Python Basics – Notes

PYTHON OVERVIEW

Python is a high-level, interpreted, and dynamically typed programming language. It is widely used because its syntax is simple and close to the English language. Python allows developers to implement ideas quickly using fewer lines of code.

Example:

```
print('Hello, Python')
```

DATA TYPES

Data types define the kind of data a variable can store. Python automatically understands the data type based on the value assigned to the variable.

INTEGER (int)

An integer represents whole numbers without any decimal part. Integers can be positive, negative, or zero.

Example:

```
a = 10

b = 20

print(a + b)  # Output: 30
```

TYPE CONVERSION

Type conversion means changing one data type into another.

Explicit Conversion:

The user manually converts the data type using functions like int(), float(), or str().

Implicit Conversion:

Python automatically converts the data type when required.

Example:

```
integer_number = 12

float_number = 4.25

result = integer_number + float_number

print(result)
```

FLOAT

A float is a numeric data type that contains decimal values. It is commonly used for precise calculations.

Example:

```
x = 3.5

print(x)

print(type(x))
```

BOOLEAN (bool)

Boolean data type represents logical values. It can only have two values: True or False.

Example:

```
is_active = True

is_logged_in = False

print(is_active)

print(type(is_active))
```

STRING (str)

A string is a sequence of characters used to store text. Strings can be defined using single quotes, double quotes, or triple quotes.

Example:

```
name = "My name is Alice"

print(name)
```

Escape Characters Example:

```
sentence = "She said, \"It's a beautiful day!\""

print(sentence)
```

COMPLEX DATA TYPE

Complex numbers contain a real part and an imaginary part. The imaginary part is represented using 'j'.

Example:

```
z = 1.0 - 2.0j

print(z)

print(type(z))
```

VARIABLES

Variables act as containers for storing data values. A variable is created when a value is assigned to it.

Example:

```
x = 10

x = 15

print(x)
```

The output is 15 because the variable stores the latest assigned value.

DYNAMIC TYPING

Python uses dynamic typing, which means you do not need to declare the data type explicitly. The type is decided at runtime.

Example:

```
x = 42

print(type(x))
```

```
x = 'Python'

print(type(x))
```

OPERATORS

Operators are symbols used to perform operations on values.

Arithmetic Operators Example:

```
a = 10

b = 5

print(a + b)

print(a - b)

print(a * b)

print(a / b)

print(a % b)

print(a ** b)

print(a // b)
```

COMPARISON OPERATORS

Comparison operators compare two values and return a Boolean result.

Example:

```
a = 10

b = 20

print(a == b)

print(a != b)

print(a > b)

print(a < b)
```

LOGICAL OPERATORS

Logical operators are used to combine conditions.

Example:

```
x = True

y = False

print(x and y)

print(x or y)

print(not y)
```

ASSIGNMENT OPERATORS

Assignment operators assign or update values stored in variables.

Example:

```
x = 5

x += 3

print(x)
```

INPUT FUNCTION

The input() function is used to accept data from the user at runtime.

Example:

```
name = input('Enter your name: ')

print('Hello', name)
```

STRING FORMATTING

String formatting allows combining text and variables.

F-string Example:

```
name = 'Nancy'

age = 22

print(f'My name is {name} and I am {age} years old')
```

format() Example:

```
txt = 'For only {price:.2f} dollars'

print(txt.format(price=49))
```

BRANCHING (IF-ELIF-ELSE)

Branching is used to make decisions based on conditions.

Example:

```
age = 25

if age <= 12:
   print('Child')
elif age <= 19:
   print('Teenager')
elif age <= 35:
   print('Young Adult')
else:
   print('Adult')
```