INTRODUCTION TO PYTHON - II

Basic Python Syntax

- **print() function:** Used to display output on the screen. [1]
  - It can accept multiple, comma-separated items to print at once. [2]
  - The **sep** parameter specifies the separator to use between multiple items. [3]
  - The **end** parameter specifies what character to print at the end of the line; it defaults to a newline. [4][4][4][4][4][4][4][4][4]

```Python
# Using sep and end
print("apple", "banana", "orange", sep=", ") # Output: apple, banana, orange
print("Hello", end=" ")
print("World!") # Output: Hello World!
```

- **input() function:** Pauses the program to accept input from the user. [5]**Important:** It always returns the user's input as a **string**[6], regardless of what is entered.

```Python
name = input("Enter your name: ")
print("Hello, " + name)
```

- **Comments:** Used to explain code and improve readability. [7]
  - **Single-line comments** start with a # symbol. [8]
  - **Multi-line comments** are enclosed within triple quotes (""" or '''). [9]

```Python
# This is a single-line comment.
x = 5

"""
This is a multi-line comment.
It can span several lines.
"""
y = 10
```

- **Indentation:** Python uses indentation to define blocks of code. [10] Correct indentation is mandatory for the code to run.

```Python
# The print statement is indented, so it's inside the if block.
if 5 > 2:
    print("Five is greater than two!")
```

Variables, Identifiers, and Keywords

- **Keywords:** Reserved words with special meanings in Python (e.g., if, for, while, True). [11]They **cannot** be used as variable names. [12]
- **Identifiers:** A name given to a programming element like a variable or function. [13]
- **Variables:** A specific type of identifier used as a container to store data values. [14]Their values can be changed. [15]

Working with Variables

- **Assignment:** Variables are created the moment you first assign a value to them. [16]

```python
# Assigning an integer to the variable 'age'
age = 25
# Assigning a string to the variable 'name'
name = "John"
```

- **Naming Rules:**
  - Must start with a letter (a-z, A-Z) or an underscore (_). [17]
  - Can contain letters, numbers (0-9), and underscores. [18]
  - Cannot contain spaces or special characters like -. [19]
  - Are **case-sensitive** (`age` and `Age` are different variables). [20]

```python
# Legal variable names
my_variable = 10
user_name = "Alice"
_value = 3.14

# Illegal variable names
# 2nd_value = 20 # Cannot start with a digit
# user-name = "Bob" # Cannot contain a hyphen
```

- **Multiple Assignment:**
  - You can assign multiple values to multiple variables in one line. [21]
  - You can assign one value to multiple variables. [22]

```python
# Assigning multiple values
x, y, z = 10, 20, 30

# Assigning one value to multiple variables
a = b = c = "Hello"
```

Data Types and Type Conversion

Data types define the kind of value a variable can hold. [23]You can check a variable's type using the **`type()`** function. [24]

- **Common Data Types:**
  - **Numbers:** **`int`** (whole numbers) [25], **`float`** (decimal numbers) [26], and **`complex`**. [27]
  - **String (`str`):** A sequence of characters. [28]
  - **Boolean (`bool`):** Represents **`True`** or **`False`**. [29292929]

```python
x = 10        # int
y = 3.14      # float
name = "Alice"  # str
is_valid = True # bool

print(type(name)) # Output: <class 'str'>
```

- **Handling Strings:**
  - To include quotes inside a string, use a **backslash (`\`)** as an escape character. [30]

    - **Escape Characters:** \n creates a newline, and \t creates a tab. [31313131]

```
print('She is Ana\'s sister.') # Output: She is Ana's sister.
print("First line\nSecond line") # Prints on two separate lines
```

- **Type Conversion (Typecasting):**
    - The process of converting a value from one data type to another using functions like **int()**, **float()**, and **str()**. [32]
    - This is crucial for converting user input, which is always a string. [33]

```
age_str = input("Enter your age: ") # e.g., user enters "30"
age_int = int(age_str) # Convert the string "30" to the integer 30
next_year = age_int + 1

print("Next year you will be " + str(next_year)) # Convert int back to str for printing
```