| عنوان المحاضرة | المادة | المحاضرة | المرحلة | القسم |
|---|---|---|---|---|
| Condition statement | برمجة – لغة بايثون | الخامسة | الثانية | علوم الحاسبات |

# University of Anbar

## College of Computer Science and Information Technology

م.م. محمد عبدالرزاق رجب

# PYTHON PROGRAMMING LANGUAGE

# Lecture Five

## Conditional Statements

### 1. What are Conditional Statements?

- Conditional statements are programming constructs that allow a program to make decisions based on specific conditions. They enable the program to execute different code blocks depending on whether a given condition is **true** or **false**.

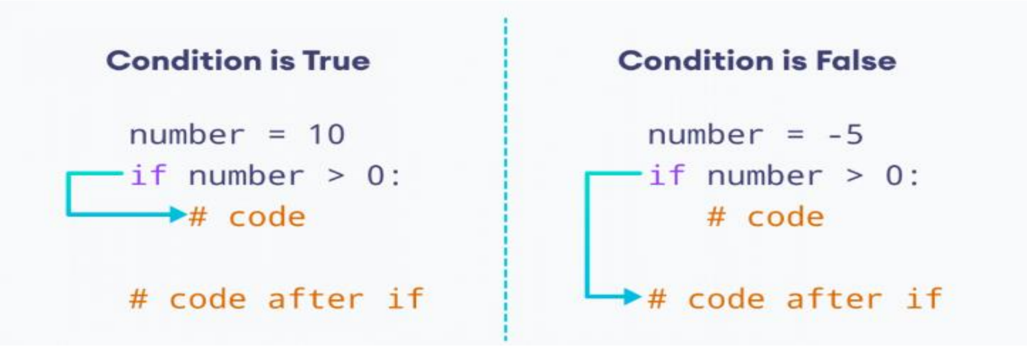
### 2. Types of Conditional Statements in Python

- Python offers several conditional statements to control the flow of execution based on specific conditions. The primary tools for managing conditions in Python are:

1. **if statement**
2. **if-else statement**
3. **if-elif-else statement**
4. **nested if statements**
5. **logical operators** (and, or, not)
6. **comparison operators** (==, !=, <, >, <=, >=)

### 1. `if` Statement

- The **if** statement is used to test a specific condition. If the condition evaluates to **True**, the block of code under if is executed.



**Condition is True**

```
number = 10
if number > 0:
    # code

# code after if
```

**Condition is False**

```
number = -5
if number > 0:
    # code

# code after if
```

```python
# Example: Check if a number is positive
number = 5

if number > 0:
    print("The number is positive.")

print("Program ended")
```

### 2. `if-else` Statement

- The **if-else** statement provides an alternative path of execution when the if condition is **False**.

```python
# Example: Check if a number is positive or negative
number = -3

if number > 0:
    print("The number is positive.")
else:
    print("The number is negative.")

print("Program ended")
```

