Chapter 6 – Conditional Expressions.

In Python programming, we often need to make decisions based on certain conditions. Conditional expressions help us execute instructions when specific conditions are met. These are achieved using `if`, `else`, and `elif` statements.

If, Else, and Elif Statements in Python

`if`, `else`, and `elif` statements allow the program to make multiway decisions based on conditions.

Syntax:

```
if condition1:  # if condition1 is True
    print("yes")
elif condition2:  # if condition2 is True
    print("no")
else:  # otherwise
    print("maybe")
```

Example:

```
a = 22
if a > 9:
    print("greater")
else:
    print("lesser")
# Output: greater
```

Quick Quiz: Write a program to print "yes" when the age entered by the user is greater than or equal to 18.

Solution:

```python
age = int(input("Enter your age: "))
if age >= 18:
    print("yes")
else:
    print("no")
```

Relational Operators

Relational operators are used to evaluate conditions inside `if` statements. Some examples include:

- **`==`: equals**
- **`>=`: greater than or equal to**
- **`<=`: lesser than or equal to**

Logical Operators

Logical operators operate on conditional statements. Examples include:

- **`and`: true if both operands are true, else false**
- **`or`: true if at least one operand is true, else false**
- **`not`: inverts true to false and false to true**

Elif Clause

The `elif` clause in Python means "else if." It allows us to chain multiple conditions together. The ladder will stop once a condition in an `if` or `elif` statement is met.

Syntax:

```
if condition1:
    # code
elif condition2:
    # code
elif condition3:
    # code
else:
    # code
```

Important Notes:

1. **There can be any number of `elif` statements.**
2. **The last `else` is executed only if all the conditions inside `elif` statements fail.**

Examples of Operators in Python

1. Relational Operators

Relational operators are used to compare values and return a boolean result (`True` or `False`). Here are some basic examples:

- **Equal to (`==`):**

```
a = 10
b = 10
print(a == b)  # Output: True, because 10 is equal to 10
```

Not equal to (`!=`):

```
a = 10
b = 5
print(a != b)  # Output: True, because 10 is not equal to 5
```

Greater than (>):

```
a = 15
b = 10
print(a > b)   # Output: True, because 15 is greater than 10
```

Less than (<):

```
a = 5
b = 10
print(a < b)   # Output: True, because 5 is less than 10
```

Greater than or equal to (>=):

```
a = 10
b = 10
print(a >= b)  # Output: True, because 10 is equal to 10
```

Less than or equal to (<=):

```
a = 5
b = 10
print(a <= b)  # Output: True, because 5 is less than or equal to 10
```

2. Logical Operators

Logical operators are used to combine conditional statements. Here are some basic examples:

- **Logical AND (and):**

```
a = 5
b = 10
print(a < 10 and b > 5)  # Output: True, because both conditions are true
```

Logical OR (`or`):

```
a = 5
b = 10
print(a > 10 or b < 15)  # Output: True, because at least one condition is true
```

Logical NOT (`not`):

```
a = 5
print(not a > 10)  # Output: True, because a > 10 is False, and not False is True
```

3. Conditional Expressions

Conditional expressions (ternary operator) provide a shorthand way to write `if-else` statements:

- **Conditional Expression:**

```
a = 5
result = "Even" if a % 2 == 0 else "Odd"
print(result)  # Output: "Odd", because 5 is not divisible by 2
```

These examples illustrate how to use relational and logical operators in Python to make decisions and perform comparisons.