Chapter-3

GET READY TO CODE
Introducing Variables, Print and Input

Get ready to level up your coding skills with the power of Python! In this chapter, we'll uncover the magic of variables, printing, and input. Learn how to talk to the computer, store information in variables, and make your programs interactive by taking user input. Get set to write your first lines of code and witness the wonders of Python unfold before your eyes. Let's dive in and explore the exciting world of coding together!

VARIABLES & DATA TYPES

Let us understand the concept of Variables using your favourite **Pizza**.

Assume

- You are in Domino's Pizza store.
- You want to eat Pizza at home.
- You have ordered delicious "Cheese Mushroom" Pizza
- Your pizza is now perfectly baked.

Think

- What happens if the baker gives you the Pizza without box?
- Will you be able to carry it to home?
- What do you think you need?

To carry this delicious Pizza.
You need

Container

(In this case a Box)

Think it like this from computers perspective

- **Pizza is your data (or value)**
- **Box is your variable (or container)**

This box (variable) contains your pizza (value)

General Definition – A Variable is just like a container that contains something which can be accessed. For example

- Bottle is a variable for water
- Diary is a variable for your tasks
- Your report card is a variable for your result information

Technical Definition – Variables are the **names** you give to computer memory locations which are used to store values in a computer program.

In other words - Variable in computer programming language is just like a general container which contains information that can be

- Accessed
- Modified
- Replaced
- Named or Renamed

*"In computer science, a variable has a **name** and it contains a **value**"*

For Example – Store a value of 5 in computer's memory and retrieve it later. Your logical steps will be to

- Create a variable with appropriate name "Number"
- Store the value in it "Number = 5"
- Retrieve the value using variable name "print(Number)"

Code snippet in Python

```
Number = 5
print(Number)
```

Output

```
5
```

Important Facts:

- Every programming language uses variables for the same purpose of storing & retrieving value. The value of variable can be changed and thus the name ***"Variable"***.
- In any computer software or program, the information or data is stored in 2 ways
  - Either the data is already stored in the program Or,
  - The data comes from the user

Example of variable (Name) in which data is already stored

```
Name = "Rahul Hinduja"
```

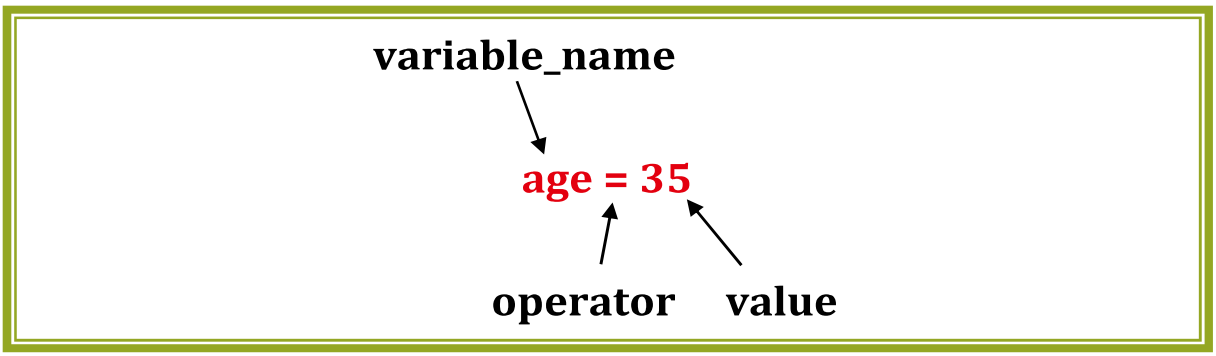
Example of variable (Name) in which data is asked from user and stored into a variable

