1. Python Data Types

Python provides various built-in data types, which are essential for handling different types of data.

- **Common Data Types:**
  - **Numeric Types: int, float, complex**

- **Integer (`int`)**: Represents whole numbers, positive or negative, without decimals.

```
age = 25
temperature = -5
print(type(age))  # Output: <class 'int
```

Output :
<class 'int'>

- **Floating-Point (float)**: Represents numbers with a decimal point.

```
pi = 3.14159
negative_float = -2.71828
print(type(pi))  # Output: <class 'float'>
```

Output :
<class 'float'>

- **Complex (complex):** Represents complex numbers with real and imaginary parts.
- **Output :**
- **Output :**

```python
# Creating complex numbers
z1 = 3 + 4j
z2 = complex(2, -5)

# Accessing real and imaginary parts
print("z1:", z1)  # Output: (3+4j)
print("Real part of z1:", z1.real)  # Output: 3.0
print("Imaginary part of z1:", z1.imag)  # Output: 4.0

print("z2:", z2)  # Output: (2-5j)
print("Real part of z2:", z2.real)  # Output: 2.0
print("Imaginary part of z2:", z2.imag)  # Output: -5.0
```

Output:

```
z1: (3+4j)
Real part of z1: 3.0
Imaginary part of z1: 4.0
z2: (2-5j)
Real part of z2: 2.0
Imaginary part of z2: -5.0
```

Text Type

- **String (str):** Represents sequences of characters. Strings can be defined using single, double, or triple quotes.

```
greeting = "Hello, World!"
print(type(greeting))  # Output: <class 'str'>
```

Output :
<class 'str'>

- **Boolean Type**
- **Boolean (bool):** Represents one of two values: True or False.

```
is_valid = True
is_expired = False
print(type(is_valid))  # Output: <class 'bool'>
```

Output :
<class 'bool'>

- **Sequence Types**

1. **List (list):** An ordered, mutable collection of items. Lists can contain elements of different data types.

```
fruits = ['apple', 'banana', 'cherry']
print(type(fruits))  # Output: <class 'list'>
```

Output :
<class 'list'>

2. **Tuple (tuple):** An ordered, unchallengeable collection of items.

```
coordinates = (10.0, 20.0)
print(type(coordinates))  # Output: <class 'tuple'>
```

3.

Output :
<class 'tuple'>

4. **Range (range):** Represents an immutable sequence of numbers, commonly used for looping a specific number of times in for loops.

```
numbers = range(5)
print(type(numbers))  # Output: <class 'range'>
```

output :

<class 'range'>

5. **Mapping Type**

- **Dictionary (dict):** An unordered, mutable collection of key-value pairs. Keys must be unique and immutable.

```
student = {'name': 'John', 'age': 20}
print(type(student))  # Output: <class 'dict'>
```

output :

<class 'dict'>

Set Types

1. **Set (set):** An unordered collection of unique, immutable items. Sets are mutable.

```
unique_numbers = {1, 2, 3, 4, 5}
print(type(unique_numbers))  # Output: <class 'set'>
```

output :

<class 'set'>

Frozen Set (frozenset): An immutable version of a set.

```
frozen_numbers = frozenset([1, 2, 3, 4, 5])
print(type(frozen_numbers))  # Output: <class 'frozenset'>
```

output :

