Concept of Problem Solving

1. **What is the first step in problem-solving?**
   a) Writing the program
   b) Understanding the problem
   c) Debugging the program
   d) Testing the solution

   Answer: b) Understanding the problem

2. **Which of the following is NOT a characteristic of a good problem-solving approach?**
   a) Efficiency
   b) Complexity
   c) Clarity
   d) Accuracy

   Answer: b) Complexity

3. **What does "algorithm" mean in programming?**
   a) A programming language
   b) A graphical representation of a problem
   c) A step-by-step procedure for solving a problem
   d) An error in the program

   Answer: c) A step-by-step procedure for solving a problem

4. **A problem that cannot be broken into smaller parts is called:**
   a) Modular problem
   b) Atomic problem
   c) Complex problem
   d) Simple problem

   Answer: b) Atomic problem

5. **Which tool is commonly used for planning solutions in problem-solving?**
   a) Compiler
   b) Debugger
   c) Flowchart
   d) Editor

   Answer: c) Flowchart

6. **Which of the following is a systematic approach to solving problems?**
   a) Guessing the solution
   b) Following a random sequence

c) Using a structured process like algorithms or flowcharts
d) Ignoring problem constraints

Answer: c) Using a structured process like algorithms or flowcharts

7. **Which of these is an example of a heuristic approach to problem-solving?**
a) Finding the shortest path using Dijkstra's algorithm
b) Trial and error to identify a solution
c) Using a fixed formula
d) Writing code directly without a plan

Answer: b) Trial and error to identify a solution

8. **What is a 'well-defined problem'?**
a) A problem with unclear goals
b) A problem that has precise input, process, and output requirements
c) A problem that requires guessing
d) A problem without constraints

Answer: b) A problem that has precise input, process, and output requirements

9. **In problem-solving, constraints refer to:**
a) Random inputs to the problem
b) Rules or conditions that must be followed
c) Steps to debug the code
d) The programming language used

Answer: b) Rules or conditions that must be followed

10. **What is the final step in the problem-solving process?**
a) Testing the solution
b) Defining the problem
c) Designing the algorithm
d) Writing pseudocode

Answer: a) Testing the solution

11. **A clear problem definition ensures:**
a) More errors in the solution
b) Faster coding without understanding the problem
c) Better communication between stakeholders and developers
d) Reducing the time spent on design

Answer: c) Better communication between stakeholders and developers

12. **Which of the following is an example of a poorly defined problem?**
   a) Finding the sum of two given numbers
   b) Writing a program to manage inventory without specifying inputs or outputs
   c) Designing a calculator with defined functionality
   d) Writing a program to sort a list

   Answer: b) Writing a program to manage inventory without specifying inputs or outputs

13. **Why is it important to identify inputs in problem definition?**
   a) To reduce execution time
   b) To understand what data is required for processing
   c) To identify the programming language
   d) To design better hardware

   Answer: b) To understand what data is required for processing

14. **When defining a problem, understanding the 'output' means:**
   a) Knowing what the program should produce as a result
   b) Deciding on the hardware for the program
   c) Debugging the code
   d) Ignoring the constraints

   Answer: a) Knowing what the program should produce as a result

15. **Which of the following is an essential part of a problem definition?**
   a) Input data, processing steps, and desired output
   b) Only the input data
   c) The choice of programming language
   d) Ignoring constraints

   Answer: a) Input data, processing steps, and desired output

Problem Definition

16. **What is problem definition in program planning?**
   a) Writing code
   b) Documenting the steps for solving the problem
   c) Clearly defining the problem's inputs, outputs, and constraints
   d) Testing the program

   Answer: c) Clearly defining the problem's inputs, outputs, and constraints

17. **Which of these is NOT part of problem definition?**
   a) Input
   b) Output

c) Memory location
d) Constraints

Answer: c) Memory location

18. **What is the purpose of defining a problem clearly?**
a) To avoid debugging
b) To minimize coding time
c) To ensure that the program solves the intended problem
d) To create flowcharts

Answer: c) To ensure that the program solves the intended problem

19. **In problem-solving, identifying assumptions is important because:**
a) It simplifies coding
b) It reduces errors
c) It clarifies limitations of the problem
d) It speeds up the execution

Answer: c) It clarifies limitations of the problem

20. **Which element is part of a well-defined problem?**
a) Undefined inputs
b) Clear objectives
c) Random constraints
d) Ambiguous results

Answer: b) Clear objectives

Program Design

21. **Program design involves which of the following steps?**
a) Execution
b) Implementation
c) Planning the solution and writing an algorithm
d) Debugging

Answer: c) Planning the solution and writing an algorithm

22. **Which of these is a common design tool?**
a) Compiler
b) Flowchart
c) Assembler
d) Editor

Answer: b) Flowchart

23. **What is pseudocode?**
    a) A programming language
    b) A way to describe algorithms using natural language and symbols
    c) A graphical representation of a program
    d) A debugging tool

    Answer: b) A way to describe algorithms using natural language and symbols

24. **Which of the following is NOT a characteristic of a good program design?**
    a) Scalability
    b) Simplicity
    c) Efficiency
    d) Redundancy

    Answer: d) Redundancy

25. **Which phase comes after program design in the software development life cycle?**
    a) Problem definition
    b) Testing
    c) Coding
    d) Debugging

    Answer: c) Coding

26. **Which of these is the first step in program design?**
    a) Writing code
    b) Understanding the problem and defining objectives
    c) Testing the program
    d) Debugging the solution

    Answer: b) Understanding the problem and defining objectives

27. **What is the purpose of using pseudocode in program design?**
    a) To write executable code
    b) To document the program
    c) To plan and describe the logic of the program in simple terms
    d) To define inputs

    Answer: c) To plan and describe the logic of the program in simple terms

28. **Modular programming involves dividing a problem into:**
    a) Irregular parts
    b) Smaller, independent sub-problems or modules

c) Complex loops
d) Debugging tools

Answer: b) Smaller, independent sub-problems or modules

29. **What is the primary goal of program design?**
a) To write efficient and readable code
b) To skip problem analysis
c) To write only complex programs
d) To avoid errors during testing

Answer: a) To write efficient and readable code

30. **A top-down design approach:**
a) Starts with the details and builds up to the main problem
b) Focuses on solving one big problem at once
c) Breaks a problem into smaller, more manageable parts
d) Avoids planning and starts coding

Answer: c) Breaks a problem into smaller, more manageable parts

Flowcharts

31. **What shape is used to represent a decision in a flowchart?**
a) Rectangle
b) Oval
c) Diamond
d) Circle

Answer: c) Diamond

32. **Which shape in a flowchart represents a process or action?**
a) Rectangle
b) Oval
c) Parallelogram
d) Diamond

Answer: a) Rectangle

33. **What does an arrow indicate in a flowchart?**
a) A calculation
b) A connection between steps
c) An error
d) A condition

Answer: b) A connection between steps

34. **Which of these best describes the use of flowcharts?**
    a) They execute the program
    b) They visually represent the steps in a process
    c) They debug code
    d) They generate pseudocode

    Answer: b) They visually represent the steps in a process

35. **Why are flowcharts used in program planning?**
    a) To avoid errors
    b) To visually organize the sequence of operations
    c) To write code efficiently
    d) To reduce programming time

    Answer: b) To visually organize the sequence of operations

---

Algorithms

36. **What is a key property of a good algorithm?**
    a) Ambiguity
    b) Infinite loops
    c) Clarity
    d) Redundancy

    Answer: c) Clarity

37. **Which of the following is NOT an algorithm design technique?**
    a) Divide and conquer
    b) Debugging
    c) Dynamic programming
    d) Greedy approach

    Answer: b) Debugging

38. **An algorithm must always have:**
    a) A single solution
    b) A finite number of steps
    c) A complex structure
    d) Multiple inputs

    Answer: b) A finite number of steps

39. **What is the purpose of dry running an algorithm?**
    a) To check for syntax errors

b) To manually trace its steps and verify its logic
c) To execute the program
d) To debug code

Answer: b) To manually trace its steps and verify its logic

40. **Which is NOT a part of an algorithm?**
a) Input
b) Process
c) Output
d) Compilation

Answer: d) Compilation

Miscellaneous

41. **What does modular programming involve?**
a) Writing all code in one function
b) Dividing the program into small, manageable parts
c) Avoiding functions
d) Ignoring flowcharts

Answer: b) Dividing the program into small, manageable parts

42. **What is the benefit of using structured programming?**
a) Makes programs complex
b) Simplifies debugging and maintenance
c) Eliminates the need for testing
d) Improves execution speed

Answer: b) Simplifies debugging and maintenance

43. **In top-down design, the problem is divided into:**
a) Simple inputs
b) Logical errors
c) Smaller, manageable sub-problems
d) Complex loops

Answer: c) Smaller, manageable sub-problems

44. **What is the primary goal of problem-solving in programming?**
a) Writing the code
b) Finding the fastest algorithm

c) Delivering a solution that meets the requirements
d) Compiling the program

Answer: c) Delivering a solution that meets the requirements

45. **Debugging is the process of:**
a) Writing algorithms
b) Finding and fixing errors in the code
c) Testing program efficiency
d) Planning program design

Answer: b) Finding and fixing errors in the code

Python History and Features

42. Who developed Python programming language?
a) James Gosling
b) Guido van Rossum
c) Dennis Ritchie
d) Bjarne Stroustrup

Answer: b) Guido van Rossum

43. In which year was Python developed?
a) 1989
b) 1991
c) 1995
d) 2000

Answer: b) 1991

44. Python was inspired by which programming language?
a) ABC
b) Java
c) C++
d) Perl

Answer: a) ABC

45. What does Python emphasize?
a) Code readability and simplicity
b) Execution speed
c) Complex syntax
d) Strict typing

Answer: a) Code readability and simplicity

46. Which of the following is NOT a feature of Python?
a) Interpreted language
b) Easy to learn
c) Platform dependent
d) Open source

Answer: c) Platform dependent

47. What is the meaning of “Python” in the context of the language's name?
a) Named after the Python snake
b) Named after Monty Python comedy group
c) Acronym for a technical term
d) Randomly chosen

Answer: b) Named after Monty Python comedy group

48. What is the file extension of Python scripts?
a) .java
b) .py
c) .pl
d) .exe

Answer: b) .py

49. What was the first version of Python released?
a) Python 1.0
b) Python 2.0
c) Python 3.0
d) Python 0.9.0

Answer: d) Python 0.9.0

50. Which Python version discontinued support in 2020?
a) Python 3.7
b) Python 2.7
c) Python 3.0
d) Python 1.5

Answer: b) Python 2.7

51. Which organization currently manages Python?
a) Python Software Foundation
b) Microsoft
c) Oracle
d) Google

Answer: a) Python Software Foundation

Python Versions

52. What major change was introduced in Python 3.x compared to Python 2.x?
a) Removal of print as a statement
b) Inclusion of type hints
c) New syntax for loops
d) Faster execution

Answer: a) Removal of print as a statement

53. What is the latest Python version (as of 2025)?
a) Python 3.9
b) Python 3.10
c) Python 3.11
d) Python 3.12

Answer: d) Python 3.12

54. When was Python 3.x released?
a) 2000
b) 2005
c) 2008
d) 2010

Answer: c) 2008

55. What feature was introduced in Python 3.6?
a) F-strings for formatted string literals
b) Print as a function
c) List comprehensions
d) Dictionaries

Answer: a) F-strings for formatted string literals

56. Which of the following is deprecated in Python 3.x?
a) Raw_input()
b) Input()
c) Open()
d) Len()

Answer: a) Raw_input()

Python Basics

57. What type of language is Python?
a) Compiled
b) Interpreted
c) Both compiled and interpreted
d) None of the above

Answer: b) Interpreted

58. How do you execute a Python script?
a) python script.py
b) execute script.py
c) run script.py
d) start script.py

Answer: a) python script.py

59. Which of the following is a valid variable name in Python?
a) 123var
b) _var123
c) var-123
d) var 123

Answer: b) _var123

60. Which keyword is used to define a function in Python?
a) fun
b) def
c) func
d) define

Answer: b) def

61. Which of these data types is immutable in Python?
a) List
b) Dictionary
c) String
d) Set

Answer: c) String

The `print` Statement

62. What is the correct syntax for using the `print` function in Python 3?
a) print "Hello"
b) print("Hello")
c) echo "Hello"
d) Print("Hello")

Answer: b) print("Hello")

63. What does the `sep` parameter in `print` do?
a) Separates lines of text
b) Specifies the separator between arguments
c) Adds a newline
d) Formats strings

Answer: b) Specifies the separator between arguments

64. What is the default value of the `end` parameter in `print`?
a) A space
b) A tab
c) A newline (`\n`)
d) None

Answer: c) A newline (`\n`)

65. How can you print without a newline at the end?
a) Use `end=" "` in the print statement
b) Use `sep=""`
c) Use `newline=False`
d) Use `no_newline=True`

Answer: a) Use `end=" "` in the print statement

66. What is the output of `print("A", "B", sep="-")`?
a) A B
b) A-B
c) A B-
d) -A-B

Answer: b) A-B

Error Handling with `print`

67. What happens if you miss parentheses in a `print` statement in Python 3?

a) It executes normally
b) It raises a `SyntaxError`
c) It raises a `NameError`
d) It converts to Python 2 syntax

Answer: b) It raises a `SyntaxError`

68. What is the output of `print("Hello", end="")`?

a) Hello
b) Hello\n
c) Hello None
d) None

Answer: a) Hello

69. What is the purpose of `file` parameter in `print`?

a) Directs output to a specified file or stream
b) Reads input from a file
c) Appends data to a string
d) Closes the file

Answer: a) Directs output to a specified file or stream

70. Which of these results in a `TypeError` for the `print` function?

a) print(1, 2, 3)
b) print("Hello", end=123)
c) print("Hello" + 123)
d) print()

Answer: c) print("Hello" + 123)

71. What is the output of `print("The value is", 5, sep=":")`?

a) The value is 5
b) The value:5
c) The:value:is:5
d) The:value:is 5

Answer: b) The value:5

Miscellaneous

72. What function is used for formatted string output?

a) printf()

b) format()
c) write()
d) append()

Answer: b) format()

73. Which is used to comment a single line in Python?
a) //
b) /* */
c) #
d) <!-- -->

Answer: c) #

74. What is the output of `print(len("Python"))`?
a) 5
b) 6
c) 7
d) Error

Answer: b) 6

75. What is the output of `print("Hello " * 3)`?
a) HelloHelloHello
b) Hello 3
c) Hello Hello Hello
d) Error

Answer: c) Hello Hello Hello

76. What is the output of `print(10/3)`?
a) 3
b) 3.0
c) 3.33...
d) Error

Answer: c) 3.33...