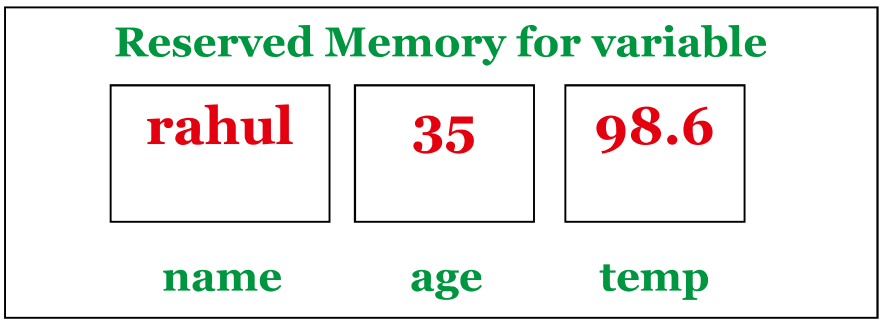
```
Name = input("Enter your name: ")
```

Variable Declaration & Data Type

Declaration of Variables

- The process of creating a variable is also known as declaration of variable.
- Declaration of variable does 2 things
  - It tells the computer what the variable name is i.e to give a name to computer memory location
  - It specifies what type of data the variable will hold i.e data type
  - Python do not require specifying data type at the start. During input, default is string in python





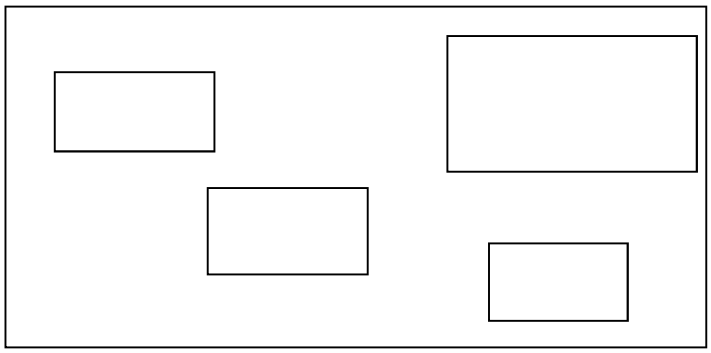
RAM (Memory)

Above memory address represents following variables which were declared

- name = "rahul" ⇨ variable name is **name**, value is **rahul** and type is **string**
- age = 35 ⇨ variable name is **age** and value is **35** and type is **int**
- temp = 98.6 ⇨ variable name is **temp** and value is **98.6** and type is **float**

Identify variables and values in the following

- City = “katni”
- state = “M.P”
- population = 1292042
- temperature = 36.5



Variable Conventions

Best Practice – There exists few conventions which are good to follow while declaring variable.

1. Variable names must **begin** with a letter, underscore, non-number

   For example
   a. age
   b. r123_456
   c. X
   d. y
   e. _temp
   f. _____
   g. _____
   h. _____
   i. _____
   j. _____

2. Python do not require specifying data type at the start. During input, default is string in python.

3. Do not use a comma with numbers.

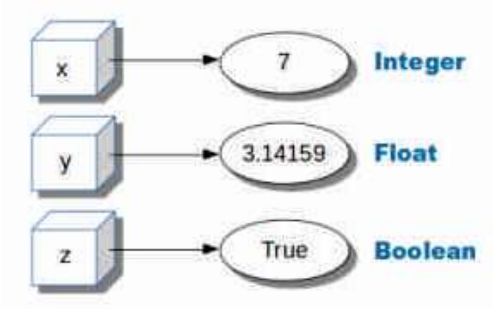
4. Once a data type is defined for the variable, then only that type of data can be stored in it. For example, if the variable is declared as Int, then it can only store integer values.

Data Types

As the name implies "Data Types" in programming language indicates the **type of data (value)** that you can store in a **variable.**

In python, a variable can store a *single type* of value. Usually these values are in the form of

- **Phrases** – Known as **String** type.
  - For example – "My name is Rahul Hinduja"
- **Numeric** – Known as **Number** type. Specifically, **Int** type or **Float** type.
  - For example – 6 or -32 are of int type and 23.6 or 2.5 is of float type
- **Truth Value** – Known as **Booleans** type.
  - For example – TRUE or FALSE



Boolean data type

- **Boolean** data type has the value which is either **TRUE** or **FALSE**
- **0 –** is represented by FALSE
- **1 –** is represented by TRUE
- These values are in-built and mostly used in conditional or loop statements.

