🧮 Interview Questions : Variables in Python

🎯 **Objective**:

Enhance your proficiency with Variables by completing a series of tasks that cover creation, manipulation, and application of Variables in various scenarios.

🟢 Beginner Level

1. **What is a variable in Python?**
    - *Answer*: A variable in Python is a symbolic name that is a reference or pointer to an object. Once an object is assigned to a variable, you can refer to the object by that name.
2. **How do you assign a value to a variable in Python?**
    - *Answer*: Using the assignment operator =. For example: `x = 10`.
3. **What are the rules for naming variables in Python?**
    - *Answer*: Variable names must start with a letter (a-z, A-Z) or an underscore (_) followed by letters, numbers (0-9), or underscores. They are case-sensitive and cannot be Python reserved keywords.
4. **What is dynamic typing in Python?**
    - *Answer*: Python is dynamically typed, meaning you don't need to declare the type of a variable; the interpreter infers the type at runtime.
5. **How can you check the type of a variable?**
    - *Answer*: Using the built-in `type()` function. For example: `type(x)`.
6. **What is the difference between = and == in Python?**
    - *Answer*: = is the assignment operator, used to assign values to variables. == is the equality operator, used to compare two values.
7. **Can variable names start with a number?**
    - *Answer*: No, variable names cannot start with a number. They must begin with a letter or an underscore.
8. **What is the difference between a variable and a constant?**
    - *Answer*: A variable's value can change during program execution, while a constant's value remains unchanged. Python doesn't have built-in constant types, but by convention, constants are written in uppercase letters.

🧮 Interview Questions : Variables in Python

9. **Is Python case-sensitive when it comes to variable names?**
    - *Answer*: Yes, Python is case-sensitive. For example, `Variable` and `variable` would be considered two different identifiers.

10. **What is the purpose of the `del` statement?**
    - *Answer*: The `del` statement is used to delete a variable, freeing up the memory space. For example: `del x`.

🟡 Intermediate Level

11. **What is variable scope in Python?**
    - *Answer*: Scope refers to the region of the program where a variable is recognized. Python has local, enclosing, global, and built-in scopes.

12. **What is the difference between global and local variables?**
    - *Answer*: A global variable is defined outside any function and is accessible throughout the program. A local variable is defined within a function and is accessible only within that function.

13. **How do you declare a global variable inside a function?**
    - *Answer*: By using the `global` keyword. For example:

```
def my_function():
    global x
    x = 10
```

14. What is the `nonlocal` keyword used for?

- *Answer*: The `nonlocal` keyword is used in nested functions to refer to variables in the nearest enclosing scope that is not global.

15. Can you change the type of a variable after it has been set?

- *Answer*: Yes, since Python is dynamically typed, you can reassign a variable to a different type. For example.

```
x = 10
x = "Hello"
```

16. What is variable shadowing in Python?

- *Answer*: Variable shadowing occurs when a variable in a local scope has the same name as a variable in an outer scope, thereby overriding access to the outer variable within the local scope.

17. How does Python handle variable memory management?

- *Answer*: Python uses reference counting and a garbage collector to manage memory. When a variable's reference count drops to zero, the memory is reclaimed.

18. What are mutable and immutable types in Python?

- *Answer*: Mutable types (like lists, dictionaries) can be changed after creation, while immutable types (like integers, strings, tuples) cannot.

19. How can you check if two variables reference the same object?

- *Answer*: Using the `is` operator. For example: `x is y` returns `True` if both variables point to the same object.

20. What is the difference between `is` and `==` operators?

- *Answer*: == checks for value equality, whereas `is` checks for identity (whether two references point to the same object).

🔴 Advanced Level

21. **Explain the LEGB rule in Python.**
    - *Answer*: LEGB stands for Local, Enclosing, Global, Built-in—the order in which Python searches for variable names.
22. **What happens if you modify a global variable inside a function without declaring it as global?**
    - *Answer*: Python treats it as a new local variable, and the global variable remains unchanged.
23. **How can you prevent a variable from being modified?**
    - *Answer*: By using immutable data types or by convention (e.g., naming constants in uppercase).

24. **What are closures in Python?**
    - *Answer*: Closures are functions that capture the bindings of free variables in their enclosing scopes. This means the function remembers the values from its enclosing scope even if that scope is no longer in memory.

25. **How do default mutable arguments lead to unexpected behavior in functions?**
    - *Answer*: Default mutable arguments retain changes between function calls, leading to potential bugs. It's recommended to use `None` as a default and assign the mutable object inside the function.

26. **What is variable interning in Python?**
    - *Answer*: Interning is an optimization technique where identical immutable objects are stored only once to save memory. Python automatically interns small integers and strings.

27. **How does Python's garbage collector handle circular references?**
    - *Answer*: Python's garbage collector can detect and collect cyclic references using its cyclic garbage collector, which is part of the `gc` module.

28. **What is the purpose of the `globals()` and `locals()` functions?**
    - *Answer*: `globals()` returns a dictionary of the current global symbol table, while `locals()` returns a dictionary of the current local symbol table.

29. **Can you have variables with the same name in different scopes?**
    - *Answer*: Yes, variables can have the same name in different scopes. The variable in the innermost scope will shadow the others.

30. **How do you manage memory efficiently when dealing with large datasets in Python?**
    - *Answer*: By using generators, efficient data structures, and tools like `gc` module to monitor and manage memory usage.