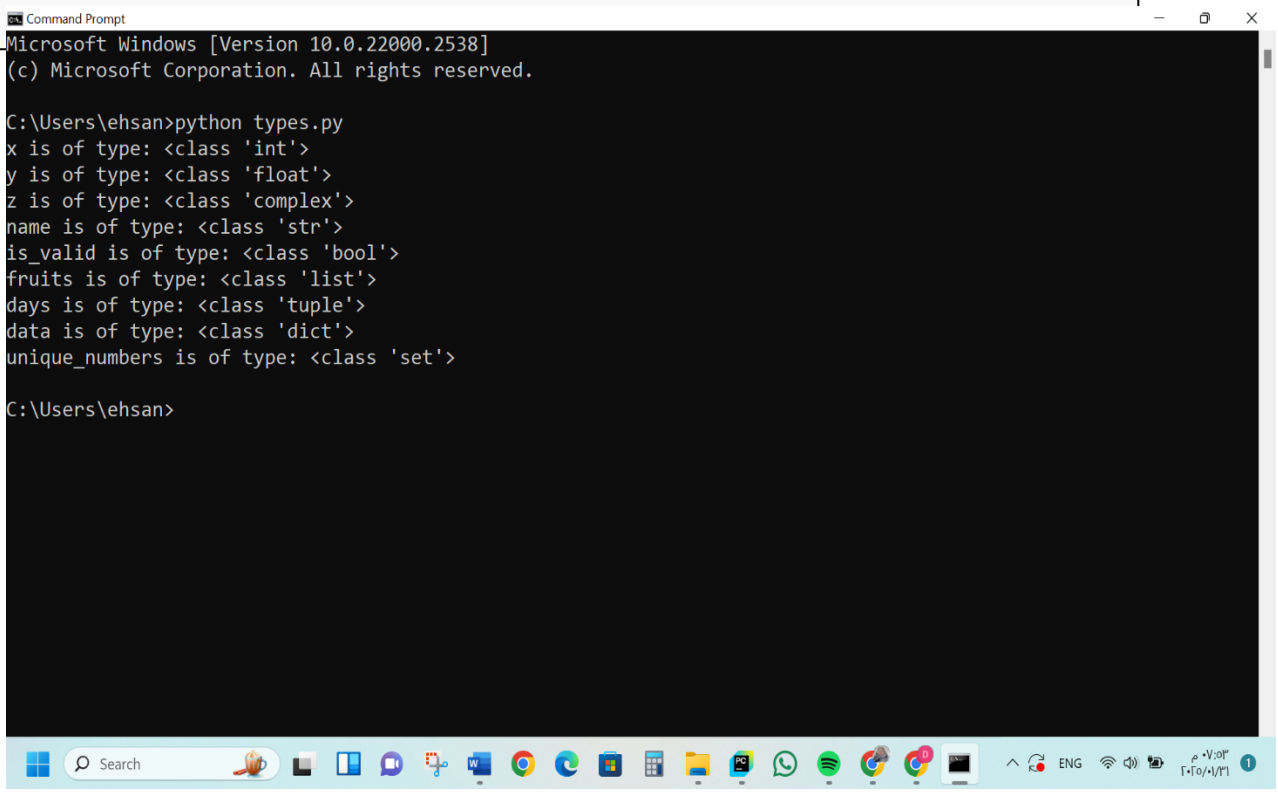
<class 'frozenset'>

Examples(1):

```
x = 10        # Integer
y = 3.14      # Float
z = 2 + 3j    # Complex number
name = "Python"  # String
is_valid = True  # Boolean
fruits = ["apple", "banana"]  # List
days = ("Monday", "Tuesday")  # Tuple
```

Q2: How can we declare and use different data types in Python?

```
x = 10        # Integer
print(f"x is of type: {type(x)}")


y = 3.14      # Float
print(f"y is of type: {type(y)}")

z = 2 + 3j    # Complex number
print(f"z is of type: {type(z)}")


name = "Python"  # String
print(f"name is of type: {type(name)}")


is_valid = True  # Boolean
print(f"is_valid is of type: {type(is_valid)}")

fruits = ["apple", "banana", "cherry"]  # List
print(f"fruits is of type: {type(fruits)}")

days = ("Monday", "Tuesday", "Wednesday")  # Tuple
print(f"days is of type: {type(days)}")

data = {"name": "Alice", "age": 25, "city": "New York"}  # Dictionary
print(f"data is of type: {type(data)}")


unique_numbers = {1, 2, 3, 4, 5}  # Set
print(f"unique_numbers is of type: {type(unique_numbers)}")
```



2. Simple Input & Output

The input() function is used to take user input, and print() is used to display output.

Example(2):

```
name = input("Enter your name: ")
print("Hello, " + name + "! Welcome to Python.")
```

3. Operators in Python

Operators are used to perform operations on variables and values.