77. Which tool is commonly used for breaking down complex problems?
a) Editor
b) Debugger
c) Flowchart
d) Compiler

Answer: c) Flowchart

78. Which of the following is an iterative process in problem-solving?
a) Debugging
b) Refining the algorithm and design after testing
c) Choosing a programming language
d) Writing documentation

Answer: b) Refining the algorithm and design after testing

79. The role of abstraction in problem-solving is to:
a) Eliminate errors in programming
b) Focus on the main problem while ignoring unnecessary details
c) Make the problem more complex
d) Avoid modularity

Answer: b) Focus on the main problem while ignoring unnecessary details

80. What is the role of testing in problem-solving?
a) To debug only minor errors
b) To validate the correctness and performance of the solution
c) To rewrite the problem definition
d) To replace program design

Answer: b) To validate the correctness and performance of the solution

81. Which is NOT an objective of program design?
a) Simplicity
b) Scalability
c) Ambiguity
d) Efficiency

Answer: c) Ambiguity

Multiple Choice Questions (MCQs) on Debugging in Python

82. What is debugging in Python?

A) Writing new code from scratch
B) Fixing errors in the code
C) Running the code without errors
D) Converting Python code to another language

Answer: B) Fixing errors in the code

83. Which of the following tools is commonly used for debugging in Python?

A) PIP
B) PyCharm Debugger
C) Python Debugger (pdb)
D) Both B and C

Answer: D) Both B and C

84. What is the default command to start debugging in Python using pdb?

A) `import debug`
B) `import pdb; pdb.set_trace()`
C) `debug.start()`
D) `python debug.py`

Answer: B) `import pdb; pdb.set_trace()`

85. What type of errors does debugging primarily help to fix?

A) Logical errors
B) Syntax errors
C) Runtime errors
D) All of the above

Answer: D) All of the above

86. Which Python statement is used to handle exceptions while debugging?

A) `if-else`
B) `try-except`
C) `switch-case`
D) `for-loop`

Answer: B) `try-except`

87. What does the `pdb` command `n` (next) do during debugging?

A) Steps into the function call
B) Executes the next line of code
C) Prints the next line of code
D) Terminates the debugging session

Answer: B) Executes the next line of code

88. What will happen if you use `assert` in Python debugging?

A) It will throw an exception if the condition is False
B) It will always execute without error
C) It will print debugging messages
D) It will log errors in a file

Answer: A) It will throw an exception if the condition is False

89. What is the main advantage of using a debugger instead of print statements for debugging?

A) Debuggers execute faster
B) Debuggers allow step-by-step execution
C) Debuggers can only detect syntax errors
D) Debuggers remove errors automatically

Answer: B) Debuggers allow step-by-step execution

90. Which of the following can be used to log errors while debugging?

A) `print()`
B) `logging` module
C) `sys.exit()`
D) `os.log()`

Answer: B) `logging` module

91. How can you ignore exceptions while debugging in Python?

A) Use `pass` in `try-except` block
B) Use `break` in `if-else` block
C) Use `continue` in a loop
D) Use `return` in a function

Answer: A) Use `pass` in `try-except` block

Multiple Choice Questions (MCQs) on Syntax Errors in Python

92. What is a syntax error in Python?

A) An error due to incorrect logic
B) An error due to incorrect grammar of the language
C) An error that occurs only at runtime
D) An error that cannot be detected by the interpreter

Answer: B) An error due to incorrect grammar of the language

---

93. When do syntax errors occur in Python?

A) During program execution
B) During program compilation
C) Before the program runs
D) After program termination

Answer: C) Before the program runs

---

94. . What will happen if there is a syntax error in the Python code?

A) The program will execute with incorrect output
B) The program will run but give warnings
C) The program will not execute at all
D) The program will automatically fix the error

Answer: C) The program will not execute at all

---

95. What does Python display when it encounters a syntax error?

A) Warning message
B) Debugging message
C) SyntaxError message
D) Logical error message

Answer: C) SyntaxError message

---

96. Which of the following will result in a syntax error?

A) `print("Hello World")`
B) `if x == 5:` (without indentation)
C) `def func(): return 5`
D) `x = 5 + 3`

Answer: B) `if x == 5:` (without indentation)

---

97. What is the cause of the syntax error in the following statement?

```
print "Hello"
```

A) Missing parentheses
B) Incorrect indentation
C) Mismatched quotation marks
D) Wrong function name

Answer: A) Missing parentheses

---

98. What is wrong with this code?

```
if 5 > 3
    print("Hello")
```

A) Incorrect indentation
B) Missing colon (:) after `if`
C) Missing parentheses in `print`
D) `if` should be `IF`

Answer: B) Missing colon (:) after `if`

---

99. Which of the following is a valid Python statement?

A) `for i in range(5) print(i)`
B) `if x = 5:`
C) `while True: pass`
D) `def func: return 5`

Answer: C) `while True: pass`

---

100. What is the cause of the syntax error in the following code?

```
def greet(name)
    print("Hello", name)
```

A) Missing parentheses in `print()`
B) Missing colon `:` in function definition
C) Missing return statement
D) Incorrect function name

Answer: B) Missing colon `:` in function definition

---

101. Which of the following statements is incorrect and will cause a syntax error?

A) `x = 5`
B) `if x == 5 print("Hello")`
C) `for i in range(5): print(i)`
D) `while True: break`

Answer: B) `if x == 5 print("Hello")`

---

102. What will cause a syntax error in Python?

A) Using an undefined variable
B) Using incorrect indentation
C) Dividing by zero
D) Using a very large number

Answer: B) Using incorrect indentation

---

103. Identify the syntax error in this statement:

```
if (x > 10) then:
    print("x is greater")
```

A) `then` is not required in Python
B) Missing parentheses in `print()`
C) Incorrect indentation
D) `if` should be uppercase

Answer: A) `then` is not required in Python

---

104. Which of the following will NOT cause a syntax error?

A) `x = 5 +`
B) `if x == 5:`
C) `print("Hello"`
D) `for i in range(5) print(i)`

Answer: B) `if x == 5:`

---

105. What is wrong with this loop?

```
for i in range(5)
    print(i)
```

A) Missing colon `:` after `range(5)`
B) `print(i)` should be inside parentheses
C) `for` should be capitalized
D) `i` should be initialized

Answer: A) Missing colon `:` after `range(5)`

---

106. How can you correct the syntax error in this function definition?

```
def add(a, b)
    return a + b
```

A) Add a colon `:` after `(a, b)`
B) Use `return a + b` inside an `if` block
C) Remove the parentheses
D) Add an `else` statement

Answer: A) Add a colon `:` after `(a, b)`

---

107. Which of the following statements will cause a syntax error?

A) `print("Hello World")`
B) `if x == 10 print("x is 10")`

C) `while True: pass`
D) `x = 100`

Answer: B) `if x == 10 print("x is 10")`

---

108. What is the issue with this function call?

```
print "Hello, World!"
```

A) Missing parentheses
B) Incorrect function name
C) `print` should be capitalized
D) `Hello, World!` should be in single quotes

Answer: A) Missing parentheses

---

109. What is wrong with this assignment statement?

```
if x = 5:
    print("x is 5")
```

A) = should be ==
B) `print` should be in uppercase
C) `if` should be capitalized
D) The colon : should be removed

Answer: A) = should be ==

---

110. What will happen if you use mismatched parentheses in an expression?

```
print("Hello"
```

A) The program will run normally
B) A `SyntaxError` will be raised
C) The program will ignore the extra parenthesis
D) The output will be `Hello`

Answer: B) A `SyntaxError` will be raised

---

111. What will cause a syntax error in Python?

A) Using `print` with parentheses
B) Declaring a variable with a number first (e.g., `1var = 10`)
C) Using `if-else` statements
D) Using `while` loops

Answer: B) Declaring a variable with a number first (e.g., `1var = 10`)

Multiple Choice Questions (MCQs) on Runtime Errors in Python

---

112. What is a runtime error in Python?

A) An error that occurs before the program runs
B) An error that occurs during the execution of the program
C) An error due to incorrect indentation
D) An error that prevents compilation

Answer: B) An error that occurs during the execution of the program

---

113. Which of the following is an example of a runtime error?

A) `SyntaxError`
B) `ZeroDivisionError`
C) `IndentationError`
D) `TypeError`

Answer: B) `ZeroDivisionError`

---

114. What will happen when the following code runs?

```
x = 10 / 0
```

A) It will execute successfully
B) It will throw a `ZeroDivisionError`
C) It will return `None`
D) It will print `0`

Answer: B) It will throw a `ZeroDivisionError`

---

115. What type of error will this code produce?

```
print(10 + "5")
```

A) `SyntaxError`
B) `NameError`
C) `TypeError`
D) `IndexError`

Answer: C) `TypeError`

---

116. What happens if a variable is used before being assigned a value?

A) `SyntaxError`
B) `TypeError`
C) `NameError`
D) `KeyError`

Answer: C) `NameError`

---

117. What error occurs when you try to access an index that doesn't exist in a list?

A) `KeyError`
B) `ValueError`
C) `IndexError`
D) `AttributeError`

Answer: C) `IndexError`

---

118. What will happen when the following code is executed?

```
x = int("Hello")
```

A) `SyntaxError`
B) `TypeError`
C) `ValueError`
D) `KeyError`

Answer: C) `ValueError`

---

119. Which of the following will cause a `KeyError`?

A) Accessing a dictionary key that does not exist
B) Dividing by zero
C) Using an undefined variable
D) Using a variable of the wrong type

Answer: A) Accessing a dictionary key that does not exist

---

120. What type of error occurs when trying to open a file that does not exist?

A) `FileNotFoundError`
B) `ImportError`
C) `SyntaxError`
D) `KeyError`

Answer: A) `FileNotFoundError`

---

121. What will happen if you try to import a module that does not exist?

A) `SyntaxError`
B) `ModuleNotFoundError`
C) `TypeError`
D) `KeyError`

Answer: B) `ModuleNotFoundError`

---

122. What happens when an infinite recursion occurs in a Python program?

A) The program runs forever
B) The program crashes with a `RecursionError`
C) The program automatically stops after a few iterations
D) No error occurs

Answer: B) The program crashes with a `RecursionError`

---

123. Which error is raised when trying to call a method that does not exist for an object?

A) `TypeError`
B) `AttributeError`
C) `IndexError`
D) `ValueError`

Answer: B) `AttributeError`

---

124. What will be the output of this code?

```
d = {"name": "Pramod"}
print(d["age"])
```

A) `"None"`
B) `"0"`
C) `KeyError`
D) `IndexError`

Answer: C) `KeyError`

---

125. What will happen when executing the following code?

```
f = open("non_existent_file.txt", "r")
```

A) It will create a new file
B) It will throw a `FileNotFoundError`
C) It will return `None`
D) It will print an empty string

Answer: B) It will throw a `FileNotFoundError`

---

126. What type of error occurs when an operation is performed on an unsupported data type?

A) `SyntaxError`
B) `ValueError`
C) `TypeError`
D) `IndexError`

Answer: C) `TypeError`

---

127. What happens when a function returns a value but the caller treats it as a different type?

A) `ValueError`
B) `TypeError`
C) `SyntaxError`
D) `IndexError`

Answer: B) `TypeError`

---

128. What type of error occurs when a function is called with the wrong number of arguments?

A) `TypeError`
B) `SyntaxError`
C) `KeyError`
D) `AttributeError`

Answer: A) `TypeError`

---

129. What will happen when this code is executed?

```
try:
    x = 1 / 0
except ValueError:
    print("Value error occurred")
```

A) `"Value error occurred"` will be printed
B) `ZeroDivisionError` will be raised
C) The program will continue without error
D) `"ZeroDivisionError handled"` will be printed

Answer: B) `ZeroDivisionError` will be raised

---

130. What is the best way to handle runtime errors in Python?

A) Ignoring the error
B) Using `try-except` blocks
C) Debugging with print statements only
D) Restarting the program manually

Answer: B) Using `try-except` blocks

---

131. Which of the following will NOT cause a runtime error?

A) `print("Hello" + 5)`
B) `print(10 / 2)`
C) `int("hello")`
D) `open("unknown.txt", "r")`

Answer: B) `print(10 / 2)`

MCQs on Logical Errors in Python

---

132. What is a logical error in Python?

A) An error that occurs due to incorrect syntax
B) An error that causes a program to crash
C) An error that allows the program to run but gives incorrect results
D) An error that occurs only at runtime

Answer: C) An error that allows the program to run but gives incorrect results

---

133. Which of the following best describes a logical error?

A) The program does not compile
B) The program runs but produces an incorrect output
C) The program throws an exception
D) The program crashes before execution

Answer: B) The program runs but produces an incorrect output

---

134. Which of the following is an example of a logical error?

A) Using = instead of == in a condition
B) Dividing by zero
C) Accessing an undefined variable
D) Incorrect indentation

Answer: A) Using = instead of == in a condition

---

135. What is the best way to detect logical errors in a Python program?

A) Using a debugger
B) Using `try-except` blocks
C) Running the program and verifying the output manually
D) Relying on Python to detect them automatically

Answer: C) Running the program and verifying the output manually

---

136. What will be the output of the following code?

```
x = 10
y = 5
if x > y:
    print("x is smaller than y")
```

A) `"x is smaller than y"`
B) `"x is greater than y"`
C) `SyntaxError`
D) No output

Answer: A) `"x is smaller than y"` (which is incorrect due to a logical error)

---

137. What type of error is present in the following code?

```
def add(a, b):
    return a - b
print(add(5, 3))
```

A) Syntax error
B) Logical error
C) Runtime error
D) Name error

Answer: B) Logical error (subtraction is used instead of addition)

---

138. How can logical errors be avoided in Python?

A) Writing clear and structured code
B) Using print statements for debugging
C) Performing proper testing
D) All of the above

Answer: D) All of the above

---

139. What will be the output of the following code?

```
for i in range(5):
    print(i + 1)
```

A) 0 1 2 3 4
B) 1 2 3 4 5
C) 1 3 5 7 9
D) None of the above

Answer: B) 1 2 3 4 5 (If the intended output was 0 to 4, then this is a logical error)

---

140. Which of the following statements is true about logical errors?

A) Logical errors are easier to find than syntax errors
B) Logical errors are detected by the Python interpreter
C) Logical errors can only be detected by testing the program output
D) Logical errors prevent a program from executing

Answer: C) Logical errors can only be detected by testing the program output

---

141. What kind of error does the following code contain?

```
def is_even(num):
    return num % 2 == 1

print(is_even(4))
```

A) Syntax error
B) Logical error
C) Runtime error
D) No error

Answer: B) Logical error (the condition should be `num % 2 == 0` to check for even numbers)

Multiple Choice Questions (MCQs) on Semantic Errors

Semantic errors occur when the program runs without crashing but produces incorrect or unintended results due to incorrect logic or meaning in the code.

---

142. What is a semantic error in Python?

A) An error due to incorrect syntax
B) An error that prevents the program from running
C) An error where the program runs but produces incorrect results
D) An error detected by the Python interpreter

Answer: C) An error where the program runs but produces incorrect results

---

143. Which of the following is an example of a semantic error?

A) Using = instead of == in a condition
B) Forgetting to close a parenthesis
C) Dividing by zero
D) Using an undefined variable

Answer: A) Using = instead of == in a condition

---

144. What will be the output of the following code?

```
def multiply(a, b):
    return a + b

print(multiply(3, 4))
```

A) `7`
B) `12`
C) `Error`
D) `None`

Answer: A) `7` (Semantic error: the function should multiply, but it adds instead)

---

145. How can semantic errors be detected?

A) By the Python interpreter
B) By running the program and checking the output
C) By using `try-except` blocks
D) By syntax highlighting in an IDE

Answer: B) By running the program and checking the output

---

146. What kind of error is in the following code?

```
def area_of_rectangle(length, width):
    return 2 * (length + width)

print(area_of_rectangle(5, 4))
```

A) Syntax error
B) Runtime error
C) Semantic error
D) No error

Answer: C) Semantic error (The formula should be `length * width` instead of `2 * (length + width)`)

---

147. What will be the output of the following code?

```
x = 10
y = 5
if x < y:
    print("x is greater than y")
else:
    print("x is smaller than y")
```

A) `"x is greater than y"`
B) `"x is smaller than y"`
C) No output
D) `Error`

Answer: B) `"x is smaller than y"` (Semantic error: incorrect message for the condition)

---

148. How do semantic errors differ from syntax errors?

A) Semantic errors prevent the code from running, but syntax errors do not
B) Syntax errors cause runtime crashes, while semantic errors do not
C) Semantic errors produce incorrect results but do not stop the program from running
D) There is no difference between them

Answer: C) Semantic errors produce incorrect results but do not stop the program from running

---

149. Which of the following is NOT a semantic error?

A) Using the wrong formula in a calculation
B) Using an incorrect comparison operator
C) Forgetting a colon at the end of a function definition
D) Assigning an incorrect value to a variable

Answer: C) Forgetting a colon at the end of a function definition (This is a syntax error, not a semantic error)

---

150. What will happen when the following code runs?

```
def is_odd(num):
    return num % 2 == 0

print(is_odd(5))
```

A) `True`
B) `False`
C) `Error`
D) `None`

Answer: B) `False` (Semantic error: The function is named `is_odd`, but it checks for even numbers)

---

151. What is the best way to avoid semantic errors?

A) Using `try-except` blocks
B) Testing the program thoroughly
C) Using syntax highlighting in an editor
D) Relying on Python to detect them automatically

Answer: B) Testing the program thoroughly

MCQs on Type Errors in Python

A **TypeError** in Python occurs when an operation or function is applied to an object of an inappropriate type.

