do not qualify for the loan.")
```

In this example, the `and` operator is used to check if both the age is greater than 18 and the income is greater than 30000. If both conditions are true, the person qualifies for the loan.

```python
day = "Saturday"

if day == "Saturday" or day == "Sunday":
    print("It's the weekend!")
else:
    print("It's a weekday.")
```

In this example, the `or` operator is used to check if the day is either Saturday or Sunday, indicating whether it's the weekend.

Logical operators provide a way to create more complex conditions by combining simpler conditions, making it possible to express a wide range of decision-making scenarios in Python.