**VARIABLES**

**What is a variable in Python?**

**a) A container to store data** b) A function to perform mathematical operations

c) A loop to iterate through data d) A conditional statement for decision-making

**Explanation:** A container to store data – This is the correct definition of a variable. In Python, variables are used to store data which can be accessed and manipulated during program execution.

Q2. Which symbol is used to assign a value to a variable in Python?

**a) =** b) : c) # d) $

**Explanation:** The equals sign (=) is used for assignment in Python. It assigns the value on the right to the variable on the left.

**Q3. What is the correct way to create an integer variable named “x” with the value 5 in Python?**

**a) x = 5** b) int x = 5c) x := 5d) 5 = x

**Explanation:** x = 5 – This is the correct syntax to create an integer variable named “x” with the value 5 in Python. The variable name comes first, followed by the assignment operator (=), and then the value.

**Q4. In Python, variable names are case-sensitive. What does this mean?**

a) Variable names cannot contain uppercase letters.

b) Variable names must always start with an uppercase letter.

c) Variable names must always be written in uppercase letters.

**d) Variable names are distinguished by uppercase and lowercase letters.**

**Explanation:** Variable names are distinguished by uppercase and lowercase letters in Python. Therefore, variables with different cases (e.g., “x” and “X”) are treated as different variables.

Q5. Which of the following is a valid variable name in Python?

a) 3total

b) my\_variable

c) global

d) class

**Answer:**b

**Explanation:** my\_variable – This is a valid variable name in Python. Variable names can contain letters, numbers, and underscores, but cannot start with a number.

Q6. What is the type of the variable “x” after executing the following code: x = 5.0?

a) int

b) str

c) float

d) bool

**Answer:**c

**Explanation:** float – After executing the code “x = 5.0”, the variable “x” will be of type float because the value assigned to it is a floating-point number.

Q7. What happens if you try to use a variable that has not been defined in Python?

a) Python will automatically define the variable. b) Python will prompt the user to define the variable.

**c) Python will raise a NameError.** d) Python will ignore the variable.

**Explanation:** Python will raise a NameError indicating that the variable is not defined if you try to use a variable that has not been defined.

Q8. Which of the following is not a valid variable name in Python?

a) my\_variable\_1

b) 1st\_variable

c) variable2

d) \_variable

**Answer:**b

**Explanation:** 1st\_variable – Variable names cannot start with a number in Python. They must start with a letter or an underscore.

Q9. What is the output of the following code?

x = 10

y = x + 5

print(y)

a) 10

b) 5

c) 15

d) x + 5

**Answer:**c

**Explanation:** The code assigns the value 10 to the variable “x”, then adds 5 to it and assigns the result to the variable “y”, and finally prints the value of “y”, which is 15.

Q10. Which data type is mutable in Python?

a) int

b) float

c) list

d) tuple

**Answer:**c

**Explanation:** Lists are mutable data types in Python, meaning that their elements can be changed after the list is created.

Q11. What is the scope of a variable in Python?

a) The range of values a variable can hold.

b) The visibility of a variable within a program.

c) The data type of a variable.

d) The memory location of a variable.

**Answer: b**

**Explanation:** Scope refers to the visibility of a variable within a program. Local variables are only accessible within the function they are defined in, while global variables are accessible from any part of the program.

Q12. What does the “global” keyword do in Python?

a) Declares a variable that is accessible from anywhere in the program.

b) Declares a variable that is local to a specific function.

c) Declares a variable that is accessible only within a specific module.

d) Declares a variable that cannot be modified.

**Answer:**a

**Explanation:** The “global” keyword is used to declare a variable that can be accessed from anywhere in the program, not just within a specific function.

Q13. What is the purpose of the “del” keyword in Python?

a) To delete a variable from memory.

b) To define a new variable.

c) To initialize a variable.

d) To declare a variable as global.