---

152. What is a `TypeError` in Python?

A) An error that occurs due to incorrect indentation
B) An error when an operation is performed on an incompatible data type
C) An error caused by a missing import statement
D) An error caused by accessing an undefined variable

Answer: B) An error when an operation is performed on an incompatible data type

---

153. Which of the following will raise a `TypeError`?

A) `print(10 / 2)`
B) `print("10" + 5)`
C) `print([1, 2, 3] + [4, 5, 6])`
D) `print("Hello" * 3)`

Answer: B) `print("10" + 5)` (Cannot concatenate a string with an integer)

---

154. What will be the output of the following code?

```
x = "Hello"
y = x + 5
print(y)
```

A) `"Hello5"`
B) `5Hello`
C) `Hello 5`
D) `TypeError`

Answer: D) `TypeError` (Cannot add a string and an integer)

---

155. Which of the following function calls will NOT result in a `TypeError`?

A) `int("42")`
B) `len(100)`

C) `5 + "hello"`
D) `sum(["1", "2", "3"])`

Answer: A) `int("42")` (Correctly converts a string to an integer)

---

156. What will happen when executing the following code?

```
x = [1, 2, 3]
y = x * "2"
print(y)
```

A) `[1, 2, 3, 1, 2, 3]`
B) `TypeError`
C) `"2,2,2"`
D) `None`

Answer: B) `TypeError` (Cannot multiply a list by a string)

---

157. What is the cause of the `TypeError` in the following code?

```
def add_numbers(a, b):
    return a + b

print(add_numbers(10, "5"))
```

A) The function is not defined properly
B) Python does not support functions
C) String and integer cannot be added directly
D) The function is missing a return statement

Answer: C) String and integer cannot be added directly

---

158. Which of the following expressions will raise a `TypeError`?

A) `print(len([1, 2, 3]))`
B) `print(len("Hello"))`
C) `print(len(42))`
D) `print(len({"a": 1, "b": 2}))`

Answer: C) `print(len(42))` (Integers do not have a length)

---

159. What is the output of the following code?

```
x = (1, 2, 3)
x[0] = 10
print(x)
```

A) `(10, 2, 3)`
B) `TypeError`
C) `None`
D) `(1, 2, 3, 10)`

Answer: B) `TypeError` (Tuples are immutable and cannot be modified)

---

160. Which of the following will cause a `TypeError`?

A) `x = float("3.14")`
B) `y = str(10)`
C) `z = 10 + "20"`
D) `a = list((1, 2, 3))`

Answer: C) `z = 10 + "20"` (Cannot add an integer and a string)

---

161. What will be the output of the following code?

```
print(abs("10"))
```

A) `10`
B) `-10`
C) `TypeError`
D) `None`

Answer: C) `TypeError` (The `abs()` function only works on numbers, not strings)

MCQs on NameError in Python

---

162. What is a `NameError` in Python?

A) An error due to incorrect syntax
B) An error when an undefined variable or function is accessed
C) An error that occurs during type mismatches
D) An error caused by dividing by zero

Answer: B) An error when an undefined variable or function is accessed

---

163. Which of the following will raise a `NameError`?

A) `print("Hello, World!")`
B) `x = 10; print(x)`
C) `print(y)` (when `y` is not defined)
D) `len([1, 2, 3])`

Answer: C) `print(y)` (when `y` is not defined)

---

164. What will be the output of the following code?

```
def my_function():
    print(x)

my_function()
```

A) `None`
B) `NameError`
C) `0`
D) `"x"`

Answer: B) `NameError` (Variable `x` is not defined inside or outside the function)

---

165. How can a `NameError` be avoided?

A) By ensuring variables are defined before use
B) By using `try-except` blocks
C) By checking variable names for typos
D) All of the above

Answer: D) All of the above

---

166. What happens when you try to access a variable that is not defined?

A) The program prints `None`
B) The program runs but ignores the undefined variable
C) A `NameError` is raised
D) The program crashes without an error message

Answer: C) A `NameError` is raised

---

167. What will happen when executing the following code?

```
print(name)
name = "Pramod"
```

A) `"Alice"`
B) `None`
C) `NameError`
D) `UndefinedVariableError`

Answer: C) `NameError` (Variable `name` is used before assignment)

---

168. What will be the output of the following code?

```
def greet():
    print(message)

message = "Hello"
greet()
```

A) `"Hello"`
B) `None`
C) `NameError`
D) `SyntaxError`

Answer: A) `"Hello"` (Since `message` is defined before the function is called)

---

169. Which of the following scenarios will NOT cause a `NameError`?

A) Using a variable before assigning it a value
B) Calling a function before defining it
C) Importing a module that does not exist
D) Accessing a variable inside a function when it's defined globally

Answer: D) Accessing a variable inside a function when it's defined globally

---

170. What will happen when executing the following code?

```
def my_func():
    x = 5
print(x)
```

A) `5`
B) `None`
C) `NameError`
D) `SyntaxError`

Answer: C) `NameError` (Variable `x` is defined inside the function and not accessible outside)

---

171. Which of the following will raise a `NameError`?

A) `x = 10; print(x)`
B) `print(len("Hello"))`
C) `print(math.sqrt(25))` (without importing `math`)
D) `list1 = [1,2,3]; print(list1[0])`

Answer: C) `print(math.sqrt(25))` (without importing `math`)

---

172. What will be the output of the following code?

```
def example():
    print(value)
    value = 10

example()
```

A) `10`
B) `None`
C) `NameError`
D) `UnboundLocalError`

Answer: D) `UnboundLocalError` (Variable `value` is referenced before assignment inside the function)

---

173. What should you do to fix a `NameError`?

A) Define the variable before using it
B) Check for typos in variable names
C) Import required modules before using them
D) All of the above

Answer: D) All of the above

---

174. What will be the output of the following code?

```
def display():
    print(number)

number = 20
display()
```

A) `20`
B) `None`
C) `NameError`
D) `RuntimeError`

Answer: A) `20` (The variable `number` is defined before calling the function)

---

175. What will be the output of the following code?

```
def test():
    print(value)

test()
value = 15
```

A) `15`
B) `None`
C) `NameError`
D) `SyntaxError`

Answer: C) `NameError` (Variable `value` is defined after the function call)

---

176. What happens when you try to access a global variable inside a function without declaring it first?

A) The function can access the variable normally
B) The function creates a new local variable
C) A `NameError` is raised
D) The function ignores the variable

Answer: A) The function can access the variable normally (if the variable is defined before function execution)

MCQs on IndexError in Python

---

177. What is an `IndexError` in Python?

A) An error that occurs when an operation is performed on an incompatible data type
B) An error that occurs when accessing an index that does not exist in a sequence
C) An error caused by missing parentheses in function calls
D) An error raised when dividing by zero

Answer: B) An error that occurs when accessing an index that does not exist in a sequence

---

178. Which of the following will raise an `IndexError`?

A) `my_list = [1, 2, 3]; print(my_list[1])`
B) `my_tuple = (10, 20, 30); print(my_tuple[3])`
C) `my_str = "Python"; print(my_str[2])`
D) `my_dict = {"a": 1, "b": 2}; print(my_dict["a"])`

Answer: B) `my_tuple = (10, 20, 30); print(my_tuple[3])` (Index 3 is out of range for the tuple)

---

179. What will be the output of the following code?

```
numbers = [10, 20, 30]
print(numbers[3])
```

A) `30`
B) `None`
C) `IndexError`
D) `0`

Answer: C) `IndexError` (Valid indices are `0, 1, 2`, but `3` is out of range)

---

180. Which of the following will NOT raise an `IndexError`?

A) `x = "hello"; print(x[4])`
B) `y = [1, 2, 3, 4]; print(y[2])`
C) `z = (5, 6, 7); print(z[3])`
D) `a = []; print(a[0])`

Answer: B) `y = [1, 2, 3, 4]; print(y[2])` (Index 2 is valid for this list)

---

181. What will be the output of the following code?

```
words = ["Python", "Java", "C++"]
print(words[-4])
```

A) `"Python"`
B) `"C++"`
C) `None`
D) `IndexError`

Answer: D) `IndexError` (Valid negative indices are `-1, -2, -3`; `-4` is out of range)

---

182. How can you avoid an `IndexError` when working with lists?

A) Always use positive indices
B) Ensure the index is within the valid range before accessing it
C) Avoid using lists in Python
D) Use `try-except` blocks to catch the error

Answer: B) Ensure the index is within the valid range before accessing it

---

183. What will happen when executing the following code?

```
my_list = [10, 20, 30]
print(my_list[-1])
```

A) `10`
B) `30`
C) `IndexError`
D) `None`

Answer: B) `30` (Negative indexing starts from the end, so `-1` refers to the last element)

---

184. What will be the output of the following code?

```
data = [5, 10, 15, 20]
index = len(data)
print(data[index])
```

A) `20`
B) `None`
C) `IndexError`
D) `5`

Answer: C) `IndexError` (Valid indices are `0` to `len(data) - 1`, but `len(data)` is out of range)

---

185. How can you safely access an element in a list without causing an `IndexError`?

A) Always use negative indices
B) Use the `try-except` block to handle `IndexError`
C) Avoid using lists
D) Use `break` statements in loops

Answer: B) Use the `try-except` block to handle `IndexError`

---

186. What will be the output of the following code?

```
items = ["apple", "banana", "cherry"]
try:
    print(items[5])
except IndexError:
    print("Index out of range")
```

A) `"cherry"`
B) `None`
C) `IndexError`
D) `"Index out of range"`

Answer: D) `"Index out of range"` (The `IndexError` is caught by the `except` block)

MCQs on AttributeError

---

187. What is an `AttributeError` in Python?

A) An error caused by an undefined variable
B) An error that occurs when an invalid attribute is accessed on an object
C) An error due to incorrect syntax in Python
D) An error raised when a list index is out of range

Answer: B) An error that occurs when an invalid attribute is accessed on an object

---

188. Which of the following will raise an `AttributeError`?

A) `x = 10; print(x.real)`
B) `"hello".upper()`
C) `numbers = [1, 2, 3]; numbers.append(4)`
D) `y = 3.14; y.append(2)`

Answer: D) `y = 3.14; y.append(2)` (`float` objects do not have an `append()` method)

---

189. What will be the output of the following code?

```
num = 42
print(num.lower())
```

A) `"42"`
B) `"42".lower()`
C) `AttributeError`
D) `None`

Answer: C) `AttributeError` (`int` objects do not have a `lower()` method)

---

190. Which of the following will NOT raise an `AttributeError`?

A) `"Hello".capitalize()`
B) `data = [1, 2, 3]; data.sort()`
C) `my_tuple = (5, 10, 15); my_tuple.append(20)`
D) `val = 3.14; val.upper()`

Answer: A) `"Hello".capitalize()` (Valid string method)

---

191. What happens when you access an attribute that does not exist for an object?

A) The program runs without errors
B) The program raises an `AttributeError`
C) The program prints `None`
D) The program automatically creates the attribute

Answer: B) The program raises an `AttributeError`

---

192. What will be the output of the following code?

```
class Car:
    def __init__(self, brand):
        self.brand = brand

c = Car("Toyota")
print(c.color)
```

A) `"Toyota"`
B) `None`
C) `AttributeError`
D) `""`

Answer: C) `AttributeError` (`color` attribute is not defined in the `Car` class)

---

193. How can you avoid an `AttributeError` when accessing object attributes?

A) Always use integer values for attributes
B) Check if an attribute exists using `hasattr(object, "attribute")` before accessing it
C) Use only built-in data types
D) Convert attributes to strings before accessing them

Answer: B) Check if an attribute exists using `hasattr(object, "attribute")` before accessing it

---

194. What will be the output of the following code?

```
text = "Python Programming"
print(text.append(" is fun"))
```

A) `"Python Programming is fun"`
B) `None`
C) `AttributeError`
D) `"Python Programmingappend is fun"`

Answer: C) `AttributeError` (Strings do not have an `append()` method)

---

195. What is the cause of the `AttributeError` in the following code?

```
x = None
print(x.upper())
```

A) `NoneType` object has no method `upper()`
B) `None` is a valid string
C) `upper()` is not a valid method in Python
D) The code will run without errors

Answer: A) `NoneType` object has no method `upper()`

---

196. How can you handle an `AttributeError` in Python?

A) By using a `try-except` block
B) By using a `while` loop
C) By converting the object into an integer
D) By using a `for` loop

Answer: A) By using a `try-except` block

MCQs on ImportError

197. What is an `ImportError` in Python?

A) An error that occurs when a module is not found or cannot be imported properly
B) An error due to incorrect syntax in Python
C) An error that occurs when accessing an invalid attribute of an object
D) An error caused by an infinite loop

Answer: A) An error that occurs when a module is not found or cannot be imported properly

---

198. Which of the following will raise an `ImportError`?

A) `import os`
B) `import math`
C) `import non_existent_module`
D) `from datetime import datetime`

Answer: C) `import non_existent_module` (This module does not exist)

---

199. What will be the output of the following code?

```
import mymodule
```

(Assume `mymodule` is not installed or not in the script's directory.)

