Python Fundamentals

Python Fundamentals: Identifiers, Keywords, and Data Types

1. Python Identifiers (Rules and Regulations)

An identifier is a name given to variables, functions, classes, modules, or other objects in Python. Python has specific rules for naming identifiers.

Rules for Python Identifiers

1. Must start with a letter (a-z, A-Z) or an underscore (_).
   - Valid: name, _age, Salary
   - Invalid: 1var, @name, #value

2. Can contain letters, digits (0-9), and underscores (_).
   - Valid: user_name, age1, total_amount
   - Invalid: user-name, first name, email@address

3. Cannot be a Python keyword or reserved word.
   - Invalid: if, for, while (these are keywords)

4. Case-sensitive (uppercase and lowercase letters are different).
   - Name and name are two different identifiers.

5. No special characters (!, @, #, %, etc.) are allowed.
   - Invalid: user@name, price$, value#

Python Fundamentals

Examples of Valid and Invalid Identifiers

Valid Identifiers	Invalid Identifiers
age	1age (starts with digit)
_salary	my-name (hyphen not allowed)
userName	class (Python keyword)
total_amount	for (reserved word)

Best Practices for Naming Identifiers

- Use meaningful names (student_name instead of sn).
- For constants, use uppercase (PI = 3.14).
- For multi-word names, use:
  - snake_case (user_name) -> Preferred in Python.
  - camelCase (userName) -> Used in some other languages.

2. Reserved Words and Keywords in Python

Python has a set of reserved words (keywords) that have special meanings and cannot be used as identifiers.

List of Python Keywords (as of Python 3.11)

Keyword	Purpose
and	Logical AND
as	Alias in imports
assert	Debugging condition

Python Fundamentals

break	Exit a loop
class	Define a class
continue	Skip current loop iteration
def	Define a function
del	Delete an object
elif	Else-if condition
else	Else block
except	Exception handling
False	Boolean false
finally	Execute code after try-except
for	For loop
from	Import specific module parts
global	Declare global variable
if	If condition
import	Import modules
in	Membership test
is	Identity comparison
lambda	Anonymous function
None	Null value
nonlocal	Modify outer function variable
not	Logical NOT
or	Logical OR
pass	Placeholder (do nothing)
raise	Raise an exception
return	Return from a function

Python Fundamentals

True Boolean true

try Exception handling block

while While loop

with Context manager

yield Return generator value

Example of Using Keywords

```
if age >= 18:  # 'if' is a keyword
    print("Adult")
else:          # 'else' is a keyword
    print("Minor")
```

Checking Python Keywords

```
import keyword
print(keyword.kwlist)
```

3. Basic Data Types in Python

Python data types are categorized based on:

- Sequential vs Non-Sequential
- Ordered vs Unordered
- Mutable vs Immutable

Classification of Data Types

Data Type	Category	Ordered	Mutable	Example

Python Fundamentals

int	Numeric	-	Immutable	x = 10
float	Numeric	-	Immutable	y = 3.14
str	Sequential	Ordered	Immutable	s = "hello"
list	Sequential	Ordered	Mutable	lst = [1, 2, 3]
tuple	Sequential	Ordered	Immutable	tup = (1, 2, 3)
set	Non-Seq	Unordered	Mutable	s = {1, 2, 3}
dict	Non-Seq	Ordered	Mutable	d = {"name": "Alice"}

Examples of Data Types

(1) Sequential Data Types (Ordered)

- String (str) -> Immutable

```
name = "Python"
print(name[0])  # 'P'
```

- List (list) -> Mutable

```
numbers = [1, 2, 3]
numbers.append(4)  # [1, 2, 3, 4]
```

- Tuple (tuple) -> Immutable

```
point = (10, 20)
print(point[1])  # 20
```

(2) Non-Sequential Data Types (Unordered)

- Set (set) -> Mutable, Unique elements

```
fruits = {"apple", "banana", "cherry"}
```

```
fruits.add("orange")
```

- Dictionary (dict) -> Key-Value pairs

```
student = {"name": "Alice", "age": 20}
print(student["name"])  # "Alice"
```

Mutable vs Immutable

- Mutable: list, set, dict
- Immutable: int, float, str, tuple

```
# Mutable Example (List)
lst = [1, 2, 3]
lst[0] = 99  # Allowed -> [99, 2, 3]

# Immutable Example (String)
s = "hello"
s[0] = 'H'  # Error! Strings are immutable
```

Conclusion

- Identifiers must follow naming rules (no keywords, no special chars).
- Keywords (if, for, while) have special meanings and cannot be used as identifiers.
- Data Types can be:
  - Sequential (str, list, tuple) or Non-Sequential (set, dict).
  - Ordered (list, tuple, str) or Unordered (set).
  - Mutable (list, dict, set) or Immutable (int, str, tuple).

Python Fundamentals