

- Checking a data type and type Conversion

Review of Lecture 2

- **Basic arithmetic** : + - * / // **
- **Priority of commands** : (), **, * and /, + and -
- **Variables**
 1. begin with a letter and can contain letters, digits and (_);
 2. Python is **case sensitive**.
 3. **Reserved keywords** cannot be used as an identifier
- **Creating and modifying variables**

```
length = 10
width = 5
length, width, height = 10, 5, 2
length1 = length2 = length3 = 5
speed = speed + 10
```
- **Data types** : int, float, str, bool, None

- **Formatted Output**

```
>>> print(f"Temperature = {temperature:.2f} ")
      Temperature = 20.57 C°
```

- **Getting data with input()**

```
age = int(input("Enter your age: "))
```

- **Input → Processing → Output**

- **Built-in functions**: abs(), max(), min(), pow(), round() .

- **math Module**:

```
import math
math.sqrt(25)
```

- **Comments**

```
temp = 25 # Temperature in degrees Celsius
```

Info 3

Introduction to Python

M. Bouzenita

2nd year Engineer - University of Jijel

Lecture 3

Conditional statements and Boolean logic

1. Introduction

```
>>> temperature = 25
>>> print(f"Temperature = {temperature} °C")
Temperature = 25 °C
```

The program displays the temperature regardless of its value.

However, many programs need to **make decisions**.

- display a **warning** if a temperature is too high;
- **classify a value** as **low**, **normal**, or **high**;
- determine whether a measurement is **within an acceptable range**;
- decide whether a condition is **satisfied** before performing an operation.

Python allows us to make such decisions using **conditional statements**.

2. The if Statement

The **if statement** allows a program to **execute an instruction** only when a condition is **true**.

if condition:
statement

the message
is not
displayed.

```
number = 20
if number > 0:
    print("Positive number")
```

Positive number

```
number = -3
if number > 0:
    print("Positive number")
```

3. Indentation

Indentation is an essential part of Python syntax.

The statements belonging to an if statement must be **indented**.

Python commonly uses **four spaces** for indentation.

```
if number > 0:  
    print("Positive number")  
    print(number)
```

4. The if-else Statement

Sometimes a program needs to **choose between two alternatives**

```
if condition:  
    statement_if_true  
else:  
    statement_if_false
```

```
temperature = 25  
  
if temperature > 35:  
    print("High temperature")  
else:  
    print("Temperature is acceptable")
```

Temperature is acceptable

4. The if-else Statement

Sometimes a program needs to **choose between two alternatives**

```
if condition:  
    statement_if_true  
else:  
    statement_if_false
```

```
temperature = 48  
  
if temperature > 35:  
    print("High temperature")  
else:  
    print("Temperature is acceptable")
```

High temperature

5. The if-elif-else Statement

The **if-elif-else** statement used when there are **more than two possible situations**.

```
if condition1:  
    statement1  
elif condition2:  
    statement2  
else:  
    statement3
```

```
score = 13  
  
if score >= 15:  
    print("Excellent")  
elif score >= 10:  
    print("Good")  
else:  
    print("Needs improvement")
```

Good

5. The if-elif-else Statement

The **if-elif-else** statement used when there are **more than two possible situations**.

```
if condition1:
    statement1
elif condition2:
    statement2
else:
    statement3
```

A program can contain **more than one elif**.

```
hour = 18
if hour < 12:
    print("Morning")
elif hour < 17:
    print("Afternoon")
elif hour < 20:
    print("Evening")
else:
    print("Night")
```

Evening

6. Logical operators

Sometimes we need **more than one condition**.

Python provides three logical operators:

and

```
if temp >= 20 and temp <= 40:  
    print("Temperature is within the desired range")
```

or

```
if score < 0 or score > 20:  
    print("Score is outside the normal range")
```

not

```
if not (speed > 80):  
    print("Speed is not high")
```

7. Nested conditional statements

A conditional statement can contain another conditional statement.

```
if temperature <= 30:  
    if pressure <= 100:  
        print("Conditions are acceptable")
```

Nested conditions are **useful when one decision depends on another**.

However, they should not be used when a simpler condition can express the same logic clearly.

For example, the previous program can also be written as:

```
if temperature <= 30 and pressure <= 100:  
    print("Conditions are acceptable")
```

8. Using Conditions with `input()`

```
score = 13

if score >= 15:
    print("Excellent")
elif score >= 10:
    print("Good")
else:
    print("Needs improvement")
```

8. Using Conditions with `input()`

In Chapter 2, we learned that `input()` returns a **string**.

Therefore, when numerical input is required, we need **type conversion**.

```
score = float(input("Enter your score:"))  
if score >= 15:  
    print("Excellent")  
elif score >= 10:  
    print("Good")  
else:  
    print("Needs improvement")
```

9. Common Beginner Mistakes

1. Using = Instead of ==

Incorrect:

```
if temperature = 30:  
    print("Temperature is 30")
```

Correct:

```
if temperature == 30:  
    print("Temperature is 30")
```

2. Forgetting the Colon

Incorrect:

```
if temperature > 30  
    print("High temperature")
```

Correct:

```
if temperature > 30:  
    print("High temperature")
```

9. Common Beginner Mistakes

3. Incorrect Indentation

Incorrect:

```
if temperature > 30:  
print("High temperature")
```

Correct:

```
if temperature > 30:  
    print("High temperature")
```

4. Forgetting Type Conversion

Incorrect:

```
temp = input("Enter temperature: ")  
  
if temp > 30:  
    print("High temperature")
```

The value returned by `input()` is a string.

Correct:

```
temp = float(input("Enter temperature: "))  
  
if temp > 30:  
    print("High temperature")
```

9. Common Beginner Mistakes

5. Incorrect Order of Conditions

Consider:

```
temperature = 50
```

```
if temperature > 25:  
    print("Normal")  
elif temperature > 45:  
    print("High")
```

The output will be always:

Normal

**The second condition is never reached because
`temperature > 20` is already True.**

A better arrangement is:

```
if temperature > 45:  
    print("High")  
elif temperature > 25:  
    print("Normal")
```

Practice