A) `None`
B) `ImportError`
C) `"Module imported successfully"`
D) `SyntaxError`

Answer: B) `ImportError` (Python cannot find `mymodule`)

---

200. Which of the following is NOT a reason for an `ImportError`?

A) The module does not exist
B) The module name is misspelled
C) The module is not installed
D) The module contains syntax errors

Answer: D) The module contains syntax errors (This would raise a `SyntaxError`, not an `ImportError`)

201. What is the difference between `ImportError` and `ModuleNotFoundError`?

A) `ImportError` occurs when a module is found but cannot be imported, while `ModuleNotFoundError` occurs when a module is missing
B) They are the same error in Python
C) `ModuleNotFoundError` occurs only in Python 2
D) `ImportError` is raised only when importing built-in modules

Answer: A) `ImportError` occurs when a module is found but cannot be imported, while `ModuleNotFoundError` occurs when a module is missing

---

202. What will be the output of the following code?

```
try:
    import randommodule
except ImportError:
    print("Module not found!")
```

A) `"Module not found!"`
B) `ImportError`
C) `"randommodule imported successfully"`
D) `SyntaxError`

Answer: A) `"Module not found!"` (The `ImportError` is caught by the `except` block)

---

203. How can you avoid an `ImportError` in Python?

A) Ensure the module is installed
B) Check for typos in the module name
C) Verify that the module is in the correct directory
D) All of the above

Answer: D) All of the above

---

204. What will be the output of the following code?

```
from math import square
```

A) No output
B) `ImportError`
C) `ModuleNotFoundError`
D) `None`

Answer: B) `ImportError` (`math` module exists, but `square` function does not exist in it)

---

205. Which command should you use to install a missing module and avoid `ImportError`?

A) `install module_name`
B) `python install module_name`
C) `pip install module_name`
D) `import module_name`

Answer: C) `pip install module_name`

---

206. What will be the output of the following code if `pandas` is not installed?

```
import pandas
```

A) `ModuleNotFoundError`
B) `SyntaxError`
C) `None`
D) `IndexError`

Answer: A) `ModuleNotFoundError` (Because `pandas` is not installed)

(MCQs) on "Memory Errors"

---

201. What is a common cause of memory leaks in Python?

a) Improper use of global variables
b) Forgetting to close files
c) Circular references in objects
d) Using the `del` statement

Answer: c) Circular references in objects

---

202. How can memory leaks be detected in a Python program?

a) Using `time.sleep()`
b) Using `gc.collect()`
c) Using a memory profiler like `memory_profiler`
d) Increasing system RAM

Answer: c) Using a memory profiler like `memory_profiler`

---

203. What happens when Python runs out of memory?

a) The program continues running but slows down
b) Python automatically allocates more memory
c) The program raises a `MemoryError`
d) The program starts using virtual memory

Answer: c) The program raises a `MemoryError`

---

204. Which Python module helps in automatic garbage collection?

a) `os`
b) `sys`
c) `gc`
d) `re`

Answer: c) `gc`

---

205. What is a circular reference in Python?

a) When a variable refers to itself
b) When two or more objects reference each other, preventing garbage collection
c) When a function calls itself recursively
d) When memory is freed before use

Answer: b) When two or more objects reference each other, preventing garbage collection

---

206. How can you manually trigger garbage collection in Python?

a) `sys.collect()`
b) `gc.collect()`
c) `os.cleanup()`
d) `memory.flush()`

Answer: b) `gc.collect()`

---

207. Which of the following can help reduce memory consumption in Python?

a) Using generators instead of lists
b) Using deep copies instead of shallow copies
c) Using a large number of global variables
d) Using infinite loops

Answer: a) Using generators instead of lists

---

208. What is a common issue when handling large data structures like lists and dictionaries in Python?

a) Python automatically deletes them after use
b) They may consume excessive memory if not managed properly
c) They cause syntax errors
d) They prevent garbage collection

Answer: b) They may consume excessive memory if not managed properly

---

209. What does the `del` statement do in Python?

a) It permanently removes an object from memory
b) It sets an object reference to `None`, but may not free memory immediately
c) It forces garbage collection immediately
d) It prevents memory leaks

Answer: b) It sets an object reference to `None`, but may not free memory immediately

---

210. What is the best way to handle large files in Python without consuming too much memory?

a) Load the entire file into a list
b) Read the file line by line using a generator (`yield`)
c) Convert the file into a dictionary before processing
d) Use deep copies of the file contents

Answer: b) Read the file line by line using a generator (`yield`)

(MCQs) on "Overflow Errors"

---

211. What is an OverflowError in Python?

a) When a variable is assigned an incorrect data type
b) When a calculation exceeds the maximum limit of a numeric type
c) When a function is called recursively too many times
d) When Python runs out of memory

Answer: b) When a calculation exceeds the maximum limit of a numeric type

---

212. In which situation is an `OverflowError` most likely to occur in Python?

a) When dividing by zero
b) When performing operations with extremely large floating-point numbers
c) When indexing a list beyond its range
d) When converting a string to an integer

Answer: b) When performing operations with extremely large floating-point numbers

---

213. How can you avoid an OverflowError in Python?

a) Use the `int` data type instead of `float`
b) Use `try-except` to catch and handle the error
c) Convert large numbers to strings
d) Restart the Python interpreter

Answer: b) Use `try-except` to catch and handle the error

---

214. Which Python data type is least likely to cause an OverflowError?

a) `float`
b) `int`
c) `complex`
d) `bool`

Answer: b) `int` (because Python integers have arbitrary precision)

---

215. What happens when an integer calculation exceeds the maximum value in Python?

a) Python raises an `OverflowError`
b) The integer wraps around to the minimum value
c) Python automatically converts it to a long integer
d) Python allows unlimited integer size without error

Answer: d) Python allows unlimited integer size without error

---

216. Which Python module can help handle large numbers without overflow issues?

a) `math`
b) `decimal`
c) `os`
d) `sys`

Answer: b) `decimal`

---

217. What will be the output of the following code?

```
import math
print(math.exp(1000))
```

a) A very large number
b) `OverflowError: math range error`
c) `inf`
d) `None`

Answer: b) `OverflowError: math range error`

---

218. How can you prevent an `OverflowError` when computing large exponents?

a) Use the `decimal` module
b) Use `math.log` instead of direct exponentiation
c) Use `try-except` to handle errors
d) All of the above

Answer: d) All of the above

---

219. What is the maximum size of an integer in Python?

a) 2^{31} - 1
b) 2^{63} - 1
c) 10^{18}
d) No fixed limit (limited by available memory)

Answer: d) No fixed limit (limited by available memory)

---

220. What happens when a floating-point operation exceeds the limit in Python?

a) It results in `inf` (infinity)
b) It raises an `OverflowError`
c) The program crashes
d) The number is converted to an integer

Answer: a) It results in `inf` (infinity)

---

221. What will be the output of the following code?

```
import sys
print(sys.float_info.max * 2)
```

a) `inf`
b) A very large number
c) `OverflowError`
d) `0`

Answer: a) `inf`

---

222. Which function helps control the precision of floating-point calculations to avoid overflow?

a) `round()`
b) `decimal.Decimal()`
c) `int()`
d) `float()`

Answer: b) `decimal.Decimal()`

---

223. What is the best way to handle an OverflowError when working with large numbers?

a) Reduce the number of decimal places
b) Convert the numbers to integers
c) Use exception handling (`try-except`)
d) Store the numbers as strings

Answer: c) Use exception handling (`try-except`)

---

224. Which of the following expressions may cause an OverflowError?

a) `math.factorial(1000)`
b) `2 ** 1000`
c) `math.exp(1000)`
d) `10 / 3`

Answer: c) `math.exp(1000)`

---

225. What will be the output of the following code?

```
import math
try:
    print(math.exp(2000))
except OverflowError:
    print("Overflow occurred")
```

a) A very large number
b) `Overflow occurred`
c) `None`
d) `inf`

Answer: b) `Overflow occurred`

(MCQs) on "Key Errors"

226. What is a KeyError in Python?

a) An error that occurs when a key is missing in a dictionary
b) An error caused by an incorrect file path
c) An error that occurs when a variable is not defined
d) An error that happens when a list index is out of range

Answer: a) An error that occurs when a key is missing in a dictionary

---

227. How can you avoid a KeyError when accessing a dictionary?

a) Use `dict.get(key)` instead of `dict[key]`
b) Use a `try-except` block
c) Check if the key exists using `if key in dict`
d) All of the above

Answer: d) All of the above

---

228. What happens if you try to access a non-existent key using `my_dict[key]`?

a) Returns `None`
b) Raises a `KeyError`
c) Returns an empty dictionary
d) Creates a new key-value pair

Answer: b) Raises a `KeyError`

---

229. How does the `get()` method help in preventing KeyErrors?

a) It returns `None` if the key is not found instead of raising an error
b) It automatically creates the key in the dictionary
c) It raises an error when a key is not found
d) It deletes the key from the dictionary

Answer: a) It returns `None` if the key is not found instead of raising an error

---

230. What will be the output of the following code?

```
my_dict = {'a': 1, 'b': 2}
print(my_dict.get('c'))
```

a) `KeyError`
b) `None`
c) `0`
d) `'c'`

Answer: b) `None`

---

231. What will be the output of the following code?

```
my_dict = {'x': 100, 'y': 200}
print(my_dict.get('z', 'Not Found'))
```

a) `KeyError`
b) `'Not Found'`

c) `None`
d) `0`

Answer: b) `'Not Found'`

---

232. Which method can be used to check if a key exists in a dictionary before accessing it?

a) `dict.exists(key)`
b) `if key in dict:`
c) `dict.contains(key)`
d) `dict.has_key(key)`

Answer: b) `if key in dict:`

---

233. What is the best way to handle a KeyError in a dictionary lookup?

a) Use `try-except` to catch the `KeyError`
b) Use the `get()` method
c) Check key existence with `in` operator
d) All of the above

Answer: d) All of the above

---

234. What will be the output of this code?

```
my_dict = {'name': 'Alice', 'age': 25}
try:
    print(my_dict['gender'])
except KeyError:
    print("Key not found")
```

a) `KeyError`
b) `'gender'`
c) `'Key not found'`
d) `None`

Answer: c) `'Key not found'`

---

235. How can you safely remove a key from a dictionary without causing a KeyError?

a) Use `del my_dict[key]`
b) Use `my_dict.pop(key, None)`
c) Use `my_dict.remove(key)`
d) Use `my_dict.clear(key)`

Answer: b) Use `my_dict.pop(key, None)`

---

236. What does the `setdefault()` method do in dictionaries?

a) Raises a `KeyError` if the key is missing
b) Returns the key's value if found, otherwise sets and returns a default value
c) Deletes the key from the dictionary
d) Converts the dictionary to a set

Answer: b) Returns the key's value if found, otherwise sets and returns a default value

---

237. What will be the output of this code?

```
my_dict = {'x': 5}
print(my_dict.setdefault('y', 10))
print(my_dict)
```

a) `KeyError`
b) `None`
c) `10` and `{'x': 5, 'y': 10}`
d) `5` and `{'x': 5, 'y': 10}`

Answer: c) `10` and `{'x': 5, 'y': 10}`

---

238. What happens if you try to update a dictionary using `update()` with a key that does not exist?

a) Raises a `KeyError`
b) Adds the new key-value pair to the dictionary
c) Ignores the update
d) Returns `None`

Answer: b) Adds the new key-value pair to the dictionary

---

239. How can you iterate over dictionary keys safely without causing a KeyError?

a) Use `for key in dict.keys()`
b) Use `try-except` for missing keys
c) Use `.get()` instead of `[]`
d) All of the above

Answer: d) All of the above

---

240. What will be the output of this code?

```
my_dict = {'a': 1, 'b': 2}
print(my_dict.get('c', 3))
```

a) `KeyError`
b) `None`
c) `3`
d) `'c'`

Answer: c) `3`

Basic Algorithm MCQs

241. What is an algorithm?

A) A programming language
B) A step-by-step procedure to solve a problem
C) A type of computer
D) A mathematical equation

Answer: B) A step-by-step procedure to solve a problem

---

242. What are the key characteristics of an algorithm?

A) It must be ambiguous
B) It must have a clear stopping point

C) It should be infinitely long
D) It must be slow

Answer: B) It must have a clear stopping point

---

243. Which of the following is NOT a property of an algorithm?

A) Finiteness
B) Ambiguity
C) Input and Output
D) Effectiveness

Answer: B) Ambiguity

---

244. What is the purpose of an algorithm?

A) To increase errors in a program
B) To provide a systematic way to solve a problem
C) To make programming harder
D) To slow down execution

Answer: B) To provide a systematic way to solve a problem

---

245. Which of the following best describes the time complexity of an algorithm?

A) The amount of time an algorithm takes to execute
B) The number of errors in an algorithm
C) The memory required by an algorithm
D) The programming language used

Answer: A) The amount of time an algorithm takes to execute

---

246. What is the best case time complexity of the Linear Search algorithm?

A) O(n)
B) O(1)
C) $O(n^2)$
D) O(log n)

Answer: B) O(1)

---

247. Which of the following sorting algorithms has the worst-case time complexity of $O(n^2)$?

A) Merge Sort
B) QuickSort
C) Bubble Sort
D) Heap Sort

Answer: C) Bubble Sort

---

248. What is the purpose of a flowchart in algorithm design?

A) To write code
B) To create hardware
C) To visually represent the steps of an algorithm
D) To slow down execution

Answer: C) To visually represent the steps of an algorithm

---

249. Which data structure is used for implementing recursion in an algorithm?

A) Queue
B) Stack
C) Linked List
D) Array

Answer: B) Stack

---

250. What is the primary goal of analyzing an algorithm?

A) To check if the algorithm is fun
B) To determine efficiency in terms of time and space
C) To compare different programming languages
D) To count the number of lines in the code

Answer: B) To determine efficiency in terms of time and space

---

251. What does Big O notation represent?

A) The name of an algorithm
B) The worst-case time complexity of an algorithm
C) The best-case time complexity of an algorithm
D) The number of inputs in an algorithm

Answer: B) The worst-case time complexity of an algorithm

---

252. What is the first step in solving a problem using an algorithm?

A) Writing the code
B) Understanding the problem
C) Choosing a random solution
D) Ignoring the problem

Answer: B) Understanding the problem

---

253. Which searching algorithm works best with sorted data?

A) Linear Search
B) Binary Search
C) Bubble Sort
D) Depth-First Search

Answer: B) Binary Search

---

254. Which sorting algorithm follows the divide and conquer technique?

A) Bubble Sort
B) Merge Sort
C) Selection Sort
D) Insertion Sort

Answer: B) Merge Sort

---

255. What does "space complexity" of an algorithm refer to?

A) The time taken by an algorithm
B) The amount of memory required by an algorithm
C) The number of errors in an algorithm
D) The programming language used

Answer: B) The amount of memory required by an algorithm

---

256. Which of the following is a searching algorithm?

A) Bubble Sort
B) Merge Sort
C) Binary Search
D) QuickSort

Answer: C) Binary Search

---

257. Which algorithm is used to find the shortest path in a graph?

A) Merge Sort
B) Dijkstra's Algorithm
C) Bubble Sort
D) QuickSort

Answer: B) Dijkstra's Algorithm

---

258. Which of the following is NOT an algorithm design technique?

A) Divide and Conquer
B) Dynamic Programming
C) Greedy Approach
D) Debugging

Answer: D) Debugging

---

259. What is the worst-case time complexity of QuickSort?

A) O(n log n)
B) $O(n^2)$
C) O(n)
D) O(log n)

Answer: B) $O(n^2)$

260. Which of the following problems is solved using recursion?

A) Binary Search
B) Fibonacci Sequence
C) Tower of Hanoi
D) All of the above

Answer: D) All of the above

Basic Flowchart MCQs

261. What is a flowchart?

A) A type of programming language
B) A graphical representation of an algorithm
C) A hardware diagram
D) A type of mathematical formula

Answer: B) A graphical representation of an algorithm

262. Which symbol is used to represent the start and end of a flowchart?

A) Rectangle
B) Oval (Ellipse)
C) Diamond
D) Parallelogram

Answer: B) Oval (Ellipse)

263. What is the purpose of a flowchart?

A) To make coding more complex
B) To represent an algorithm visually
C) To replace coding
D) To confuse programmers

Answer: B) To represent an algorithm visually

---

264. Which flowchart symbol is used for decision-making?

A) Rectangle
B) Oval
C) Diamond
D) Parallelogram

Answer: C) Diamond

---

265. Which symbol is used to represent input/output in a flowchart?

A) Oval
B) Rectangle
C) Parallelogram
D) Diamond

Answer: C) Parallelogram

---

266. Which symbol is used to represent a process or operation in a flowchart?

A) Rectangle
B) Parallelogram
C) Oval
D) Diamond

Answer: A) Rectangle

---

267. What is the direction of flow in a flowchart?

A) Random
B) Left to Right or Top to Bottom
C) Right to Left only
D) Bottom to Top

Answer: B) Left to Right or Top to Bottom

---

268. What does an arrow represent in a flowchart?

A) A decision
B) A connector showing the flow of steps
C) An input
D) A process

Answer: B) A connector showing the flow of steps

---

269. Which of the following is an advantage of using a flowchart?

A) Increases complexity
B) Makes the logic of a program easy to understand
C) Eliminates the need for coding
D) Slows down execution