Python Code

```
check = TRUE
if check:
   print("This is checked")
```

Output

```
This is checked
```

String data type

- String data type has the value which is string.
- **String** is **"A series of characters"**. Usually denoted within single quote ' ' or double quotes " "
- This is default data type in python when you input it from user.
  For example
    - Cityname = "Pune"
    - State_Name = 'Maharashtra'
    - pin_code = "411048"
- **Important Note** – Use either single quotes or double quotes to open and close the string.
    - **Correct_var1 = "This is string variable using double quote"**
    - **Correct_var2 = 'This is string variable using single quote'**
    - **Incorrect_var = ' This is going to error in the programming language"** **(Note it start with ' and ends with ")**

Create a variable name and assign a String value in it –

Numeric data type

- **Numeric** data type has the value which is a **number**.
- These numbers can be **positive** or **negative** in nature.
- There are 2 different numeric data types in python
    - **Int** (**plain integers**) – Any positive or negative **whole number** is called an integer (int) data type. For example,
        - Age = 35
        - your_score = -25
    - **Float** (**Floating point**) – Basically the number with a **decimal value** is called float data type. For example,
        - Fever = 100.4
        - result = 67.5
- **Important Note** – If your program asks the user to input a whole number then you should declare it with int or float data type explicitly.

For example,

age = int(input("Enter your age: ")) - This will accept "age = 35"

age = float(input("Enter your age:")) - This will accept "age = 35.5"

Create a variable name and assign a int or float value in it –

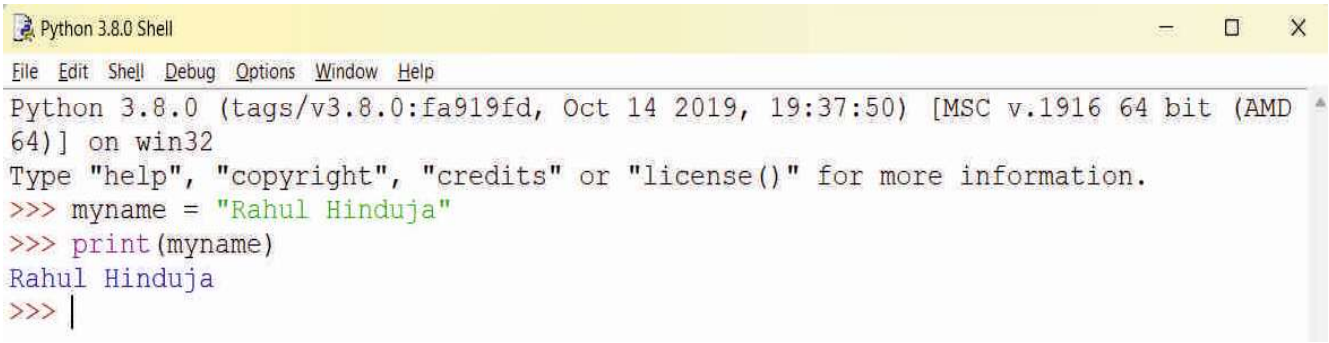
-
-
-
-
-
-
-

Exercise 1: Create a string variable with name myname and assign it with your real name.

Open IDLE and write below Python Code. Hit Enter Key

```
myname = “Rahul Hinduja”
print(myname)
```

IDLE Snippet



Code Explanation

1. The line **myname = "Rahul Hinduja"** creates a variable named `myname` and assigns it the value "Rahul Hinduja". Note that the double quotes “ “ are used to represt the string value at the start and the end.

2. The line `print(myname)` is used to display the value stored in the variable `myname`. The `print()` function is a built-in function in Python that allows us to output text or variable values to the console.