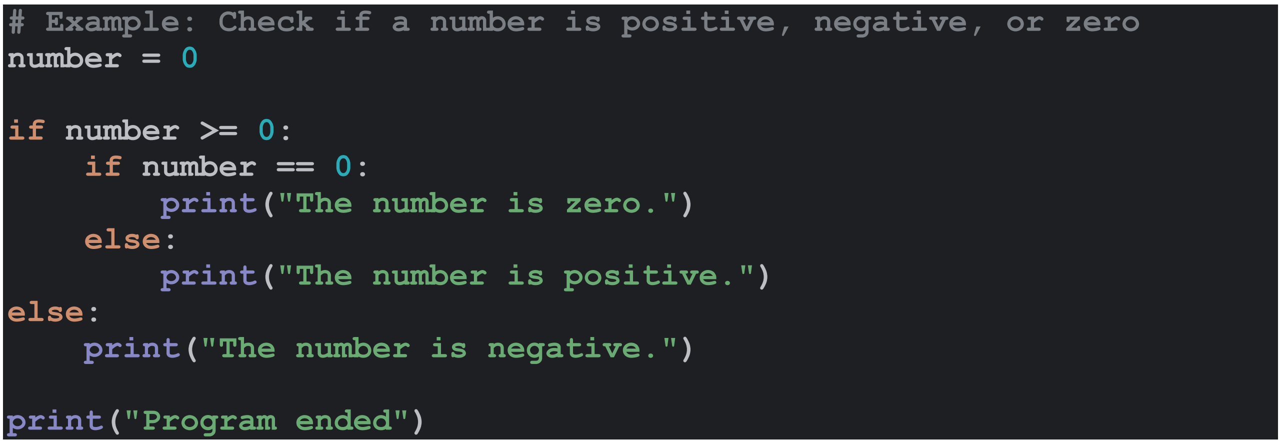
### 3. `if-elif-else` Statement

- The **if-elif-else** statement is used for multiple conditions. Only the first **True** condition's block will be executed.

```python
# Example: Grade calculation
score = 75

if score >= 90:
    print("Grade: A")
elif score >= 80:
    print("Grade: B")
elif score >= 70:
    print("Grade: C")
else:
    print("Grade: F")

print("Program ended")
```

### 4. Nested `if` Statements

- Nested if statements are **if** statements inside another **if** statement. They allow testing multiple conditions.

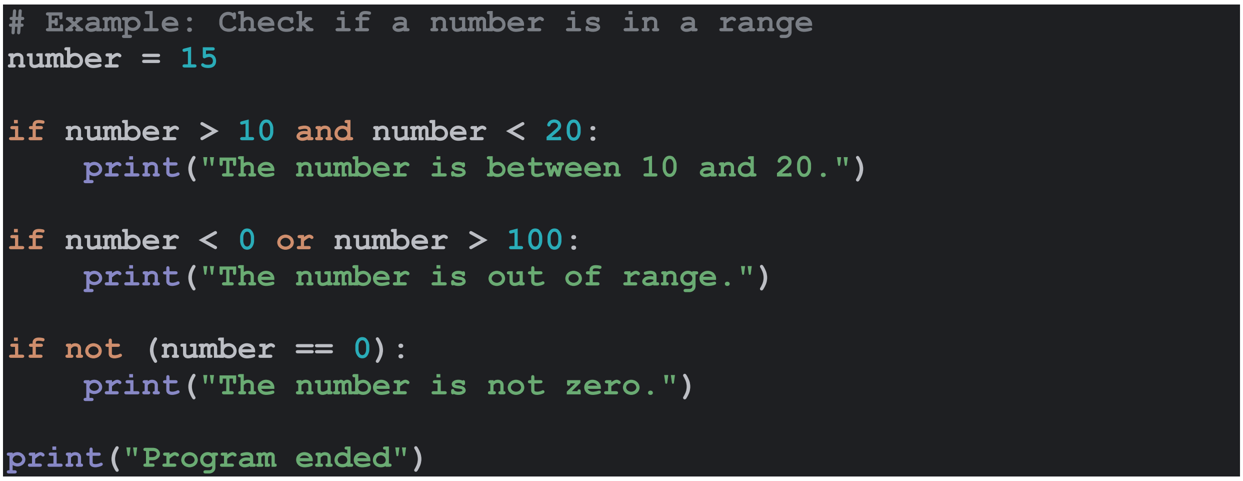


```python
# Example: Check if a number is positive, negative, or zero
number = 0

if number >= 0:
    if number == 0:
        print("The number is zero.")
    else:
        print("The number is positive.")
else:
    print("The number is negative.")

print("Program ended")
```

### 5. Logical Operators (and, or, not)

➢ Logical operators combine multiple conditions.

- **and**: Both conditions must be **True**.
- **or**: At least one condition must be **True**.
- **not**: Inverts the truth value of the condition.



```python
# Example: Check if a number is in a range
number = 15

if number > 10 and number < 20:
    print("The number is between 10 and 20.")

if number < 0 or number > 100:
    print("The number is out of range.")

if not (number == 0):
    print("The number is not zero.")

print("Program ended")
```