**Answer:**a

**Explanation:** The “del” keyword in Python is used to delete a variable, freeing up the memory it occupied.

Q14. Which of the following statements about Python variables is correct?

a) Variables must be declared before use.

b) Variables can only store numeric values.

c) Variables can be reassigned to different data types.

d) Variables cannot be assigned values within loops.

**Answer:**c

**Explanation:** Unlike some other programming languages, Python allows variables to be reassigned to different data types.

Q15. What is the output of the following code?

x = 10  
x += 5  
print(x)

a) 5

b) 10

c) 15

d) 20

**Answer:**c

**Explanation:** The code increments the value of “x” by 5 using the “+=” operator and then prints the updated value, which is 15.

Q16. What is the difference between “==” and “is” in Python when comparing variables?

a) “==” checks for equality of values, while “is” checks for equality of memory location.

b) “==” checks for equality of memory location, while “is” checks for equality of values.

c) There is no difference; they are interchangeable.

d) “==” is used for assignment, while “is” is used for comparison.

**Answer:**a

**Explanation:** The “==” operator checks whether the values of two variables are equal, while the “is” operator checks whether the two variables refer to the same object in memory.

Q17. What is the output of the following code?

x = 5y = xx = 10print(y)

a) 5

b) 10

c) Error

d) 0

**Answer:**a

**Explanation:** The value of “y” is assigned to be the same as the initial value of “x” (which is 5) before “x” is reassigned to 10. Therefore, printing “y” will output 5.

Q18. Which of the following is not a valid way to comment out a line in Python?

a) // This is a comment

b) # This is a comment

c) ”’ This is a comment ”’

d) “”” This is a comment “””

**Answer:**a

**Explanation:** In Python, the “//” symbol is used for integer division, not for commenting. Comments in Python start with the “#” symbol.

Q19. What is the output of the following code?

x = "Hello"y = xx += " World"print(y)

a) Hello

b) World

c) Hello World

d) Error

**Answer:**a

**Explanation:** The variable “y” is assigned the same value as “x” before “x” is modified. Therefore, printing “y” will output the original value of “x”, which is “Hello”.

Q20. What is the purpose of the “id()” function in Python?

a) To identify the data type of a variable.

b) To generate random numbers.

c) To return the memory address of an object.

d) To convert a variable to an integer.

**Answer:**c

**Explanation:** The “id()” function in Python returns the memory address of an object, which is a unique identifier for that object.

Q21. What is the difference between local and global variables in Python?

a) Local variables are declared outside of any function, while global variables are declared within functions.

b) Local variables are accessible only within the function they are defined in, while global variables are accessible from any part of the program.

c) Local variables are stored in the global namespace, while global variables are stored in local namespaces.

d) Local variables are immutable, while global variables are mutable.

**Answer:**b

**Explanation:** Local variables are only accessible within the function they are defined in, while global variables are accessible from any part of the program.

Q22. What does the “nonlocal” keyword do in Python?

a) Declares a variable that is local to a specific function.

b) Declares a variable that is accessible from anywhere in the program.

c) Declares a variable that is accessible only within a specific module.

d) Declares a variable that is not local to the current function but not global either.

**Answer:**d

**Explanation:** The “nonlocal” keyword is used to declare a variable in an enclosing scope (outside of the current function), but not in the global scope.

Q23. What is the output of the following code?

def test():

global x

x = 10

test()

print(x)

a) 0

b) 10

c) Error

d) None

**Answer:**b

**Explanation:** Even though “x” is declared as global within the function “test()”, it is still accessible globally. Therefore, printing “x” after calling the function will output the value assigned to it within the function, which is 10.

Q24. Which of the following is true about immutable data types in Python?

a) Their values cannot be changed after assignment.

b) They can only store numeric values.

c) They are always passed by reference.

d) They include lists and dictionaries.