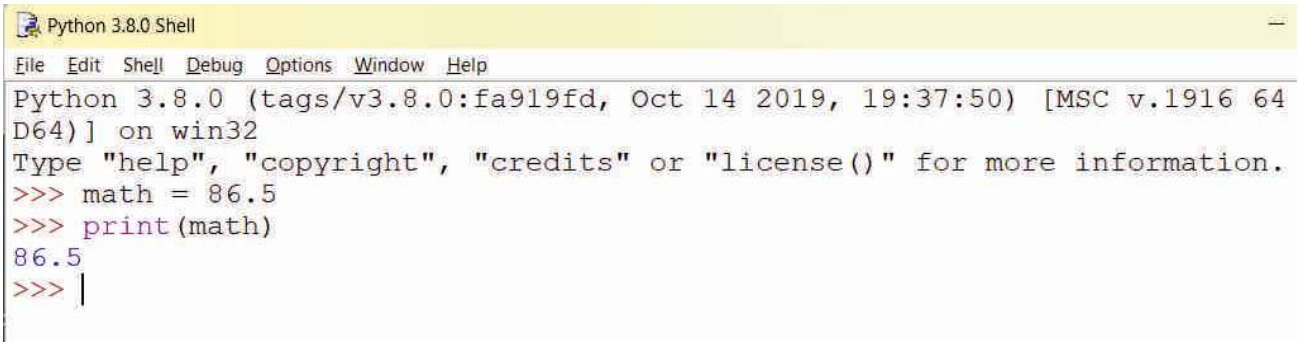
3. When we run this code, the output will be `Rahul Hinduja`, which is the value stored in the `myname` variable.

Exercise 2: Create an integer variable with name math and assign your last score from math exam.

Open IDLE and write below Python Code. Hit Enter Key

```
math = 86.5
print(math)
```

IDLE Snippet



Code Explanation

1. The line **math = 86.5** creates a variable named `math` and assigns it the float value 86.5. Note that the value 86.5 is a float data type value.

2. The line `print(math)` is used to display the value stored in the variable `math`. The `print()` function is a built-in function in Python that allows us to output text or variable values to the console.

3. When we run this code, the output will be `86.5`, which is the value stored in the `math` variable.

Exercise 3: What would be the output of the below python codes when executed. Provide explanation too.

Code 1: Python Code

```
monthly_expense = 12000
monthly_expense = 15000
print(monthly_expense)
```

