**Level 1: The Digital Cell**

**Intro (History + Need)**

**MCQs:**

1. Who created Python?  
   a) Dennis Ritchie  
   b) Guido van Rossum ✅  
   c) James Gosling  
   d) Bjarne Stroustrup
2. Python was released in the year:  
   a) 1989  
   b) 1991 ✅  
   c) 1995  
   d) 2000
3. Python is an example of a \_\_\_\_\_\_\_\_\_\_ language.  
   a) Compiled  
   b) Interpreted ✅  
   c) Assembly  
   d) Machine

**Fill in the Blanks:**  
4. The name Python was inspired by the show \_\_\_\_\_\_\_\_\_\_\_\_\_\_. *(Monty Python’s Flying Circus)*  
5. Python is known for its \_\_\_\_\_\_\_\_\_\_\_\_\_ syntax, making it easy to read. *(simple/clean)*

**True/False:**  
6. Python supports both functional and object-oriented programming. *(True ✅)*  
7. Python is not an open-source language. *(False ❌)*

**Basic Syntax**

**MCQs:**

1. What is used to indicate a block of code in Python?  
   a) Curly braces {}  
   b) Indentation ✅  
   c) Semicolons ;  
   d) Parentheses ()
2. Which of the following is a valid variable name in Python?  
   a) 2variable  
   b) var\_1 ✅  
   c) var-2  
   d) @variable

**Fill in the Blanks:**  
3. In Python, indentation is used to define \_\_\_\_\_\_\_\_\_\_. *(blocks of code)*  
4. The # symbol is used to write \_\_\_\_\_\_\_\_\_\_ in Python. *(comments)*

**True/False:**  
5. Python is case-sensitive. *(True ✅)*  
6. The semicolon ; is required at the end of every Python statement. *(False ❌)*

**Comments**

**MCQs:**

1. How do you write a multi-line comment in Python?  
   a) // This is a comment  
   b) /\* This is a comment \*/  
   c) ''' This is a comment ''' ✅  
   d) # This is a comment
2. What is the purpose of comments in Python?  
   a) Improve performance  
   b) Explain code ✅  
   c) Increase execution speed  
   d) None of the above

**Fill in the Blanks:**  
3. Single-line comments in Python start with the \_\_\_\_\_\_\_\_\_\_ symbol. *(#)*  
4. A \_\_\_\_\_\_\_\_\_\_ is used to document a function or module. *(docstring)*

**True/False:**  
5. Comments affect the execution of the program. *(False ❌)*  
6. Python does not support multi-line comments. *(False ❌ – It supports docstrings)*

**Print Statement**

**MCQs:**

1. What will be the output of print("Hello", "World", sep="-")?  
   a) HelloWorld  
   b) Hello World  
   c) Hello-World ✅  
   d) Hello,World
2. What is the default value of the sep parameter in print()?  
   a) , ✅  
   b) -  
   c) \_  
   d) .

**Fill in the Blanks:**  
3. The print() function is used to \_\_\_\_\_\_\_\_\_\_ messages on the screen. *(display/output)*  
4. The parameter used to prevent a new line in print() is \_\_\_\_\_\_\_\_\_\_. *(end="")*

**True/False:**  
5. print(5 + "10") will work correctly. *(False ❌ – TypeError: cannot concatenate int and str)*  
6. print() can be used to print multiple values in one statement. *(True ✅)*

**String Manipulation (Concatenation)**

**MCQs:**

1. What is the output of "Python" + "3"?  
   a) Python 3  
   b) Python3 ✅  
   c) Python+3  
   d) Error
2. What method is used to join a list of strings into a single string?  
   a) split()  
   b) join() ✅  
   c) append()  
   d) concat()

**Fill in the Blanks:**  
3. The + operator is used for \_\_\_\_\_\_\_\_\_\_ strings. *(concatenating)*  
4. str(100) + "Python" will result in \_\_\_\_\_\_\_\_\_\_. *(100Python)*

**True/False:**  
5. "Hello" + 5 is a valid operation in Python. *(False ❌ – TypeError: cannot concatenate str and int)*  
6. " ".join(["Hello", "World"]) results in "Hello World". *(True ✅)*

**Level 2: The Firewall**

**Variables and Operations**

**MCQs:**

1. Which of the following is a valid variable name in Python?  
   a) 1var  
   b) var\_1 ✅  
   c) var-2  
   d) @variable
2. What will be the output of x = 10; y = 5; print(x + y \* 2)?  
   a) 30  
   b) 20 ✅  
   c) 15  
   d) 10
3. Which operator is used for exponentiation in Python?  
   a) ^  
   b) \*\* ✅  
   c) //  
   d) %

**Fill in the Blanks:**

1. A variable in Python is declared by simply assigning a \_\_\_\_\_\_\_\_\_\_ to it. *(value)*
2. The operator used for integer division in Python is \_\_\_\_\_\_\_\_\_\_. *(//)*
3. The = operator is used for \_\_\_\_\_\_\_\_\_\_ a value to a variable. *(assigning)*
4. The modulo operator % returns the \_\_\_\_\_\_\_\_\_\_ of a division. *(remainder)*

