1. What is Python and what are its key features?

Python is a high-level, interpreted language known for simplicity, readability, OOP support, and a large library.

2. What is the difference between Python 2 and Python 3?

Python 2 is outdated. Python 3 is current with Unicode support, print as function, and better division handling.

3. Is Python interpreted or compiled?

Python is interpreted line by line but internally converted to bytecode executed by the PVM.

4. What are the different data types in Python?

int, float, str, bool, list, tuple, set, dict, and NoneType.

5. What is the difference between is and == ?

`is` checks memory identity, `==` checks value equality.

6. What are variables in Python?

Variables are names pointing to objects in memory and are dynamically typed.

7. What is the purpose of None ?

`None` means no value or null. Used as placeholders or default returns.

8. What is dynamic typing?

Variables can change type during execution based on assigned value.

9. What is the difference between mutable and immutable types?

Mutable objects (list, dict, set) can change. Immutable (int, str, tuple) cannot.

10. What are Python's immutable types?

int, float, str, tuple, frozenset, bool.

11. What is a keyword in Python?

Keywords are reserved words with special meaning (e.g., if, else, for).

12. What is PEP8?

PEP8 is Python's style guide for clean and readable code.

13. What is the use of id() function?

It returns the unique memory address of an object.

14. How does Python manage memory?

Python uses private heap and garbage collection with reference counting.

15. What are the rules for naming variables?

Start with letters/underscore, case-sensitive, no keywords, can contain digits.

16. What is the difference between a script and a module?

Script is a runnable file. Module is reusable code imported in programs.

17. What are comments and docstrings in Python?
Comments use # for notes. Docstrings use triple quotes for documentation.

18. What are literals in Python?
Literal values directly written like numbers, strings, True/False, None.

19. What are magic constants like __name__ used for?
`__name__` shows how a file is run. '__main__' means run directly.

20. What is the purpose of indentation in Python?
Indentation defines code blocks instead of braces.

21. What is a namespace?
Namespace is a mapping of names to objects, like local and global scopes.

22. What is the type() function?
It shows the type/class of an object.

23. What is a dynamic language?
Language where types are checked at runtime, like Python.

24. How does Python handle variable swapping?
x, y = y, x swaps values in one line.

25. What does pass do in Python?
It’s a placeholder statement that does nothing.

36. What are control flow statements in Python?
They change execution order, e.g., if, for, while, break, continue.

37. What is the difference between if, elif, and else?
if checks condition, elif checks next if previous fails, else is default.

38. What is the difference between while and for loops?
while runs until condition fails, for iterates over a sequence.

39. What is the use of break and continue?
break exits loop, continue skips current iteration.

40. What is the use of the else block with loops?
else runs if loop finishes without break.

41. How can we loop over a range of numbers?
Use range(start, stop, step) in for loops.

42. What is the use of range()?

It generates sequences of numbers for iteration.

43. What is the difference between range() and enumerate()?
range() gives numbers, enumerate() gives index with value.

44. What is zip() used for in loops?
It combines multiple iterables into tuples element-wise.

45. What is a nested loop?
Loop inside another loop.

46. What are infinite loops?
Loops that never end if condition is always True.

47. How to find if a number is even or odd in Python?
Use n % 2 == 0 for even, else odd.

48. How does Python evaluate truthy and falsy values?
0, '', None, [] are False. Non-empty or non-zero are True.

49. What is a conditional expression (ternary operator)?
Syntax: x if condition else y.

50. How do you use short-circuiting in Python?
In `and`/`or`, evaluation stops once result is known.

51. What are the types of operators in Python?
Arithmetic, Comparison, Logical, Bitwise, Assignment, Membership, Identity.

52. What is the difference between == and is?
`==` checks value, `is` checks memory identity.

53. How does Python handle operator precedence?
Operators follow rules, e.g., *, / before +, -.

54. What are logical operators in Python?
and, or, not.

55. What is identity vs equality in Python?
Identity compares objects, equality compares values.

56. What is membership testing (in, not in)?
Checks if element exists in a sequence.

57. What is the difference between += and =+ ?
a += 1 increments, a =+ 1 assigns +1.

58. How does the not operator work?

It inverts boolean value.

59. What is the difference between bitwise and logical operators?
Bitwise works on bits (&, |), logical on booleans (and, or).

60. Can you overload operators in Python?
Yes, using special methods like __add__, __eq__.

61. What is a function in Python?
Reusable block of code that performs a task.

62. How do you define a function?
Use def keyword with name, parameters, body.

63. What is the use of return?
Sends a result back to caller.

64. What are *args and **kwargs?
*args for variable positional arguments, **kwargs for keyword arguments.

65. What is a default argument?
Function parameter with a preset value.

66. What is a keyword-only argument?
Argument that must be passed by name, not position.

67. What is a lambda function?
Anonymous one-line function created with lambda.

68. What is the difference between yield and return?
return ends function, yield returns generator values one at a time.

69. What is the scope of variables in functions?
Local, Enclosing, Global, Built-in (LEGB rule).

70. What are global and nonlocal keywords?
global accesses global variable, nonlocal accesses enclosing scope variable.

71. How do you pass a list to a function?
By giving list name as argument.

72. What is recursion?
Function calling itself until base case is met.

73. What is a higher-order function?
Function that takes/returns another function.

74. What are map(), filter(), and reduce()?

map applies function, filter selects items, reduce aggregates values.

75. What is the use of enumerate() in functions?
Gives index with element when iterating.

76. What is a closure in Python?
Function remembering variables from enclosing scope.

77. What is a decorator?
Function that modifies another function's behavior.

78. Can a function return another function?
Yes, functions are first-class objects.

79. How to define an anonymous function?
Using lambda keyword.

80. What is function caching in Python?
Storing results of expensive calls for reuse (functools.lru_cache).

81. What is the difference between a list and a tuple?
List is mutable, tuple is immutable.

82. What are the key features of lists?
Ordered, mutable, allow duplicates.

83. What are list comprehensions?
Concise way to create lists using loops in one line.

84. What is a dictionary?
Key-value data structure, unordered, mutable.

85. What is the difference between a list and a set?
List allows duplicates, set stores unique elements.

86. What is a set? What are its use cases?
Unordered collection of unique items, useful for membership tests.

87. How do you remove duplicates from a list?
Convert list to set, then back to list.

88. What are nested lists?
Lists inside other lists.

89. How do you merge two dictionaries?
Use dict1.update(dict2) or {**d1, **d2}.

90. How to sort a list or dictionary?

Use sorted() for lists, sorted(dict.items()) for dict.

91. What is a defaultdict?
dict subclass providing default values for missing keys.

92. What is a Counter from the collections module?
Dict subclass that counts elements.

93. What is a namedtuple?
Tuple subclass with named fields.

94. What is the difference between pop(), remove(), and del?
pop removes by index, remove by value, del deletes reference.

95. What is the use of slicing?
Extracts sub-parts of sequences like list[1:4].

96. What is object-oriented programming?
Programming paradigm using objects and classes.

97. What are classes and objects in Python?
Class is a blueprint, object is an instance of class.

98. What is __init__()?
Constructor method called when object is created.

99. What is self in a class?
Reference to the current object instance.

100. What is the difference between instance variables and class variables?
Instance variables are per object, class variables are shared across class.