Code Explanation

Code 2: Python Code

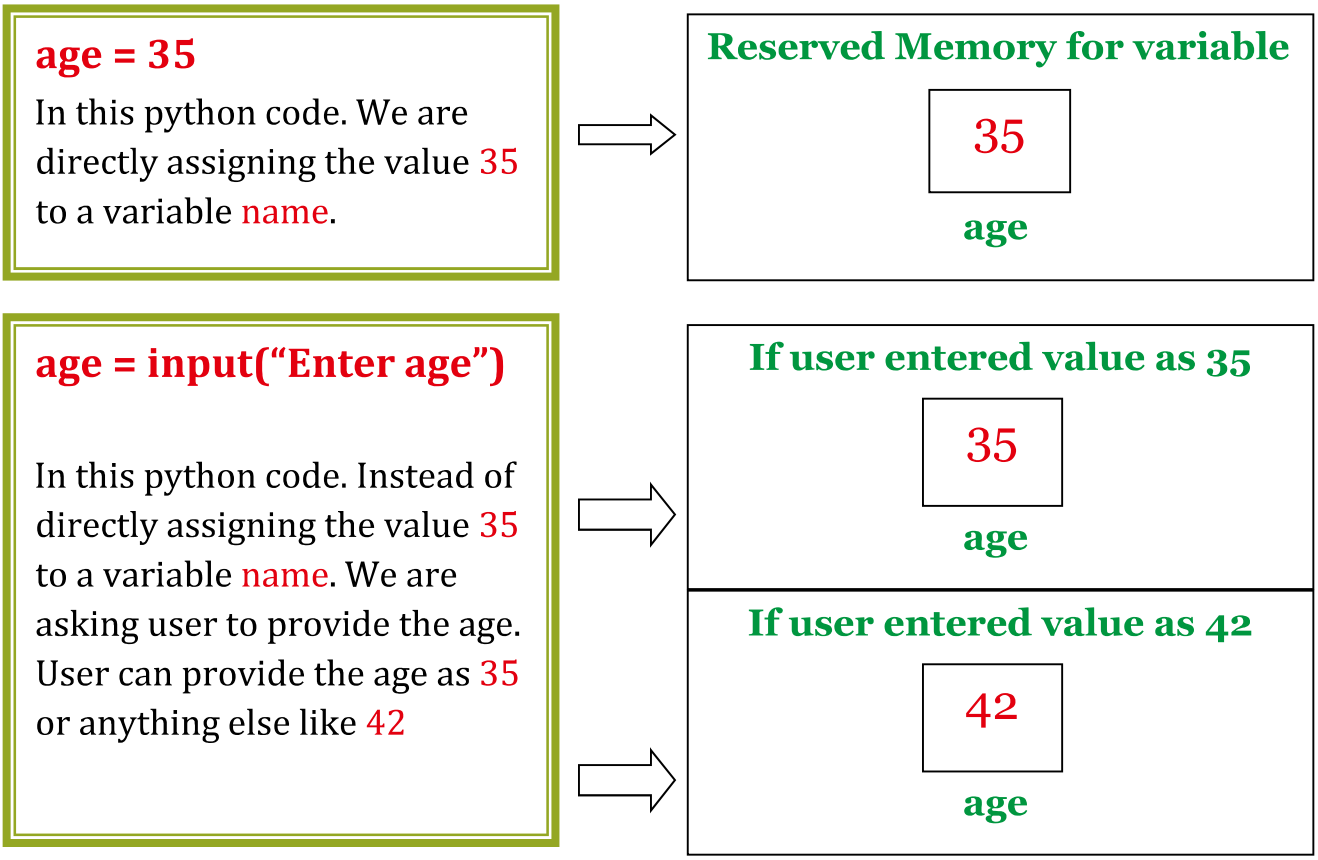
```
indian_currency = "Rupee'
print(Indian_currency)
```

Code Explanation

input( ) function

So far, you have used variables and directly assigned values to them. However, in real-life scenarios, we often need to get data or values from the user and store them in variables.

This is where the `input()` function comes in handy.



The `input()` function:

- Allows us to ask the user a question or prompt and wait for their response.
- Once the user provides input, we can store it in a variable and use it in our program.

Let's explore input( ) function with more examples:

Example 1: Asking for the user's name

```python
name = input("What's your name? ")
print("Hello, " + name + "!")
```

Output

```
What's your name? John
Hello, John!
```

Explanation:

In this example, we use the `input()` function to **ask** the user for their name. The input is stored as a value in the `name` variable, and then we print a greeting using the value stored in `name`.

Example 2: Getting the user's age

```python
age = input("How old are you? ")
print("You are " + age + " years old.")
```

Output

```
How old are you? 25
You are 25 years old.
```

Explanation: _____
_____
_____

Example 3: Calculating the sum of two numbers

```python
num1 = input("Enter the first number: ")
num2 = input("Enter the second number: ")
sum = int(num1) + int(num2)
print("The sum is:", sum)
```

Output

```yaml
Enter the first number: 5
Enter the second number: 7
The sum is: 12
```

Explanation: In this example, **we ask** the user for **two numbers** using the `input()` function. The inputs are stored in the variables `num1` and `num2`, and we use the `int()` function **to convert them** to integers. Then, we **calculate the sum** of the numbers and **print** the result.

Example 4: Getting the user's age and calculating their birth year

```python
current_year = 2023
age = input("How old are you? ")
birth_year = current_year - int(age)
print("You were born in", birth_year)
```

Output

```
How old are you? 25
You were born in 1998
```

Explanation: In this example, we ask the user for their age using the `input()` function. The input is stored in the `age` variable. We assume the current year is **2023** and **subtract** the user's age from it to **calculate** their **birth year**. The result is stored in the `birth_year` variable, and then we **print** it as part of a message.