**True/False:**

1. Variables in Python need to be declared with a specific type. *(False ❌ – Python is dynamically typed)*
2. x = 10; x += 5 is the same as x = x + 5. *(True ✅)*
3. 10 // 3 will return 3.33. *(False ❌ – It returns 3 as an integer division)*

**Data Types**

**MCQs:**

1. Which of the following is **not** a valid data type in Python?  
   a) int  
   b) float  
   c) double ✅  
   d) complex
2. What is the type of type(True)?  
   a) bool  
   b) str  
   c) int  
   d) type ✅
3. What will be the output of type(5.0)?  
   a) int  
   b) float ✅  
   c) double  
   d) str

**Fill in the Blanks:**

1. The Boolean values in Python are \_\_\_\_\_\_\_\_\_\_ and \_\_\_\_\_\_\_\_\_\_. *(True, False)*
2. A complex number in Python is written as \_\_\_\_\_\_\_\_\_\_. *(a + bj)*
3. Strings in Python are enclosed in \_\_\_\_\_\_\_\_\_\_ or \_\_\_\_\_\_\_\_\_\_. *(single quotes, double quotes)*
4. A list is a \_\_\_\_\_\_\_\_\_\_ collection, while a tuple is an \_\_\_\_\_\_\_\_\_\_ collection. *(mutable, immutable)*

**True/False:**

1. bool(0) returns True. *(False ❌ – bool(0) is False)*
2. In Python, the str data type is immutable. *(True ✅)*
3. A dictionary is an unordered collection of key-value pairs. *(True ✅)*

**Input and Type Casting**

**MCQs:**

1. What function is used to take input from the user in Python?  
   a) read()  
   b) scan()  
   c) input() ✅  
   d) get()
2. What is the output of type(input("Enter a number: ")) if the user enters 10?  
   a) int  
   b) str ✅  
   c) float  
   d) bool
3. How do you convert a string "10" to an integer?  
   a) str(10)  
   b) int("10") ✅  
   c) float("10")  
   d) convert(10)

**Fill in the Blanks:**

1. The input() function always returns a value of type \_\_\_\_\_\_\_\_\_\_. *(str)*
2. To convert a float to an integer, we use the \_\_\_\_\_\_\_\_\_\_ function. *(int())*
3. To get a float input from the user, we use \_\_\_\_\_\_\_\_\_\_. *(float(input()))*
4. The str() function is used to convert \_\_\_\_\_\_\_\_\_\_ into a string. *(any data type)*

**True/False:**

1. int("10.5") will work correctly. *(False ❌ – It will raise an error)*
2. float(5) will return 5.0. *(True ✅)*
3. Type casting is the process of converting one data type into another. *(True ✅)*

**Operators**

**MCQs:**

1. What is the result of 5 > 3 and 2 < 4?  
   a) True ✅  
   b) False  
   c) None  
   d) Error
2. What is the result of not (5 > 3)?  
   a) True  
   b) False ✅  
   c) None  
   d) Error
3. Which of the following operators is used to check membership in a sequence?  
   a) ==  
   b) is  
   c) in ✅  
   d) and

**Fill in the Blanks:**

1. The == operator is used for \_\_\_\_\_\_\_\_\_\_ comparison. *(equality)*
2. The and operator returns \_\_\_\_\_\_\_\_\_\_ if both conditions are True. *(True)*
3. The identity operator used to check if two variables refer to the same object is \_\_\_\_\_\_\_\_\_\_. *(is)*
4. The not operator is used to \_\_\_\_\_\_\_\_\_\_ a boolean value. *(negate/invert)*

**True/False:**

1. The or operator returns True if at least one condition is True. *(True ✅)*
2. The is operator checks for value equality. *(False ❌ – It checks for object identity)*
3. 5 in [1, 2, 3, 4, 5] will return True. *(True ✅)*

**Bitwise Operators (Optional)**

**MCQs:**

1. What is the result of 5 & 3?  
   a) 1 ✅  
   b) 3  
   c) 5  
   d) 15
2. What is the result of 5 | 3?  
   a) 1  
   b) 3  
   c) 5  
   d) 7 ✅
3. What is the result of 5 << 1?  
   a) 2  
   b) 10 ✅  
   c) 20  
   d) 15

**Fill in the Blanks:**

1. The bitwise AND operator is represented by \_\_\_\_\_\_\_\_\_\_. *(&)*
2. The bitwise OR operator is represented by \_\_\_\_\_\_\_\_\_\_. *(|)*
3. The bitwise NOT operator is represented by \_\_\_\_\_\_\_\_\_\_. *(~)*
4. The left shift operator << shifts bits towards the \_\_\_\_\_\_\_\_\_\_. *(left)*

**True/False:**

1. The XOR operator ^ returns 1 if both bits are the same. *(False ❌ – It returns 0 if both bits are the same, 1 if different)*
2. The right shift operator >> moves bits to the left. *(False ❌ – It moves bits to the right)*
3. Bitwise operations work only on integers. *(True ✅)*