**Answer:**a

**Explanation:** Immutable data types in Python, such as tuples and strings, cannot be modified after they are created.

Q25. What is the output of the following code?

x = [1, 2, 3]y = xx.append(4)print(y)

a) [1, 2, 3]

b) [1, 2, 3, 4]

c) [1, 2, 4]

d) Error

**Answer:**b

**Explanation:** Both “x” and “y” refer to the same list object, so appending a value to “x” will also affect “y”.

Q26. What is the purpose of the “locals()” function in Python?

a) To return a dictionary of local variables in the current scope.

b) To return the memory address of a local variable.

c) To check if a variable is local or global.

d) To assign values to local variables.

**Answer:**a

**Explanation:** The “locals()” function in Python returns a dictionary containing all local variables in the current scope.

Q27. What does the “id()” function return in Python?

a) The identity of an object.

b) The memory address of an object.

c) The data type of an object.

d) The size of an object.

**Answer:**b

**Explanation:** The “id()” function in Python returns the memory address of an object, which uniquely identifies it.

**Q28. What is the output of the following code?**

x = 10

def test():

print(x)

x = 20

test()

a) 10

b) 20

c) Error

d) None

**Answer:**c

**Explanation:** This code will raise an UnboundLocalError because “x” is being referenced before it is assigned within the function “test()”.

Q29. Which of the following is not a valid way to delete a variable in Python?

a) del x

b) x.delete()

c) x = None

d) x = 0

**Answer:**b

**Explanation:** There is no method called “delete()” to delete a variable in Python. The correct ways to delete a variable are using “del” or assigning it to None.

Q30. What is the output of the following code?

x = 10

def test():

global x

x += 5

test()

print(x)

a) 5

b) 10

c) 15

d) Error

**Answer:**c

**Explanation:** The “global” keyword inside the function “test()” allows the function to modify the global variable “x”, so after calling the function, the value of “x” becomes 15.

Q31. What is the output of the following code snippet?

**def** **test**():

x = 10

**def** **inner\_test**():

**nonlocal** x

x += 5

inner\_test()

print(x)

test()

a) 1

b) 15

c) Error

d) 5

**Answer:**b

**Explanation:** The nonlocal keyword allows modification of a variable in an outer (but non-global) scope. The inner\_test() function modifies the value of x by adding 5 to it. Therefore, the output will be 15.

Q32. What is the output of the following code snippet?

x = 10

**def** **test**():

x = 20

**def** **inner\_test**():

**nonlocal** x

x += 5

inner\_test()

print(x)

test()

print(x)

a) 25, 10

b) 25, 25

c) 20, 10

d) 20, 20

**Answer:**c

**Explanation:** Inside the inner\_test() function, the nonlocal keyword allows modification of the x variable from the outer scope, adding 5 to it. However, this change is local to the test() function. Therefore, the output within the test() function is 25, but outside of it, x remains unaffected and prints 10.

Q33. What is the output of the following code snippet?

x = 10

**def** **test**():

**global** x

x = 20

**def** **inner\_test**():

**nonlocal** x

x += 5

inner\_test()

print(x)

test()

print(x)

a) 15, 20

b) 25, 20

c) Error

d) 15, 25

**Answer:**c

**Explanation:** This code will raise a SyntaxError because a variable cannot be declared both global and nonlocal in the same scope.

Q34. What is the output of the following code snippet?

x = 10

def **test**():

x = 20

def **inner\_test**():

**global** x

x += 5

inner\_test()

print(x)

test()

print(x)

a) 25, 25

b) 20, 10

c) Error

d) 20, 25

**Answer:**a

**Explanation:** Inside the inner\_test() function, the global keyword is used to access the global variable x and modify its value by adding 5 to it. Therefore, the output within the test() function is 25, and outside of it, x remains 25 as well.

Q35. What is the output of the following code snippet?

x = 10

**def** **test**():

x = 20

**def** **inner\_test**():

**nonlocal** x

x += 5

inner\_test()

print(x)

test()

print(x)

a) 20, 10 b) 25, 10 c) 20, 20 d) 25, 25 **Answer:**c

**Explanation:** Inside the inner\_test() function, the nonlocal keyword allows modification of the x variable from the outer test() function. Therefore, the output within the test() function is 25, but outside of it, x remains 20.