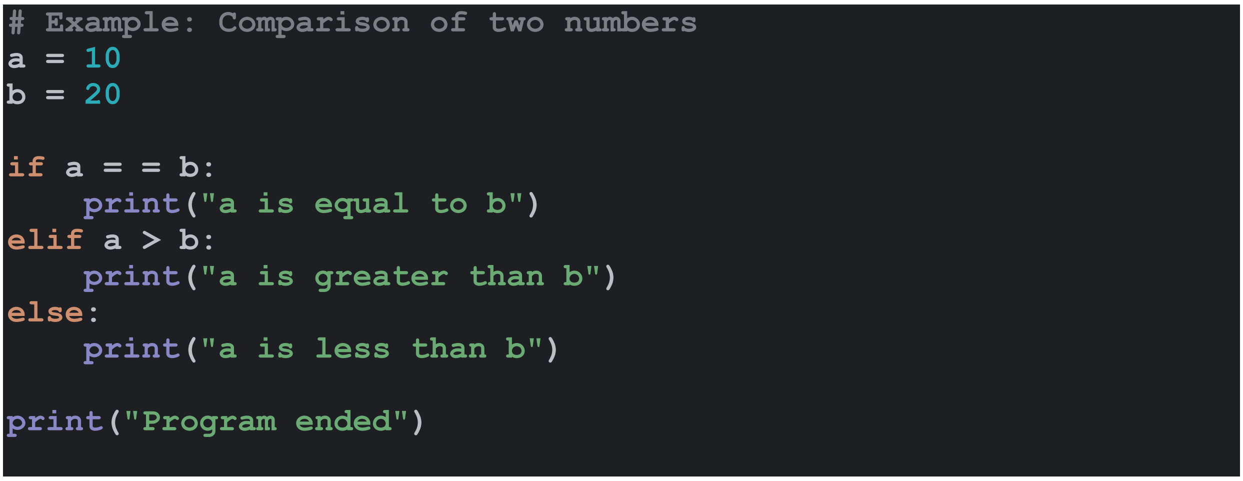
### 6. Comparison Operators

➢ Comparison operators are used to compare values. These include:

- **== (equal)**
- **!= (not equal)**
- **> (greater than)**
- **< (less than)**
- **>= (greater than or equal to)**

- **<= (less than or equal to)**

➢ These operators can be used in combination with conditional statements.

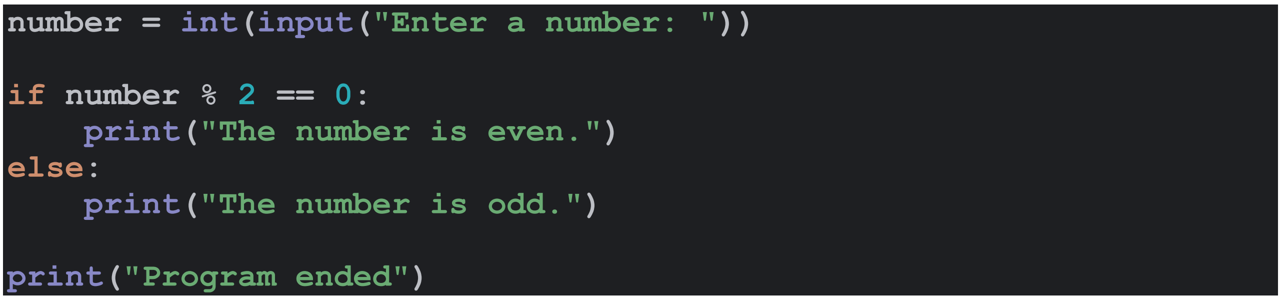


```
# Example: Comparison of two numbers
a = 10
b = 20

if a = = b:
    print("a is equal to b")
elif a > b:
    print("a is greater than b")
else:
    print("a is less than b")

print("Program ended")
```

➢ Python's conditional statements and operators allow you to control the flow of your program based on different conditions. They are essential for decision-making in Python programming.

**Example 1//** Write a Python program that prompts the user to enter a number. If the number is even, print "**The number is even**." Otherwise, print "**The number is odd**."

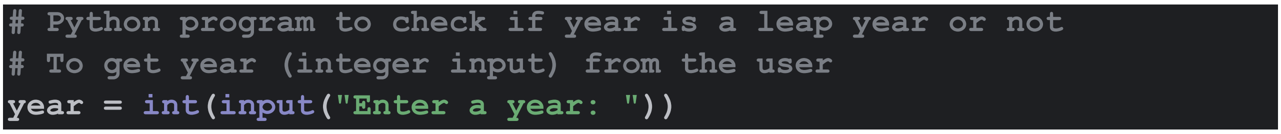
**Solution//**



```
number = int(input("Enter a number: "))

if number % 2 == 0:
    print("The number is even.")
else:
    print("The number is odd.")

print("Program ended")
```

**Example 2//** Write a Python program that determines whether a given year is a leap year. A leap year is exactly divisible by 4 except for century years (years ending with 00). The century year is a leap year only if it is perfectly divisible by 400.