Answer: B) Makes the logic of a program easy to understand

---

270. Which symbol is used to connect different parts of a flowchart?

A) Circle
B) Rectangle
C) Diamond
D) Parallelogram

Answer: A) Circle

---

271. In a flowchart, what does the parallelogram symbol represent?

A) Process
B) Decision
C) Input/Output
D) Termination

Answer: C) Input/Output

---

272. Which of the following is NOT a standard flowchart symbol?

A) Oval
B) Rectangle
C) Star
D) Diamond

Answer: C) Star

---

273. Which of the following is a limitation of using flowcharts?

A) They are difficult to interpret
B) They are time-consuming to draw for complex problems
C) They cannot be used in programming
D) They have no use in problem-solving

Answer: B) They are time-consuming to draw for complex problems

---

274. What is the first step in creating a flowchart?

A) Writing the code
B) Understanding the problem
C) Drawing the final flowchart
D) Implementing the solution

Answer: B) Understanding the problem

---

275. What should be done if a flowchart becomes too complex?

A) Stop using flowcharts
B) Use sub-processes or modular flowcharts
C) Ignore the complexity
D) Remove decision-making steps

Answer: B) Use sub-processes or modular flowcharts

---

276. Which type of flowchart represents the entire system at a higher level?

A) Detailed flowchart
B) System flowchart
C) Program flowchart
D) Data flowchart

Answer: B) System flowchart

---

277. What type of flowchart is used for representing a computer program?

A) System flowchart
B) Program flowchart
C) Data flowchart
D) Hardware flowchart

Answer: B) Program flowchart

---

278. In a flowchart, how is a loop represented?

A) Using a decision symbol (Diamond) with arrows looping back
B) Using a rectangle only
C) Using an oval
D) Loops cannot be represented in flowcharts

Answer: A) Using a decision symbol (Diamond) with arrows looping back

---

279. Which of the following tools can be used to create flowcharts?

A) Microsoft Word
B) Microsoft PowerPoint
C) Online flowchart tools like Lucidchart or Draw.io
D) All of the above

Answer: D) All of the above

---

280. What should a well-structured flowchart avoid?

A) Clear logic
B) Proper sequence of steps
C) Too many crossing lines and confusion
D) Proper use of symbols

Answer: C) Too many crossing lines and confusion

Basic Decision Table MCQs

281. What is a decision table?

A) A type of programming language
B) A graphical representation of an algorithm
C) A table used to represent complex decision logic
D) A type of hardware component

Answer: C) A table used to represent complex decision logic

---

282. What are the main components of a decision table?

A) Rows and Columns
B) Conditions and Actions
C) Inputs and Outputs
D) Loops and Functions

Answer: B) Conditions and Actions

---

283. What is the purpose of a decision table?

A) To make programming more difficult
B) To visually represent complex decision-making logic
C) To replace flowcharts
D) To slow down execution

Answer: B) To visually represent complex decision-making logic

---

284. Which of the following is NOT a part of a decision table?

A) Condition Stubs
B) Action Stubs
C) Decision Tree
D) Rules

Answer: C) Decision Tree

---

285. What do the "Condition Stubs" in a decision table represent?

A) The possible conditions that affect a decision
B) The final result of the decision table
C) The actions to be taken
D) The number of rules in the table

Answer: A) The possible conditions that affect a decision

---

286. What do the "Action Stubs" in a decision table represent?

A) The possible conditions
B) The actions taken when conditions are met
C) The number of columns in the table
D) The conditions that do not change

Answer: B) The actions taken when conditions are met

---

287. What do the "Rules" in a decision table represent?

A) The list of programming languages
B) The conditions and corresponding actions
C) The number of columns in the table
D) The step-by-step process of an algorithm

Answer: B) The conditions and corresponding actions

---

288. Which of the following is an advantage of decision tables?

A) They make complex decision-making simple and clear
B) They increase coding errors
C) They are only useful for small problems
D) They slow down decision-making

Answer: A) They make complex decision-making simple and clear

---

289. What is the use of the "Don't Care" condition in a decision table?

A) To make the table more complex
B) To indicate that the condition does not affect the decision
C) To reduce the number of rules
D) To increase the number of conditions

Answer: B) To indicate that the condition does not affect the decision

---

290. How can a decision table be converted into a program?

A) By using decision rules as conditions in an if-else statement
B) By using a flowchart only
C) By writing the program without considering the table
D) A decision table cannot be converted into a program

Answer: A) By using decision rules as conditions in an if-else statement

---

291. What is a Limited Entry Decision Table?

A) A table that allows only a limited number of rules
B) A table where each condition has only two possible values (Yes/No or True/False)
C) A table with unlimited conditions
D) A table used only for mathematical calculations

Answer: B) A table where each condition has only two possible values (Yes/No or True/False)

---

292. How many parts does a decision table typically have?

A) 2
B) 3
C) 4
D) 5

Answer: C) 4 (Condition Stubs, Condition Entries, Action Stubs, Action Entries)

---

293. What is an Extended Entry Decision Table?

A) A decision table with multiple condition values instead of just Yes/No
B) A table with a limited number of conditions
C) A table used only for small problems
D) A table with only one condition

Answer: A) A decision table with multiple condition values instead of just Yes/No

---

294. Which of the following is NOT an advantage of using a decision table?

A) Helps in identifying missing cases
B) Simplifies complex decision logic
C) Requires less memory than flowcharts
D) Can be automated for business rules

Answer: C) Requires less memory than flowcharts

---

295. Which software development phase commonly uses decision tables?

A) Coding
B) Testing
C) Requirement Analysis
D) Debugging

Answer: C) Requirement Analysis

---

296. What is the main difference between a Decision Table and a Decision Tree?

A) A Decision Table is graphical, while a Decision Tree is tabular
B) A Decision Table is tabular, while a Decision Tree is graphical
C) Both are the same
D) Decision Trees do not involve conditions

Answer: B) A Decision Table is tabular, while a Decision Tree is graphical

---

297. Which of the following statements is true about decision tables?

A) Decision tables help in identifying missing test cases
B) Decision tables cannot be used in real-life scenarios
C) Decision tables are only useful for small programs
D) Decision tables are never used in business applications

Answer: A) Decision tables help in identifying missing test cases

---

298. When should you use a decision table?

A) When conditions and outcomes are simple
B) When there are multiple conditions leading to different actions
C) When no decisions need to be made
D) When programming without logic

Answer: B) When there are multiple conditions leading to different actions

---

299. What is an Action Entry in a decision table?

A) A description of possible actions
B) The execution result of an algorithm
C) The entry that lists all conditions
D) The part of the table that determines the number of rules

Answer: A) A description of possible actions

300. Which field commonly uses decision tables?

A) Agriculture
B) Business Rules and Software Testing
C) Cooking
D) Music Production

Answer: B) Business Rules and Software Testing

MCQs on Structured Programming Concepts

301. What is structured programming?

A) A method of writing programs using only loops
B) A programming paradigm that emphasizes clear, readable, and organized code
C) A way to write machine code directly
D) A type of hardware programming

Answer: B) A programming paradigm that emphasizes clear, readable, and organized code

302. Which of the following is NOT a characteristic of structured programming?

A) Use of functions and procedures
B) Breaking a program into modules
C) Extensive use of GOTO statements
D) Use of loops and conditionals

Answer: C) Extensive use of GOTO statements

303. Which of the following control structures are fundamental in structured programming?

A) Sequence, Selection, Iteration
B) Input, Output, Storage
C) Compilation, Execution, Debugging
D) Arrays, Functions, Pointers

Answer: A) Sequence, Selection, Iteration

---

304. What does the "Sequence" control structure in structured programming refer to?

A) Executing instructions in a specific order
B) Making decisions using IF statements
C) Repeating a block of code multiple times
D) Skipping parts of a program

Answer: A) Executing instructions in a specific order

---

305. Which of the following is an example of the "Selection" control structure?

A) for loop
B) while loop
C) if-else statement
D) switch statement

Answer: C) if-else statement

---

306. Which control structure is used to repeat a block of code multiple times?

A) Sequence
B) Iteration
C) Selection
D) Compilation

Answer: B) Iteration

---

307. Which of the following loops is NOT available in structured programming?

A) for loop
B) while loop
C) do-while loop
D) GOTO loop

Answer: D) GOTO loop

---

308. What is the main advantage of structured programming?

A) Programs are easier to understand and modify
B) Programs run faster than unstructured programs
C) It eliminates the need for debugging
D) It does not require functions or procedures

Answer: A) Programs are easier to understand and modify

---

309. Which programming language is most closely associated with structured programming?

A) Assembly Language
B) C
C) HTML
D) Machine Language

Answer: B) C

---

310. What is modular programming in structured programming?

A) Writing a program as one single function
B) Breaking a program into small independent functions or modules
C) Using only loops to control program execution
D) Avoiding all types of conditional statements

Answer: B) Breaking a program into small independent functions or modules

---

311. Which of the following statements about structured programming is true?

A) It allows the use of the GOTO statement extensively
B) It is difficult to debug structured programs
C) It improves readability and maintainability
D) It cannot be used for large programs

Answer: C) It improves readability and maintainability

---

312. Which of the following is NOT a structured programming language?

A) Pascal
B) C
C) Python
D) Assembly

Answer: D) Assembly

---

313. What is the main problem with using the GOTO statement in a program?

A) It makes the program faster
B) It makes the program difficult to understand and maintain
C) It improves program readability
D) It increases memory usage

Answer: B) It makes the program difficult to understand and maintain

---

314. Which programming construct allows code execution based on a condition?

A) Sequence
B) Selection
C) Iteration
D) Compilation

Answer: B) Selection

---

315. What is the purpose of the "break" statement in structured programming?

A) To stop the execution of a loop or switch statement
B) To define a function
C) To jump to another part of the program
D) To create a new variable

Answer: A) To stop the execution of a loop or switch statement

---

316. In structured programming, what is the role of a function?

A) To store data permanently
B) To allow reusability and modularization of code
C) To replace loops
D) To make a program execute faster

Answer: B) To allow reusability and modularization of code

---

317. What is recursion in structured programming?

A) A loop that runs indefinitely
B) A function that calls itself
C) A way to define global variables
D) A method to stop program execution

Answer: B) A function that calls itself

---

318. Which of the following is NOT a benefit of structured programming?

A) Code is easier to understand
B) Code is more reusable
C) Code runs faster than machine language
D) Code is easier to debug and modify

Answer: C) Code runs faster than machine language

---

319. What is the purpose of the "continue" statement in structured programming?

A) To terminate the program
B) To exit the loop immediately
C) To skip the current iteration of a loop and continue with the next iteration
D) To move execution to a different function

Answer: C) To skip the current iteration of a loop and continue with the next iteration

320. Which principle of structured programming helps in reducing code duplication?

A) Using GOTO statements
B) Writing long, continuous blocks of code
C) Using functions and modular programming
D) Avoiding loops and conditional statements

Answer: C) Using functions and modular programming

MCQs on Top-down and Bottom-up Programming

321. What is a programming methodology?

A) A set of rules for writing machine code
B) A structured approach to software development
C) A method used only in AI programming
D) A way to design hardware

Answer: B) A structured approach to software development

322. What is the main principle of the top-down approach?

A) Breaking a complex problem into smaller subproblems
B) Combining small modules into a larger system
C) Writing code without any design phase
D) Using only object-oriented programming

Answer: A) Breaking a complex problem into smaller subproblems

323. What is the first step in the top-down approach?

A) Identifying the smallest functions
B) Designing the overall system before coding
C) Writing all functions first
D) Implementing detailed algorithms before structure

Answer: B) Designing the overall system before coding

324. Which of the following is a key characteristic of the bottom-up approach?

A) Starting with the high-level design and breaking it into submodules
B) Combining small, well-defined modules to form a complete system
C) Writing code without defining a structure
D) Using only procedural programming

Answer: B) Combining small, well-defined modules to form a complete system

325. In the top-down approach, how are problems typically solved?

A) By starting with the simplest components and integrating them
B) By focusing on coding first and designing later
C) By breaking down the main problem into subproblems and solving them step by step
D) By writing the entire program as a single function

Answer: C) By breaking down the main problem into subproblems and solving them step by step

326. What is the main focus of the bottom-up approach?

A) Understanding the entire system first
B) Developing and testing small, reusable components before integrating them
C) Avoiding modularization
D) Writing the entire program in a single flow

Answer: B) Developing and testing small, reusable components before integrating them

327. Which of the following is an advantage of the top-down approach?

A) It requires minimal initial planning
B) It helps in better understanding of system architecture
C) It starts with coding and testing simultaneously
D) It focuses only on low-level details

Answer: B) It helps in better understanding of system architecture

---

328. Which programming paradigm is most closely related to the bottom-up approach?

A) Structured programming
B) Object-oriented programming
C) Functional programming
D) Assembly programming

Answer: B) Object-oriented programming

---

329. Which methodology is generally used in procedural programming languages like C?

A) Bottom-up approach
B) Top-down approach
C) Neural network approach
D) Data-driven approach

Answer: B) Top-down approach

---

330. In a bottom-up approach, how is the system built?

A) By defining high-level modules first
B) By writing functions first and integrating them later
C) By avoiding modularization and coding everything together
D) By skipping testing and debugging

Answer: B) By writing functions first and integrating them later

---

331. Which of the following is an advantage of the bottom-up approach?

A) It provides better system-level understanding from the beginning
B) It encourages reusability of components
C) It focuses only on the user interface
D) It requires no initial planning

Answer: B) It encourages reusability of components

---

332. Which of the following is a disadvantage of the top-down approach?

A) It makes debugging easier
B) It requires a clear understanding of the overall system from the start
C) It starts with small components
D) It encourages object-oriented design

Answer: B) It requires a clear understanding of the overall system from the start

---

333. Which of the following best describes the relationship between top-down and bottom-up approaches?

A) They are completely independent and unrelated
B) They are opposing methodologies but can be used together in software development
C) Only the top-down approach is effective
D) The bottom-up approach is only used in assembly language

Answer: B) They are opposing methodologies but can be used together in software development

---

334. What is a key benefit of using the top-down approach in large projects?

A) It allows teams to focus on smaller, manageable parts before implementation
B) It eliminates the need for documentation
C) It avoids planning and allows developers to code immediately
D) It is only useful for small-scale projects

Answer: A) It allows teams to focus on smaller, manageable parts before implementation

---

335. What is the main drawback of the bottom-up approach?

A) It does not allow reusability
B) It does not provide a clear high-level view of the system in the beginning
C) It cannot be used in object-oriented programming
D) It requires the use of GOTO statements

Answer: B) It does not provide a clear high-level view of the system in the beginning

---

336. In which approach do developers first build and test low-level modules before combining them into higher-level structures?