Types of Operators:

Arithmetic Operators: +, -, *, /, //, %, **

Comparison Operators: ==, !=, >, <, >=, <=

Logical Operators: and, or, not

Assignment Operators: =, +=, -=, *=, /=, //=, %=

Example(3):

```
a = 10
b = 3
print("Addition:", a + b)
print("Multiplication:", a * b)
print("Exponentiation:", a ** b)
```

output :

Addition: 13
Multiplication: 30
Exponentiation: 1000

Python supports the following arithmetic operators:

+ (Addition)
- (Subtraction)
* (Multiplication)
/ (Division)
// (Floor Division)
% (Modulus)
** (Exponentiation)

4. Numbers and Math Functions

Python has built-in functions for mathematical operations.

Common Math Functions:

import math: Provides additional math functions (sqrt, ceil, floor, etc.)

abs(x): Returns the absolute value of x
pow(x, y): Computes x raised to the power y
round(x, n): Rounds x to n decimal places
max(x, y, z): Returns the maximum value
min(x, y, z): Returns the minimum value
math.sqrt(x): Returns the square root of x
math.ceil(x): Returns the smallest integer greater than or equal to x
Example ():

```
import math
num = float(input("Enter a number: "))
print("Square Root:", math.sqrt(num))
print("Ceiling of number:", math.ceil(num))
```

Output :

```
Enter a number: 4.1
Square Root: 2.0248456731316584
Ceiling of number: 5
```

Example(4): Ask the user for two numbers and print their sum, difference, product, and quotient.

Solution:

```
# Prompt the user for two numbers
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
# Perform arithmetic operations
sum_result = num1 + num2
difference = num1 - num2
product = num1 * num2
quotient = num1 / num2 if num2 != 0 else "Undefined (division by zero)"
# Display the results
print(f"Sum: {sum_result}")
print(f"Difference: {difference}")
print(f"Product: {product}")
print(f"Quotient: {quotient}")
```

Output :

```
Enter the first number: 4
Enter the second number: 3
Sum: 7.0
Difference: 1.0
Product: 12.0
Quotient: 1.3333333333333333
```

Example(5): simple Python program to calculate the area of a circle:

```
import math

# Function to calculate area of a circle
def circle_area(radius):  1 usage
    return math.pi * radius ** 2

# Example usage
radius = float(input("Enter the radius: "))
area = circle_area(radius)
print(f"The area of the circle with radius {radius} is {area:.2f}")
```

Output :

```
Enter the radius: 4
The area of the circle with radius 4.0 is 50.27
```

- **Takes the radius as input from the user.**
- **Uses the formula $A = \pi r^2$ to compute the area.**
- **Prints the result rounded to two decimal places.**

```
sentence = input("Enter a sentence: ")
word_count = len(sentence.split())

print(f"The number of words in the sentence is: {word_count}")
```

Output :

```
Enter a sentence: hi yasmin good day
The number of words in the sentence is: 4
```

Example (6):Find the Largest Among Three Numbers

```
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
num3 = float(input("Enter third number: "))

largest = max(num1, num2, num3)

print(f"The largest number is: {largest}")
```

Output :

```
Enter first number: 8
Enter second number: 4
Enter third number: 5
The largest number is: 8.0
```

Example (7):Check If a Number Is Even or Odd

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

if num % 2 == 0:
    print(f"{num} is even.")
else:
    print(f"{num} is odd.")
```

Output :

```
Enter a number: 5
5 is odd.
```

Example (8) :Write a script to swap two variables without using a temporary variable.

```
a = int(input("Enter first number: "))
b = int(input("Enter second number: "))

a, b = b, a  # Swapping without a temporary variable

print(f"After swapping: First number = {a}, Second number = {b}")
```

Output :

```
Enter first number: 8
Enter second number: 4
After swapping: First number = 4, Second number = 8
```

Example (9) :Prompt the user to enter their birth year and calculate their age.

```
from datetime import datetime

birth_year = int(input("Enter your birth year: "))
current_year = datetime.now().year
age = current_year - birth_year

print(f"You are {age} years old.")
```

Output :

```
Enter your birth year: 1999
You are 26 years old.
```

Exercises

Implement a simple calculator that performs basic arithmetic operations based on user choice.