**Solution//**



```
# Python program to check if year is a leap year or not
# To get year (integer input) from the user
year = int(input("Enter a year: "))
```

```
# divided by 100 means century year (ending with 00)
# century year divided by 400 is leap year
if (year % 400 == 0) and (year % 100 == 0):
    print("{0} is a leap year".format(year))

# not divided by 100 means not a century year
# year divided by 4 is a leap year
elif (year % 4 ==0) and (year % 100 != 0):
    print("{0} is a leap year".format(year))
# if not divided by both 400 (century year) and 4 (not century year)
# year is not leap year
else:
    print("{0} is not a leap year".format(year))

print("Program ended")
```

**# {0}:** This is a placeholder for a value that will be inserted into the string**.**

**Example 3//** Write a Python program that determines whether a given letter by a user is a vowel or a consonant.

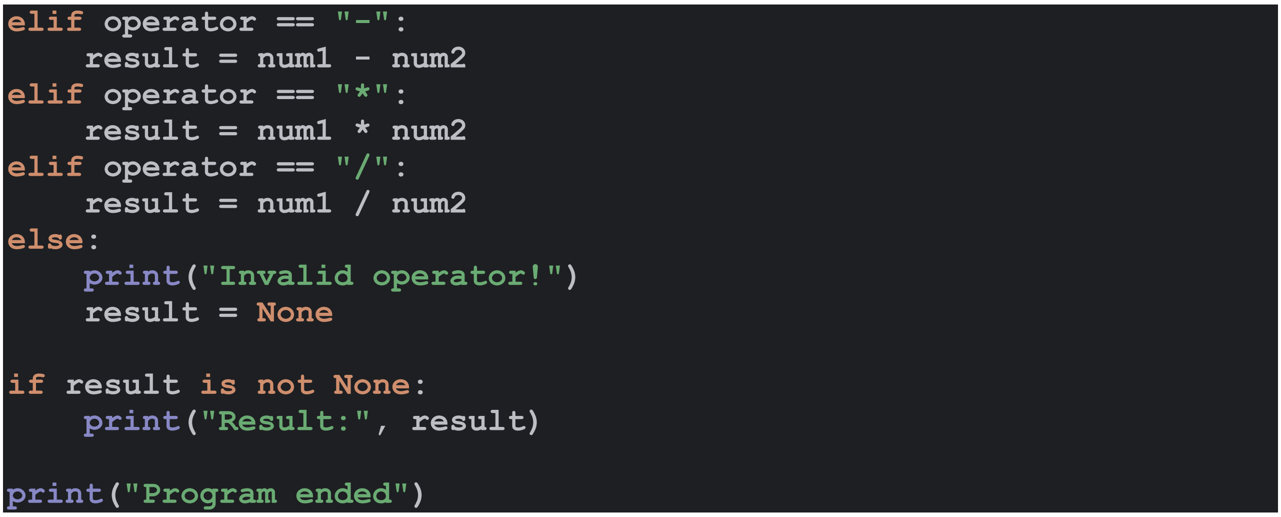
**Solution//**

```
letter = input("Enter a letter: ").lower()

vowels = "aeiou"

if letter in vowels:
    print(letter, "is a vowel.")
else:
    print(letter, "is a consonant.")

print("Program ended")
```

**Example 4//** Write a Python program that acts as a simple calculator, allowing users to perform basic arithmetic operations (addition, subtraction, multiplication, and division).