A) Top-down
B) Bottom-up
C) Waterfall
D) Agile

Answer: B) Bottom-up

---

337. Which of the following best represents the top-down approach in problem-solving?

A) Designing the entire architecture first and then implementing details
B) Writing random pieces of code and integrating them later
C) Avoiding modular design
D) Only focusing on small functions without considering the entire system

Answer: A) Designing the entire architecture first and then implementing details

---

338. Which software development model is more aligned with the top-down approach?

A) Agile Model
B) Waterfall Model
C) Spiral Model
D) Prototyping Model

Answer: B) Waterfall Model

---

339. How does the bottom-up approach improve code reuse?

A) By forcing developers to rewrite every function from scratch
B) By designing small modules that can be reused in different parts of the system
C) By avoiding modularity and making all code dependent on a single function
D) By focusing only on hardware implementation

Answer: B) By designing small modules that can be reused in different parts of the system

---

340. Which of the following is a real-world example of a bottom-up approach?

A) Designing a car by first sketching the entire structure and then creating the parts
B) Building a house by constructing each room separately and then assembling them
C) Writing a book by first drafting the entire story and then refining the details
D) Developing an application by designing the user interface first

Answer: B) Building a house by constructing each room separately and then assembling them

1. What is the correct extension for Python files?

- a) .py
- b) .pt
- c) .pyt
- d) .pyth

  ✅ **Answer:** a) .py

---

2. Who developed Python?

- a) Dennis Ritchie
- b) Guido van Rossum
- c) James Gosling
- d) Bjarne Stroustrup

  ✅ **Answer:** b) Guido van Rossum

---

3. Which of the following is the correct way to print in Python?

- a) echo("Hello")
- b) print("Hello")
- c) print Hello
- d) printf("Hello")

  ✅ **Answer:** b) print("Hello")

---

4. What is the output of `print(2**3)`?

- a) 6
- b) 9
- c) 8
- d) 5

  ✅ **Answer:** c) 8

---

5. Which of the following is used to take input in Python?

- a) scanf()
- b) cin
- c) input()

- d) read()
  ✅ **Answer:** c) input()

6. What is the correct syntax to declare a list in Python?

- a) list = {1, 2, 3}
- b) list = [1, 2, 3]
- c) list = (1, 2, 3)
- d) list = <1, 2, 3>
  ✅ **Answer:** b) list = [1, 2, 3]

7. Which data type is immutable in Python?

- a) List
- b) Set
- c) Dictionary
- d) Tuple
  ✅ **Answer:** d) Tuple

8. How do you comment a single line in Python?

- a) //
- b) <!-- -->
- c) #
- d) /* */
  ✅ **Answer:** c) #

9. What is the output of `type(5)`?

- a) int
- b) float
- c) str
- d) bool
  ✅ **Answer:** a) int

10. What is the output of `10 % 3`?

- a) 3
- b) 1
- c) 0
- d) 10
  ✅ **Answer:** b) 1

---

11. Which method is used to remove an element from a list?

- a) delete()
- b) pop()
- c) remove()
- d) discard()
  ✅ **Answer:** c) remove()

---

12. How do you create an infinite loop in Python?

- a) while True:
- b) for i in range(0, ∞):
- c) while(1):
- d) Both a and c
  ✅ **Answer:** d) Both a and c

---

13. What is the output of `bool(0)`?

- a) True
- b) False
- c) 0
- d) 1
  ✅ **Answer:** b) False

---

14. Which keyword is used to define a function in Python?

- a) func
- b) define
- c) def

- d) function
  ✅ **Answer:** c) def

---

15. What is the result of `len("Python")`?

- a) 5
- b) 6
- c) 7
- d) Error
  ✅ **Answer:** b) 6

---

16. Which function is used to convert a string into an integer?

- a) str()
- b) float()
- c) int()
- d) bool()
  ✅ **Answer:** c) int()

---

17. Which symbol is used for comments in Python?

- a) //
- b) #
- c) <!-- -->
- d) %
  ✅ **Answer:** b) #

---

18. How do you create an empty dictionary in Python?

- a) dict = []
- b) dict = ()
- c) dict = {}
- d) dict = set()
  ✅ **Answer:** c) dict = {}

---

19. What is the result of `2 + 3 * 4`?

- a) 20
- b) 14
- c) 24
- d) 12
  ✅ **Answer:** b) 14

---

20. What is the output of `"Python".upper()`?

- a) python
- b) PYTHON
- c) Python
- d) Error
  ✅ **Answer:** b) PYTHON

---

21. Which method is used to split a string into a list?

- a) split()
- b) separate()
- c) break()
- d) slice()
  ✅ **Answer:** a) split()

---

22. What is the output of `bool([])`?

- a) True
- b) False
- c) None
- d) Error
  ✅ **Answer:** b) False

---

23. Which operator is used for exponentiation in Python?

- a) ^
- b) **
- c) %

- d) //

  ✅ **Answer:** b) **

24. What is the result of `"Hello" + "World"`?

- a) HelloWorld
- b) Hello World
- c) Error
- d) Hello+World

  ✅ **Answer:** a) HelloWorld

25. Which of the following is used to define a block of code in Python?

- a) {}
- b) ()
- c) :
- d) ;

  ✅ **Answer:** c) :

26. What will `round(4.567, 2)` return?

- a) 4.6
- b) 4.57
- c) 4.56
- d) 4.5

  ✅ **Answer:** b) 4.57

27. Which function is used to read files in Python?

- a) read()
- b) fread()
- c) open()
- d) file()

  ✅ **Answer:** c) open()

28. What is the correct way to create a tuple?

- a) [1, 2, 3]
- b) {1, 2, 3}
- c) (1, 2, 3)
- d) <1, 2, 3>

✅ **Answer:** c) (1, 2, 3)

29. Which method is used to remove whitespace from the start and end of a string?

- a) strip()
- b) remove()
- c) trim()
- d) clean()

✅ **Answer:** a) strip()

30. What is the output of `3 == 3.0`?

- a) True
- b) False
- c) Error
- d) None

✅ **Answer:** a) True

31. What is the Python interpreter?

- a) Compiler
- b) Translator
- c) Interpreter
- d) Assembler

✅ **Answer:** c) Interpreter

32. What is the default Python interpreter name in Linux?

- a) python.exe
- b) py

- c) python
- d) cmd

✅ **Answer:** c) python

---

33. Which command is used to check the installed Python version?

- a) python --version
- b) python -v
- c) py -version
- d) python version

✅ **Answer:** a) python --version

---

34. What will happen if you type `python` in the terminal?

- a) Starts Python interactive mode
- b) Opens Python IDE
- c) Shows error
- d) Nothing

✅ **Answer:** a) Starts Python interactive mode

---

35. What is the extension of bytecode files generated by the Python interpreter?

- a) .class
- b) .obj
- c) .pyc
- d) .exe

✅ **Answer:** c) .pyc

---

36. What type of code does the Python interpreter execute?

- a) Assembly code
- b) Machine code
- c) Bytecode
- d) Source code

✅ **Answer:** c) Bytecode

---

37. How do you exit the Python interpreter in interactive mode?

- a) exit()
- b) quit()
- c) Ctrl + Z
- d) All of the above
  ✅ **Answer:** d) All of the above

38. What does the Python interpreter do before executing code?

- a) Compiles code into machine code
- b) Converts code to bytecode
- c) Converts code to binary
- d) Directly executes code
  ✅ **Answer:** b) Converts code to bytecode

39. How is the Python interpreter different from a compiler?

- a) It translates the code line by line
- b) It executes the entire code at once
- c) It generates machine code
- d) It optimizes the code
  ✅ **Answer:** a) It translates the code line by line

40. Which command is used to run a Python script from the terminal?

- a) run file.py
- b) python file.py
- c) py file.py
- d) Both b and c
  ✅ **Answer:** d) Both b and c

41. What is the primary role of the Python interpreter?

- a) Compile code
- b) Execute code
- c) Debug code

- d) Optimize code

  ✅ **Answer:** b) Execute code

---

42. Which function is used to display help in Python interpreter?

- a) help()
- b) info()
- c) guide()
- d) manual()

  ✅ **Answer:** a) help()

---

43. How do you restart the Python interpreter in the terminal?

- a) restart()
- b) Ctrl + C
- c) exit() and reopen
- d) refresh()

  ✅ **Answer:** c) exit() and reopen

---

44. What does `__pycache__` contain?

- a) Source code
- b) Bytecode files
- c) Log files
- d) Error logs

  ✅ **Answer:** b) Bytecode files

---

45. Which file format is used by the Python interpreter to store bytecode?

- a) .class
- b) .exe
- c) .pyc
- d) .jar

  ✅ **Answer:** c) .pyc

---

46. How can you run a Python script without opening the interpreter?

- a) python filename.py
- b) open filename.py
- c) run filename.py
- d) execute filename.py

✅ **Answer:** a) python filename.py

---

47. What will `python -m py_compile file.py` do?

- a) Run the program
- b) Compile into bytecode
- c) Open Python IDE
- d) Delete the file

✅ **Answer:** b) Compile into bytecode

---

48. What does the `sys.version` command return?

- a) Current Python version
- b) OS name
- c) Python IDE name
- d) Interpreter location

✅ **Answer:** a) Current Python version

---

49. Which module helps to interact with the Python interpreter?

- a) os
- b) sys
- c) time
- d) random

✅ **Answer:** b) sys

---

50. Which of the following can be used to install third-party libraries in Python interpreter?

- a) install
- b) apt-get

- c) pip
- d) pkg

✅ **Answer:** c) pip

51. What is the result of the expression `5 + 3` in Python?

- a) 53
- b) 8
- c) 15
- d) Error

✅ **Answer:** b) 8

52. Which operator is used for exponentiation in Python?

- a) ^
- b) **
- c) *
- d) %

✅ **Answer:** b) **

53. What will `10 / 3` return in Python?

- a) 3
- b) 3.0
- c) 3.33
- d) 3.3333333333333335

✅ **Answer:** d) 3.3333333333333335

54. What will be the output of `10 // 3`?

- a) 3
- b) 3.33
- c) 3.0
- d) 4

✅ **Answer:** a) 3

55. Which operator is used for modulus (remainder) in Python?

- a) %
- b) /
- c) //
- d) **

✅ **Answer:** a) %

56. What will `2 ** 3` return?

- a) 6
- b) 8
- c) 9
- d) 5

✅ **Answer:** b) 8

57. What will `7 % 3` return?

- a) 1
- b) 2
- c) 3
- d) 0

✅ **Answer:** b) 1

58. What will `round(3.456, 2)` return?

- a) 3.46
- b) 3.45
- c) 3.50
- d) 3.4

✅ **Answer:** a) 3.46

59. What is the result of `abs(-10)`?

- a) 10
- b) -10
- c) 0
- d) Error

✅ **Answer:** a) 10

60. What is the output of `max(4, 9, 2)`?

- a) 4
- b) 2
- c) 9
- d) Error

✅ **Answer:** c) 9

61. What will `min(3, 7, -2, 5)` return?

- a) 7
- b) 3
- c) -2
- d) 5

✅ **Answer:** c) -2

62. What will `10 + 2 * 3` return?

- a) 36
- b) 16
- c) 40
- d) 20

✅ **Answer:** b) 16

63. How do you calculate the square root of 16 in Python?

- a) sqrt(16)
- b) 16 ** 0.5
- c) square(16)
- d) power(16, 0.5)

✅ **Answer:** b) 16 ** 0.5

64. What is the result of `3 * 3 ** 2`?

- a) 27

- b) 18
- c) 9
- d) 81

✅ **Answer:** a) 27

---

65. What will `pow(2, 3)` return?

- a) 6
- b) 8
- c) 9
- d) 5

✅ **Answer:** b) 8

---

66. Which function is used to round off numbers in Python?

- a) round()
- b) ceil()
- c) floor()
- d) abs()

✅ **Answer:** a) round()

---

67. What will `divmod(10, 3)` return?

- a) (3, 1)
- b) (3.33, 1)
- c) (3, 0)
- d) (10, 3)

✅ **Answer:** a) (3, 1)

---

68. What is the result of `10 ** -2`?

- a) 0.1
- b) 0.01
- c) 100
- d) Error

✅ **Answer:** b) 0.01

---

69. Which operator has the highest precedence in Python?

- a) +
- b) *
- c) **
- d) %

✅ **Answer:** c) **

---

70. What is the result of `int(3.8)`?

- a) 3
- b) 4
- c) 3.8
- d) Error

✅ **Answer:** a) 3

71. What is Python Shell?

- a) Text Editor
- b) Interactive Interpreter
- c) Compiler
- d) Debugger

✅ **Answer:** b) Interactive Interpreter

---

72. How do you open the Python Shell on Windows?

- a) python
- b) python.exe
- c) py
- d) Both a and c

✅ **Answer:** d) Both a and c

---

73. What is the default prompt symbol in Python Shell?

- a) $

- b) >
- c) >>>
- d) #

  ✅ **Answer:** c) >>>

74. Which command exits the Python Shell?

- a) close()
- b) exit()
- c) quit()
- d) Both b and c

  ✅ **Answer:** d) Both b and c

75. How can you check the Python version inside the Python Shell?

- a) version()
- b) sys.version
- c) python --version
- d) shell.version()

  ✅ **Answer:** b) sys.version

76. What does the Python Shell allow you to do?

- a) Write scripts
- b) Execute code line by line
- c) Debug code
- d) All of the above

  ✅ **Answer:** d) All of the above

77. How do you enter multi-line code in Python Shell?

- a) Using {}
- b) Using ()
- c) Using backslash \
- d) Press Enter twice

  ✅ **Answer:** b) Using ()

---

78. Which module provides access to the Python Shell interpreter?

- a) shell
- b) sys
- c) os
- d) subprocess

✅ **Answer:** b) sys

---

79. What happens if you type `5 + 5` in the Python Shell?

- a) Error
- b) Nothing
- c) 10
- d) 55

✅ **Answer:** c) 10

---

80. How can you restart the Python Shell?

- a) restart()
- b) Ctrl + Z
- c) exit() and reopen
- d) refresh()

✅ **Answer:** c) exit() and reopen

---

81. What will `print("Hello")` display in the Python Shell?

- a) Hello
- b) "Hello"
- c) Error
- d) None

✅ **Answer:** a) Hello

---

82. Which method is used to display documentation of any function in Python Shell?

- a) help()
- b) doc()
- c) info()
- d) details()

  ✅ **Answer:** a) help()

---

83. What will `type(5)` return in the Python Shell?

- a) int
- b) float
- c) str
- d) bool

  ✅ **Answer:** a) int

---

84. What is the command to clear the Python Shell screen on Windows?

- a) clear
- b) cls
- c) clean
- d) reset

  ✅ **Answer:** b) cls (in terminal, not directly in shell)

---

85. How can you access the last result in Python Shell?

- a) $
- b) _
- c) @
- d) #

  ✅ **Answer:** b) _

---

86. Which keyword is used to define a function in Python Shell?

- a) func
- b) def
- c) function
- d) define

  ✅ **Answer:** b) def

87. How can you view all built-in functions in Python Shell?

- a) list()
- b) dir()
- c) help()
- d) methods()

  ✅ **Answer:** b) dir()

88. What will `3 ** 2` return in Python Shell?

- a) 6
- b) 9
- c) 32
- d) 12

  ✅ **Answer:** b) 9

89. What will `10 // 3` return in Python Shell?

- a) 3.33
- b) 3
- c) 4
- d) 3.0

  ✅ **Answer:** b) 3

90. How do you import a module in Python Shell?

- a) use module
- b) include module
- c) import module
- d) module()

  ✅ **Answer:** c) import module

91. What does indentation in Python indicate?

- a) Start of a comment
- b) Code block
- c) End of a line

- d) Function call

  ✅ **Answer:** b) Code block

---

92. What will happen if indentation is not used properly in Python?

- a) No effect
- b) Code will execute normally
- c) IndentationError
- d) SyntaxWarning

  ✅ **Answer:** c) IndentationError

---

93. How many spaces are recommended for indentation in Python?

- a) 2
- b) 3
- c) 4
- d) 5

  ✅ **Answer:** c) 4

---

94. Which error is raised if the indentation is incorrect?

- a) SyntaxError
- b) TypeError
- c) IndentationError
- d) ValueError

  ✅ **Answer:** c) IndentationError

---

95. Is indentation optional in Python?

- a) Yes
- b) No

  ✅ **Answer:** b) No

---

96. What is the purpose of indentation in Python?

- a) Improve performance
- b) Define code blocks
- c) Add comments
- d) Decorate code

