**Python Assignment**  
  
 Key Features of Python

Python is a popular programming language for various reasons:

1. Readability and Simplicity: Python's syntax is clear and readable, which makes it easy for beginners to learn and understand.

2. Interpreted Language: Python code is executed line-by-line, which helps in debugging and prototyping.

3. Dynamically Typed: Variables in Python do not need explicit declaration of their data types.

4. Extensive Standard Library: Python comes with a large standard library that supports many common programming tasks.

5. Versatility: Python is used in web development, data analysis, artificial intelligence, scientific computing, and more.

6. Community Support: Python has a large and active community, which contributes to its extensive documentation and third-party modules.

Role of Predefined Keywords in Python

Predefined keywords in Python are reserved words that have special meaning and are used to define the syntax and structure of the Python language. Examples include `if`, `else`, `for`, `while`, `def`, `class`, `import`, `try`, `except`, and `return`.

Examples:

- `if` and `else`:

```python

x = 10

if x > 5:

print("x is greater than 5")

else:

print("x is less than or equal to 5")

```

- `for` loop:

```python

for i in range(5):

print(i)

```

- `def` (for defining functions):

```python

def greet(name):

return f"Hello, {name}!"

```

Mutable vs Immutable Objects in Python

Mutable Objects:

- Can be changed after their creation.

- Examples: lists, dictionaries, sets.

Example:

```python

my\_list = [1, 2, 3]

my\_list.append(4) my\_list is now [1, 2, 3, 4]

```

Immutable Objects:

- Cannot be changed after their creation.

- Examples: strings, tuples, integers, floats.

Example:

```python

my\_string = "Hello"

new\_string = my\_string + " World" my\_string remains "Hello"; new\_string is "Hello World"

```

Different Types of Operators in Python

Arithmetic Operators:

- Used for basic arithmetic operations.

- Examples: `+`, `-`, ``, `/`, `%`, ``, `//`

Example:

```python

a = 10

b = 3

print(a + b) 13

print(a - b) 7

print(a b) 30

print(a / b) 3.3333...

print(a % b) 1

print(a b) 1000

print(a // b) 3

```

Comparison Operators:

- Used to compare values.

- Examples: `==`, `!=`, `>`, `<`, `>=`, `<=`

Example:

```python

a = 10

b = 5

print(a == b) False

print(a != b) True

print(a > b) True

print(a < b) False

print(a >= b) True

print(a <= b) False

```

Logical Operators:

- Used to combine conditional statements.

- Examples: `and`, `or`, `not`

Example:

```python

a = True

b = False

print(a and b) False

print(a or b) True

print(not a) False

```

Assignment Operators:

- Used to assign values to variables.

- Examples: `=`, `+=`, `-=`, `=`, `/=`, `%=`, `=`, `//=`

Example:

```python

a = 5

a += 2 a is now 7

a = 3 a is now 21

```

Type Casting in Python

Type casting is the process of converting one data type into another. Python supports both implicit and explicit type casting.

Implicit Type Casting:

Python automatically converts a value from one type to another.

Example:

```python

a = 5

b = 2.5

c = a + b c is automatically converted to float

print(c) 7.5

```

Explicit Type Casting:

You manually convert a value from one type to another.

Example:

```python

a = 5

b = 2.5

c = int(b) c is now 2

d = str(a) d is now '5'

print(c, type(c)) 2 <class 'int'>

print(d, type(d)) 5 <class 'str'>

```

Conditional Statements in Python

Conditional statements are used to execute code based on certain conditions.

Example:

```python

x = 10

if x > 0:

print("x is positive")

elif x == 0:

print("x is zero")

else:

print("x is negative")

```

Different Types of Loops in Python

`for` Loop:

Used for iterating over a sequence (such as a list, tuple, or string).

Example:

```python

for i in range(5):

print(i)

```

`while` Loop:

Repeats as long as a condition is true.

Example:

```python

i = 0

while i < 5:

print(i)

i += 1

```

Use Cases:

- `for` loop: Useful for iterating over elements in a collection.

- `while` loop: Useful when the number of iterations is not known in advance and depends on a condition.