**Solution//**

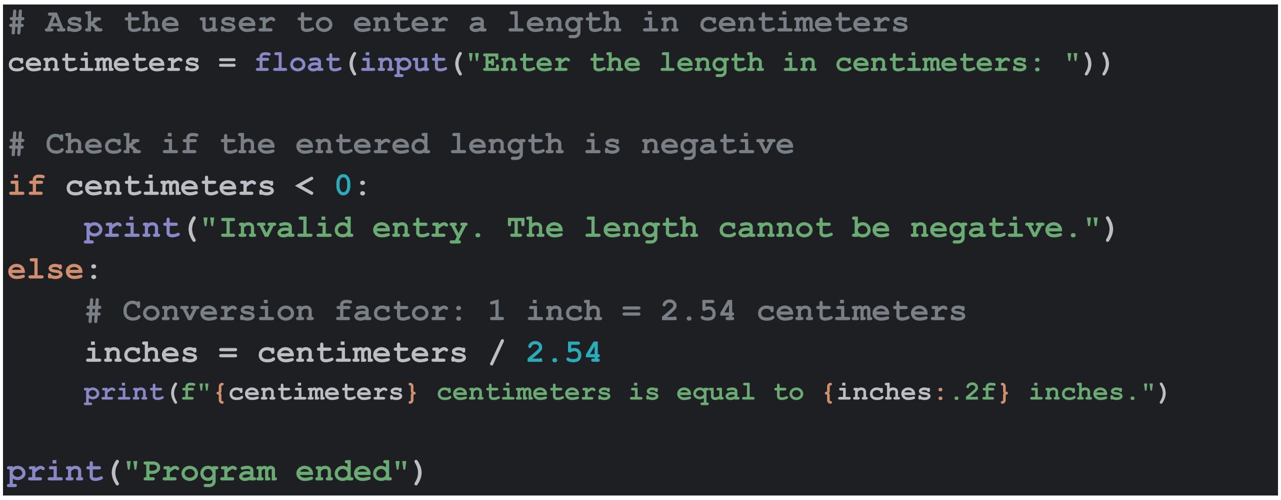
```
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
operator = input("Enter the operator (+, -, *, /): ")

if operator == "+":
    result = num1 + num2
```



```
elif operator == "-":
    result = num1 - num2
elif operator == "*":
    result = num1 * num2
elif operator == "/":
    result = num1 / num2
else:
    print("Invalid operator!")
    result = None

if result is not None:
    print("Result:", result)

print("Program ended")
```

**Example 5//** Write a program that asks the user to enter a length in centimeters. If the user enters a negative length, the program should tell the user that the entry is invalid. Otherwise, the program should convert the length to inches and print out the result. There are **2.54** centimeters in an inch.

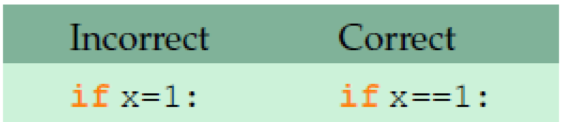
**Solution//**



```
# Ask the user to enter a length in centimeters
centimeters = float(input("Enter the length in centimeters: "))

# Check if the entered length is negative
if centimeters < 0:
    print("Invalid entry. The length cannot be negative.")
else:
    # Conversion factor: 1 inch = 2.54 centimeters
    inches = centimeters / 2.54
    print(f"{centimeters} centimeters is equal to {inches:.2f} inches.")

print("Program ended")
```

**Common Mistakes:**

- **Mistake 1:** The operator for equality consists of two equals signs. It is a really common error to forget one of the equals signs.



| Incorrect | Correct |
|---|---|
| `if x=1:` | `if x==1:` |

- **Mistake 2:** A common mistake is to use **and** where **or** is needed or vice-versa. Consider the following if statements:

```
if x>1 and x<100:
if x>1 or x<100:
```

The first statement is the correct one. If **x** is any value between **1** and **100**, then the statement will be true. The idea is that **x** has to be both greater than **1** and less than **100**. On the other hand, the second statement is not what we want because for it to be true, either **x** has to be greater than **1** or **x** has to be less than **100**. But every number satisfies this. The lesson here is if your program is not working correctly, check your **and**’s and **or**’s.

- **Mistake 3**: Another very common mistake is to write something like below:

```
if grade>=80 and <90:
```

This will lead to a syntax error. We have to be explicit. The correct statement is

```
if grade>=80 and grade<90:
```

On the other hand, there is a nice shortcut that does work in Python (though not in many other programming languages):

```
if 80<=grade<90:
```

**H.W//**

**Question 1:** Write a Python program that asks the user to enter their exam score. Based on the score, print the corresponding grade. Use the following grading scale:

- A: 90-100
- B: 80-89
- C: 70-79
- D: 60-69
- F: 0-59

**Question 2:** Given three numbers, write a Python code to find the Maximum of these two numbers.

**Question 3:** Write a python program to ask the user to enter their age and determine whether they are eligible to vote. In many places, the voting age is 18, so this program will use that as the threshold.