  ✅ **Answer:** b) Define code blocks

---

97. Which of the following statements needs indentation?

- a) if statement
- b) for loop
- c) while loop
- d) All of the above

  ✅ **Answer:** d) All of the above

---

98. What will the following code output?

```
if 5 > 2:
    print("Five is greater")
print("End")
```

- a) Five is greater End
- b) Error
- c) End
- d) Five is greater

  ✅ **Answer:** a) Five is greater End

---

99. Can you mix tabs and spaces in Python indentation?

- a) Yes
- b) No

  ✅ **Answer:** b) No

---

100. What is the standard indentation size followed in Python PEP 8 guidelines?

- a) 2 spaces
- b) 4 spaces
- c) 6 spaces

- d) 8 spaces
  ✅ **Answer:** b) 4 spaces

101. What is an Atom in Python?

A) Smallest unit of code execution
B) Smallest element in a Python program
C) Smallest indivisible unit like numbers, strings, and constants
D) An operator

✅ **Answer:** C) Smallest indivisible unit like numbers, strings, and constants

---

102. Which of the following is an example of an Atom in Python?

A) 10
B) 'Hello'
C) [1, 2, 3]
D) All of the above

✅ **Answer:** D) All of the above

---

103. Which of the following are types of Atoms in Python?

A) Identifiers
B) Literals
C) Containers
D) All of the above

✅ **Answer:** D) All of the above

---

104. What will be the type of the following atom in Python?

```
3.14
```

A) int
B) float
C) complex
D) str

✅ **Answer:** B) float

105. Which of the following is NOT considered an atom in Python?

A) Tuple
B) Dictionary
C) String
D) If-else statement

✅ **Answer:** D) If-else statement

106. What type of Atom is used to represent a unique identifier in Python?

A) Numbers
B) Identifiers
C) Strings
D) Keywords

✅ **Answer:** B) Identifiers

107. What will be the output of the following code?

```
type(5 + 2j)
```

A) int
B) float
C) complex
D) str

✅ **Answer:** C) complex

108. Which of the following atoms is immutable in Python?

A) List
B) Dictionary
C) String
D) Set

✅ **Answer:** C) String

---

109. What is the type of the following atom?

```
(True)
```

A) int
B) bool
C) str
D) None

✅ **Answer:** B) bool

---

110. Which function is used to get the data type of any atom in Python?

A) type()
B) atom()
C) id()
D) isinstance()

✅ **Answer:** A) type()

1. Arithmetic Operators

1. What is the result of `5 + 3 * 2`?
   a) 16
   b) 11
   c) 13
   d) 10
   Answer: b) 11
2. What will `10 % 3` return?
   a) 1
   b) 3
   c) 0
   d) 10
   Answer: a) 1
3. What is the result of `2 ** 3` in Python?
   a) 6
   b) 8
   c) 9
   d) 16
   Answer: b) 8

4. What is the output of `10 // 3`?
   a) 3
   b) 3.33
   c) 4
   d) 3.0
   Answer: a) 3
5. What is the result of `-7 % 4`?
   a) 1
   b) -1
   c) 3
   d) -3
   Answer: c) 3

---

2. Relational (Comparison) Operators

6. What is the output of `5 == 5.0`?
   a) True
   b) False
   Answer: a) True
7. What is the output of `10 != 20`?
   a) True
   b) False
   Answer: a) True
8. What does `5 > 10` return?
   a) True
   b) False
   Answer: b) False
9. What is the output of `3 <= 3`?
   a) True
   b) False
   Answer: a) True
10. What is the result of `4.0 == 4`?
   a) True
   b) False
   Answer: a) True

---

3. Logical (Boolean) Operators

11. What is the result of `True and False`?
   a) True
   b) False
   Answer: b) False

12. What does `not True` return?
    a) True
    b) False
    Answer: b) False
13. What is the output of `True or False`?
    a) True
    b) False
    Answer: a) True
14. What is the result of `False and False`?
    a) True
    b) False
    Answer: b) False
15. What is the output of `not (10 > 5 and 5 < 2)`?
    a) True
    b) False
    Answer: a) True

---

4. Assignment Operators

16. What does `x += 5` mean?
    a) `x = x + 5`
    b) `x = x - 5`
    c) `x = x * 5`
    d) `x = x / 5`
    Answer: a) `x = x + 5`
17. If `x = 10`, what is `x -= 3`?
    a) 7
    b) 13
    c) -7
    d) 10
    Answer: a) 7
18. If `x = 4`, what is `x *= 2`?
    a) 2
    b) 8
    c) 4
    d) 6
    Answer: b) 8
19. If `y = 15`, what is `y //= 2`?
    a) 7.5
    b) 7
    c) 8
    d) 10
    Answer: b) 7

20. What is `x %= 4` equivalent to?
    a) `x = x % 4`
    b) `x = x + 4`
    c) `x = x * 4`
    d) `x = x // 4`
    Answer: a) `x = x %4

5. Ternary Operator

21. What does `x = 10 if 5 > 2 else 0` return?
    a) 10
    b) 0
    Answer: a) 10
22. What is the output of `y = "Even" if 4 % 2 == 0 else "Odd"`?
    a) Even
    b) Odd
    Answer: a) Even
23. What does `print(10 if False else 20)` output?
    a) 10
    b) 20
    Answer: b) 20
24. What is the syntax of the ternary operator?
    a) `if condition else expression`
    b) `expression if condition else expression`
    Answer: b) `expression if condition else expression`
25. What is the result of `x = 30 if 5 < 3 else 40`?
    a) 30
    b) 40
    Answer: b) 40

---

6. Bitwise Operators

26. What does `5 & 3` return?
    a) 1
    b) 3
    c) 2
    d) 5
    Answer: c) 1
27. What does `5 | 3` return?
    a) 7
    b) 8
    c) 3
    d) 5
    Answer: a) 7

28. What does `5 ^ 3` return?
    a) 6
    b) 2
    c) 1
    d) 3
    Answer: a) 6
29. What is `~5` in Python?
    a) -6
    b) -5
    c) 6
    d) 5
    Answer: a) -6
30. What is `8 >> 2`?
    a) 2
    b) 4
    c) 8
    d) 16
    Answer: b) 2

---

7. Increment and Decrement Operators

31. What does `x += 1` do?
    a) Increments x by 1
    b) Decrements x by 1
    Answer: a) Increments x by 1
32. How do you decrement a value in Python?
    a) `x--`
    b) `x -= 1`
    Answer: b) `x -= 1`
33. Does Python have `++` operator?
    a) Yes
    b) No
    Answer: b) No
34. What is the output of `x = 5; x += 2; print(x)`?
    a) 7
    b) 5
    c) 2
    Answer: a) 7
35. What happens when `x -= 3` is executed?
    a) x is decreased by 3
    b) x is increased by 3
    Answer: a) x is decreased by 3

8. Arithmetic Operators (More Questions)

36. What is the result of `7 * 3 + 6 / 2 - 4`?
    a) 21
    b) 20
    c) 22
    d) 23
    Answer: c) 22
37. What will be the output of `9 % 4`?
    a) 2
    b) 1
    c) 3
    d) 0
    Answer: b) 1
38. What does `-3 ** 2` evaluate to?
    a) -9
    b) 9
    c) -6
    d) 6
    Answer: a) -9 (Because exponentiation has higher precedence than unary minus)
39. What is `15 / 4` in Python 3?
    a) 3
    b) 3.75
    c) 4
    d) 3.0
    Answer: b) 3.75
40. What does `10 // 4` return?
    a) 2
    b) 2.5
    c) 2.0
    d) 3
    Answer: a) 2

---

9. Relational (Comparison) Operators (More Questions)

41. What is the result of `5 == 5.0` in Python?
    a) True
    b) False
    Answer: a) True
42. What does `10 != 5` return?
    a) True
    b) False
    Answer: a) True
43. What is the output of `4 > 2 and 3 < 1`?
    a) True

b) False
Answer: b) False

44. What is the result of `5 >= 5.0`?
    a) True
    b) False
    Answer: a) True
45. What does `7 < 8 and 8 > 9` return?
    a) True
    b) False
    Answer: b) False

1. Basic Input Statements

1. Which function is used to take user input in Python?
   a) `input()`
   b) `get()`
   c) `scan()`
   d) `read()`
   Answer: a) `input()`
2. What is the default data type of the value returned by `input()` in Python?
   a) int
   b) float
   c) str
   d) list
   Answer: c) `str`
3. How do you take an integer input from the user?
   a) `input(int())`
   b) `int(input())`
   c) `input().int()`
   d) `integer(input())`
   Answer: b) `int(input())`
4. What will happen if the user enters a non-numeric value when using `int(input())`?
   a) It will store the value as a string
   b) It will convert it to zero
   c) It will raise a `ValueError`
   d) It will return `None`
   Answer: c) It will raise a `ValueError`
5. How do you take multiple inputs in a single line in Python?
   a) `input().split()`
   b) `multi_input()`
   c) `input().read()`
   d) `get_multiple_inputs()`
   Answer: a) `input().split()`

---

2. Output Statements

6. Which function is used to display output in Python?
   a) `display()`
   b) `show()`
   c) `print()`
   d) `output()`
   Answer: c) `print()`
7. What will `print("Hello", "World")` output?
   a) HelloWorld

   b) Hello,World
   c) Hello World
   d) "Hello" "World"
   Answer: c) Hello World
8. What is the default separator used in the `print()` function?
   a) `,` (comma)
   b) `" "` (space)
   c) `"_"` (underscore)
   d) `"|"` (pipe)
   Answer: b) `" "` (space)
9. What does the `end` parameter in `print()` do?
   a) It specifies the separator between values
   b) It defines what is printed at the end instead of a newline
   c) It stops execution of the program
   d) It adds extra spaces at the end
   Answer: b) It defines what is printed at the end instead of a newline
10. What will be the output of `print("Hello", end="")`?
   a) Hello (without newline)
   b) Hello (with a newline)
   c) Hello (with space at the end)
   d) Syntax Error
   Answer: a) Hello (without newline)

---

3. String Formatting in Output

11. What is the correct way to format output using `format()`?
   a) `print("Name: {} Age: {}".format(name, age))`
   b) `print.format("Name: {} Age: {}", name, age)`
   c) `print("Name:", name, "Age:", age.format())`
   d) `print("Name: {} Age: {}".format) name, age`
   Answer: a) `print("Name: {} Age: {}".format(name, age))`
12. What is the output of `print("{0} {1}".format("Hello", "World"))`?
   a) World Hello
   b) Hello World
   c) {0} {1}
   d) Error
   Answer: b) Hello World
13. Which method is used for f-string formatting in Python?
   a) `print("Name is {name}")`
   b) `print(f"Name is {name}")`
   c) `print("Name is f{name}")`
   d) `print("Name is {}".format(name))`
   Answer: b) `print(f"Name is {name}")`

14. What is the output of `print(f"{5+3}")`?
    a) `{5+3}`
    b) `5+3`
    c) `8`
    d) `Error`
    Answer: c) `8`
15. What will `print(f"Value: {10:04d}")` output?
    a) Value: 0010
    b) Value: 10
    c) Value: 000010
    d) Value: 10.00
    Answer: a) Value: 0010

---

4. Advanced Input & Output

16. How do you read an entire file in Python?
    a) `file.read()`
    b) `file.input()`
    c) `read.file()`
    d) `file.get()`
    Answer: a) `file.read()`
17. How can you read multiple lines from a file?
    a) `file.readlines()`
    b) `file.read_lines()`
    c) `file.get_lines()`
    d) `file.read_all()`
    Answer: a) `file.readlines()`
18. What does `print("Python", "Programming", sep="-")` output?
    a) Python-Programming
    b) Python Programming
    c) Python,Programming
    d) Error
    Answer: a) Python-Programming
19. What is the default value of `end` in `print()`?
    a) `"\n"`
    b) `""` (empty string)
    c) `" "` (space)
    d) `None`
    Answer: a) `"\n"`
20. What does `print("Hello", end=" ")` do?
    a) Prints "Hello " without a newline
    b) Prints "Hello " with a newline
    c) Prints "Hello" without a space

d) Causes an error
Answer: a) Prints "Hello " without a newline

5. Miscellaneous Questions

21. What happens if `input()` is called with an argument?
    a) The argument is used as a prompt message
    b) It causes an error
    c) It is ignored
    d) The argument is stored in a variable
    Answer: a) The argument is used as a prompt message
22. Which statement is correct about `input().split()`?
    a) It splits input based on whitespace by default
    b) It returns a tuple
    c) It returns an integer
    d) It splits input by commas
    Answer: a) It splits input based on whitespace by default
23. What does `print(10, 20, 30, sep=":")` output?
    a) 10:20:30
    b) 10 20 30
    c) 10,20,30
    d) Error
    Answer: a) 10:20:30
24. Which of the following is **not** a valid `print()` function call?
    a) `print("Hello")`
    b) `print(Hello)`
    c) `print("Hello", end="!")`
    d) `print(10, 20, sep=",")`
    Answer: b) `print(Hello)` (since `Hello` is not in quotes)
25. How do you print a backslash (\) in Python?
    a) `print("/")`
    b) `print("\\")`
    c) `print("\"")`
    d) `print("\")`
    Answer: b) `print("\\")`

1. Conditional Statements (if, elif, else)

1. Which of the following is the correct syntax for an `if` statement in Python?
   a) `if condition {}`
   b) `if condition:`
   c) `if (condition) then:`
   d) `if condition do:`
   Answer: b) `if condition:`
2. What will be the output of the following code?

   ```
   if 0:
       print("True")
   else:
       print("False")
   ```

   a) True
   b) False
   c) Error
   d) None
   Answer: b) False

3. How do you check multiple conditions in an `if` statement?
   a) `if condition1, condition2:`
   b) `if condition1 and condition2:`
   c) `if condition1; condition2:`
   d) `if (condition1)(condition2):`
   Answer: b) `if condition1 and condition2:`
4. What is the purpose of `elif` in Python?
   a) To check multiple conditions
   b) To exit the program
   c) To repeat a loop
   d) To define a function
   Answer: a) To check multiple conditions
5. What will be the output of this code?

   ```
   x = 5
   if x > 10:
       print("Greater")
   elif x > 3:
       print("Medium")
   else:
       print("Smaller")
   ```

   a) Greater
   b) Medium
   c) Smaller
   d) None
   Answer: b) Medium

---

2. Looping Statements (for, while, nested loops)

6. What is the output of `for i in range(3): print(i)`?
   a) 1 2 3
   b) 0 1 2
   c) 0 1 2 3
   d) 1 2
   Answer: b) 0 1 2
7. What does the `range(2, 5)` generate?
   a) 2, 3, 4, 5
   b) 2, 3, 4
   c) 3, 4, 5
   d) 2, 5
   Answer: b) 2, 3, 4
8. What is the correct syntax for a `while` loop?
   a) `while (condition):`
   b) `while condition {}`
   c) `while condition do:`
   d) `while (condition) then:`
   Answer: a) `while (condition):`
9. What is the output of the following code?

   ```
   i = 1
   while i < 4:
       print(i)
       i += 1
   ```

   a) 1 2 3 4
   b) 1 2 3
   c) 2 3 4
   d) Infinite loop
   Answer: b) 1 2 3

10. Which statement is used to exit a loop early?
    a) `exit()`
    b) `break`
    c) `continue`
    d) `stop`
    Answer: b) `break`

---

3. Break, Continue, and Pass

11. What does the `break` statement do in Python loops?
    a) Skips the current iteration
    b) Exits the loop
    c) Does nothing
    d) Causes an error
    Answer: b) Exits the loop
12. What does the `continue` statement do?
    a) Stops execution
    b) Skips the current iteration and continues with the next
    c) Exits the loop
    d) Executes the loop body again
    Answer: b) Skips the current iteration and continues with the next
13. What does `pass` do in Python?
    a) Exits the program
    b) Does nothing; it is a placeholder
    c) Stops the loop
    d) Prints an error message
    Answer: b) Does nothing; it is a placeholder
14. What is the output of this code?