Example 5: Comparing direct assignment and input( )

```python
# Direct assignment
favorite_color = "blue"
print("Your favorite color is", favorite_color)

# Using input()
favorite_color = input("What is your favorite color? ")
print("Your favorite color is", favorite_color)
```

Output

```csharp
Your favorite color is blue
What is your favorite color? red
Your favorite color is red
```

Explanation: In this example, we explore the difference between **directly assigning** a value to a variable and **using** the `input()` function to get a value from the user. In the first part, we directly assign the value **"blue"** to the `favorite_color` variable and **print** it. In the second part, we use the `input()` function to **ask** the user for their favorite color, store it in the `favorite_color` variable, and **print** it. Both methods allow us to store and display the user's favorite color, but the `input()` function provides the flexibility to change the value based on user input.

Note: In the above examples, you might have noticed some lines that start with a **# symbol**. **These lines are called comments**.

```
# Direct assignment
```

Comments are like notes that we can write in our code to help us or others understand what the code is doing. When the Python interpreter **runs our code**, **it completely ignores these comment lines**. They don't affect the execution or produce any output.

So, when you see a line starting with #, you can remember that it's a comment and it won't be executed as part of the program.

print ( ) function

Since the beginning of this book, we have used `print()` function numerous times.

It is one of the most used lines of code in Python. We use it everywhere.

The `print()` function in Python is like a **talking** machine that helps us **display** messages or **information** on the screen. It allows us to show messages, variables, and other data. We can use it to communicate with the computer and display results of our programs.

Example 1: Printing a simple message

```bash
print("Hello, world!")
```

Output

```
Hello, world!
```

Explanation: In this example, we use the `print()` function to display the message `"Hello, world!"` on the screen. Whatever is inside the parentheses of the `print()` function is printed as output.
Remember to either use a single(‘ ’) quotes or double (“ “) quotes. Do not mix them up.

✓ print(“Hello, world!”) <- Note double quote at the beginning and end

✓ print(‘Hello, world!’) <- Note single quote at the beginning and end

✗ print(“Hello, world!’) <- Note double quote at the beginning and single quote at the end

Example 2: Printing multiple lines of text

```bash
print("This is the first line.")
print("This is the second line.")
```

Output

```arduino
This is the first line.
This is the second line.
```

Explanation: In this example, we use the `print()` function twice to **display two separate lines of text**. Each `print()` statement is executed one after another, resulting in the output of both lines of text being displayed on separate lines.

Example 3: Printing variables

```makefile
name = "Alice"
age = 12
print("My name is", name, "and I am", age, "years old.")
```

Output

```csharp
My name is Alice and I am 12 years old.
```

Explanation: Here, we assign values to variables `name` and `age`. Then we use the `print()` function to display a message along with the values of these variables. The values are inserted into the message using commas.

Example 4: **Printing calculations**

```bash
num1 = 5
num2 = 3
sum = num1 + num2
print("The sum of", num1, "and", num2, "is", sum)
```

Output

```python
The sum of 5 and 3 is 8
```

Explanation: In this example, we perform a calculation by adding the values of `'num1'` and `'num2'` variables. We **store the result** in the new variable called `'sum'`. Then, using the `print()` function, we display a message that includes the values of `'num1'`, `'num2'`, and `'sum'`.

Summary

variable

- Variables in Python are used to store and manipulate data, allowing us to assign values and retrieve them later.

input( )

- The input( ) function function allows you to get input from the user during program execution.

print( )

- The print( ) function allows us to display output, such as text or variable values, providing visual feedback or information to the user or developer.

Practice Programs	Write a program that asks for the user's name and age, and then prints out a greeting message with their name and age.
	Create a program that calculates and prints the sum of two numbers entered by the user.
	Write a program that converts a given temperature in Celsius to Fahrenheit.
	Create a program that asks the user for their favorite color and prints it out with a message.
	Write a program that calculates the area of a rectangle with user-provided length and width.
	Create a program that takes two user-input numbers and swaps their values without using a third variable.