    ```
    python
    CopyEdit
    for i in range(5):
        if i == 3:
            break
        print(i)
    ```

    a) 0 1 2 3 4
    b) 0 1 2
    c) 1 2 3
    d) 0 1 2 3
    Answer: b) 0 1 2

15. What is the output of this code?

    ```
    for i in range(5):
        if i == 3:
            continue
        print(i)
    ```

    a) 0 1 2 3 4
    b) 0 1 2 4
    c) 1 2 3 4
    d) 0 1 2
    Answer: b) 0 1 2 4

---

4. Exit Function (`sys.exit()`)

16. Which module provides the `exit()` function?
    a) `os`
    b) `sys`
    c) `time`
    d) `exit`
    Answer: b) `sys`
17. What does `sys.exit()` do?
    a) Ends the current function
    b) Terminates the entire program
    c) Skips the current loop iteration
    d) Restarts the program
    Answer: b) Terminates the entire program
18. What will happen if `sys.exit()` is called inside a loop?
    a) The loop will stop and the program will terminate
    b) The loop will continue running
    c) An error will occur
    d) Only the loop will exit, but not the program
    Answer: a) The loop will stop and the program will terminate
19. What argument can be passed to `sys.exit()`?
    a) String
    b) Integer
    c) No argument
    d) All of the above
    Answer: d) All of the above
20. What is the default exit status when `sys.exit()` is called without arguments?
    a) 0
    b) 1
    c) -1
    d) None
    Answer: a) 0

5. Nested Loops and Loop Control

21. What is the output of the following code?

```
for i in range(2):
    for j in range(2):
        print(i, j)
```

    a) (0,0) (0,1) (1,0) (1,1)
    b) (0,1) (1,0) (1,1)
    c) (0,0) (0,1) (1,0)
    d) (0,0) (1,0)
    Answer: a) (0,0) (0,1) (1,0) (1,1)

22. How many times will the following nested loop execute?

```
python
CopyEdit
for i in range(3):
    for j in range(2):
        print(i, j)
```

a) 3 times
b) 2 times
c) 5 times
d) 6 times
Answer: d) 6 times

23. What happens when a `break` statement is encountered inside a nested loop?
a) Only the inner loop exits
b) Both inner and outer loops exit
c) The program stops execution
d) The loop skips one iteration
Answer: a) Only the inner loop exits

24. What will be the output of this program?

```
for i in range(3):
    for j in range(3):
        if i == j:
            break
        print(i, j)
```

a) (0,0) (1,0) (1,1) (2,0) (2,1)
b) (1,0) (2,0) (2,1)
c) (0,0) (1,1) (2,2)
d) (1,0) (2,0)
Answer: b) (1,0) (2,0) (2,1)

9. Advanced Looping Concepts

41. What will be the output of this code?

```
for i in range(1, 5, 2):
    print(i, end=" ")
```

a) `1 2 3 4`
b) `1 3`
c) `1 2`
d) `1 3 5`
Answer: b) `1 3`

42. How many times will this `while` loop execute?

```
i = 1
while i < 10:
    i *= 2
```

a) 3 times
b) 4 times
c) 5 times
d) Infinite
Answer: c) 5 times

43. What will be the output of the following loop?

```
for i in range(1, 6):
    if i % 2 == 0:
        continue
    print(i, end=" ")
```

a) `1 2 3 4 5`
b) `1 3 5`
c) `2 4`
d) `None`
Answer: b) `1 3 5`

44. What is the purpose of `else` in a loop?
a) Executes when the loop condition is False
b) Executes only if the loop is exited using `break`
c) Executes only if the loop completes normally (without `break`)
d) Does nothing
Answer: c) Executes only if the loop completes normally (without `break`)

45. What is the output of this program?

```
for i in range(3):
    print(i)
else:
    print("Loop completed")
```

a) `0 1 2`
b) `0 1 2 Loop completed`
c) `Loop completed`
d) `0 1 Loop completed`
Answer: b) `0 1 2 Loop completed`

---

10. More on `break`, `continue`, and `pass`

46. What happens if `break` is used inside a `for` loop?
a) The loop skips one iteration

b) The loop exits completely
c) The program crashes
d) The loop continues executing
Answer: b) The loop exits completely

47. What is the output of this program?

```
for i in range(3):
    pass
print("Done")
```

a) `0 1 2 Done`
b) `Done`
c) `Pass Done`
d) `Error`
Answer: b) `Done`

48. What will be the output of the following code?

```
i = 1
while i < 5:
    if i == 3:
        break
    print(i)
    i += 1
```

a) `1 2 3 4`
b) `1 2`
c) `1 2 4`
d) `3 4`
Answer: b) `1 2`

49. What is the output of this program?

```
for i in range(5):
    if i == 3:
        pass
    print(i)
```

a) `0 1 2 4`
b) `0 1 2 3 4`
c) `0 1 2`
d) `1 2 3 4`
Answer: b) `0 1 2 3 4`

50. What will happen if `continue` is replaced with `pass` in a loop?
a) The program will exit
b) The loop will behave the same
c) The loop will stop executing

d) The loop will throw an error
Answer: b) The loop will behave the same

---

11. More on `while` and `for` Loops

51. What will be the output?

```
i = 5
while i > 0:
    print(i, end=" ")
    i -= 2
```

a) `5 3 1`
b) `5 4 3 2 1`
c) `5 3`
d) `5 4 3 2`
Answer: a) `5 3 1`

52. What is the output of the following loop?

```
for i in range(1, 6, 2):
    print(i, end=" ")
```

a) `1 2 3 4 5`
b) `1 3 5`
c) `1 3`
d) `1 2`
Answer: b) `1 3 5`

53. How can you create an infinite loop using `while`?
a) `while False:`
b) `while 1:`
c) `while True:`
d) `while i == 0:`
Answer: c) `while True:`

54. What happens if the loop condition never becomes False?
a) The loop runs indefinitely
b) The loop runs once and stops
c) The loop never runs
d) The program throws an error
Answer: a) The loop runs indefinitely

55. What will be the output of this loop?

```
for i in range(2):
    print("Hello")
```

a) `Hello Hello`
b) `Hello`
c) `Error`
d) `None`
Answer: a) `Hello Hello`

---

12. Miscellaneous Questions

56. What is the default starting value of `range(n)` in Python?
    a) 0
    b) 1
    c) n
    d) Undefined
    Answer: a) 0
57. What is the step value in `range(2, 10, 2)`?
    a) 2
    b) 10
    c) 8
    d) 1
    Answer: a) 2
58. What is the output of `list(range(5))`?
    a) `[1, 2, 3, 4, 5]`
    b) `[0, 1, 2, 3, 4]`
    c) `[0, 1, 2, 3, 4, 5]`
    d) `[0, 1, 2, 3]`
    Answer: b) `[0, 1, 2, 3, 4]`
59. Which of the following is used to exit a function?
    a) `break`
    b) `return`
    c) `continue`
    d) `pass`
    Answer: b) `return`
60. Which function is used to immediately terminate a program?
    a) `exit()`
    b) `sys.exit()`
    c) `return`
    d) `continue`
    Answer: b) `sys.exit()`

1. Basics of Function Definition

1. How do you define a function in Python?
   a) `def function_name():`
   b) `define function_name():`
   c) `function function_name():`
   d) `func function_name():`
   Answer: a) `def function_name():`
2. What is the output of the following code?

   ```
   def greet():
       print("Hello, World!")
   greet()
   ```

   a) `Hello, World!`
   b) `None`
   c) `greet()`
   d) Error
   Answer: a) `Hello, World!`

3. Which keyword is used to define a function in Python?
   a) `function`
   b) `define`
   c) `def`
   d) `fun`
   Answer: c) `def`
4. What will be the output of this function?

   ```
   def my_func():
       return 5
   print(my_func())
   ```

   a) `5`
   b) `None`
   c) Error
   d) `my_func()`
   Answer: a) `5`

5. Which of the following is correct syntax to return multiple values from a function?
   a) `return a, b, c`
   b) `return [a, b, c]`
   c) `return (a, b, c)`
   d) All of the above
   Answer: d) All of the above

---

2. Function Arguments

6. What is the default return value of a function that does not have a return statement?
   a) `0`
   b) `None`
   c) `Error`
   d) `Empty string`
   Answer: b) `None`
7. How many arguments can a Python function take?
   a) Only 1
   b) At most 5
   c) Any number of arguments
   d) None
   Answer: c) Any number of arguments
8. What will be the output of this function?

```
def test(a, b=3, c=5):
    print(a, b, c)
test(10, 20)
```

   a) `10 20 5`
   b) `10 3 5`
   c) `10 20`
   d) Error
   Answer: a) `10 20 5`

9. What happens if a function does not receive enough arguments?
   a) `TypeError`
   b) `SyntaxError`
   c) `None` is passed
   d) The function executes without errors
   Answer: a) `TypeError`
10. Which of the following statements is correct?
    a) Default arguments must be at the beginning
    b) Default arguments must be at the end
    c) Default arguments can be anywhere
    d) Default arguments are not allowed in Python
    Answer: b) Default arguments must be at the end

---

3. Default Arguments in Functions

11. What is the output of the following function call?

```
def func(x=2, y=3):
    return x * y
```

```
print(func(4))
```

a) `6`
b) `12`
c) `9`
d) `8`
Answer: b) `12`

12. Can a function have both required and default arguments?
    a) Yes
    b) No
    Answer: a) Yes
13. Which function call is correct for the following function?

```
def greet(name="Guest"):
    print("Hello,", name)
```

a) `greet()`
b) `greet("Alice")`
c) Both a and b
d) None of the above
Answer: c) Both a and b

14. What happens if you pass all arguments to a function with default values?
    a) Default values are ignored
    b) Error occurs
    c) Default values are used
    d) None of the above
    Answer: a) Default values are ignored
15. Which of the following statements is correct regarding default arguments?
    a) Default arguments can only be strings
    b) Default arguments can only be integers
    c) Default arguments can be any data type
    d) Default arguments must always be 0
    Answer: c) Default arguments can be any data type

---

4. Keyword and Positional Arguments

16. What is a keyword argument in Python?
    a) An argument passed without a value
    b) An argument passed with a name-value pair
    c) An argument that cannot be changed
    d) An argument with a default value
    Answer: b) An argument passed with a name-value pair
17. What is the output of this function call?

```
def my_func(a, b=2, c=3):
    print(a, b, c)
my_func(c=10, a=5)
```

a) `5 2 10`
b) `5 10 2`
c) `10 2 5`
d) Error
Answer: a) `5 2 10`

18. Can a function have both positional and keyword arguments?
    a) Yes
    b) No
    Answer: a) Yes
19. What is the output of this function?

```
def test(a, b=2, c=3):
    print(a, b, c)
test(10, c=15)
```

a) `10 2 15`
b) `10 15 2`
c) Error
d) `15 2 10`
Answer: a) `10 2 15`

20. Which of the following is NOT a valid function call?

```
def add(a, b=5, c=10):
    return a + b + c
```

a) `add(3, 4, 5)`
b) `add(3, c=8)`
c) `add(a=2, 3)`
d) `add(3)`
Answer: c) `add(a=2, 3)`

---

5. Miscellaneous Questions

21. Can a function have multiple return values?
    a) Yes, using tuples
    b) No, functions return only one value
    Answer: a) Yes, using tuples
22. What does the `return` statement do?
    a) Stops function execution

b) Returns a value to the caller
c) Both a and b
d) None of the above
Answer: c) Both a and b

23. What is the output of this code?

```
def add(a, b):
    return a + b
print(add(2, 3) + add(4, 5))
```

a) `14`
b) `9`
c) `Error`
d) `25`
Answer: a) `14`

24. Can a function return multiple values using a list?
a) Yes
b) No
Answer: a) Yes

25. What is the output of this function?

```
def hello():
    return "Hi"
    print(hello())
```

a) `Hi`
b) `None`
c) `hello()`
d) Error
Answer: a) `Hi`

1. Errors and Exceptions (15 Questions)

1. What type of error is caused by incorrect Python syntax?
   a) Logical Error
   b) Runtime Error
   c) Syntax Error
   d) Indentation Error
   Answer: c) Syntax Error
2. What does the following code raise?

   ```
   print(10 / 0)
   ```

   a) `TypeError`
   b) `ZeroDivisionError`
   c) `SyntaxError`
   d) `ValueError`
   Answer: b) `ZeroDivisionError`

3. Which exception is raised when trying to access a list index that doesn't exist?
   a) `KeyError`
   b) `IndexError`
   c) `TypeError`
   d) `AttributeError`
   Answer: b) `IndexError`
4. What will be the output of this program?

   ```
   try:
       print(5 / 0)
   except ZeroDivisionError:
       print("Cannot divide by zero")
   ```

   a) `ZeroDivisionError`
   b) `Cannot divide by zero`
   c) `None`
   d) `Runtime Error`
   Answer: b) `Cannot divide by zero`

5. What does `except:` do in Python?
   a) Catches all errors
   b) Catches only syntax errors
   c) Ignores exceptions
   d) Terminates the program
   Answer: a) Catches all errors
6. What will be the output of this code?

   ```
   try:
       x = int("hello")
   ```

```
except ValueError:
    print("Invalid conversion")
```

a) `Invalid conversion`
b) `ValueError`
c) `hello`
d) `TypeError`
Answer: a) `Invalid conversion`

7. Which keyword is used to manually raise an exception?
   a) `raise`
   b) `throw`
   c) `except`
   d) `assert`
   Answer: a) `raise`
8. Which exception is raised when a dictionary key is not found?
   a) `KeyError`
   b) `IndexError`
   c) `TypeError`
   d) `ValueError`
   Answer: a) `KeyError`
9. What does the `finally` block do in exception handling?
   a) Executes only if an error occurs
   b) Always executes
   c) Executes only if no errors occur
   d) Skips the `try` block
   Answer: b) Always executes
10. What is the output of this code?

```
try:
    x = 5 / 0
finally:
    print("Finally block executed")
```

a) `Finally block executed`
b) `ZeroDivisionError`
c) Both a and b
d) No output
Answer: c) Both a and b

---

2. Iteration and Recursion (10 Questions)

11. What will the following `while` loop do?

```
x = 1
```

```
while x < 5:
    print(x)
    x += 1
```

a) Print numbers from 1 to 5
b) Print numbers from 1 to 4
c) Infinite loop
d) Error
Answer: b) Print numbers from 1 to 4

12. What is the base case in recursion?
    a) The function calling itself
    b) The function ending the recursion
    c) Infinite loop
    d) Function execution stopping
    Answer: b) The function ending the recursion
13. Which of the following is NOT a characteristic of recursion?
    a) Base condition
    b) Function calling itself
    c) Looping mechanism
    d) Stack memory usage
    Answer: c) Looping mechanism
14. What will this recursive function output?

```
def count(n):
    if n == 0:
        return
    print(n)
    count(n-1)
count(3)
```

a) `3 2 1`
b) `1 2 3`
c) `Error`
d) Infinite loop
Answer: a) `3 2 1`

---

3. Conditional Execution & Alternative Execution (10 Questions)

15. What is the output of this code?

```
if 3 > 2:
    print("Yes")
```

a) `Yes`
b) `No`

c) `Error`
d) `None`
Answer: a) `Yes`

16. What will be the output of the following?

```
a = 10
b = 20
if a > b:
    print("A")
else:
    print("B")
```

a) `A`
b) `B`
c) `None`
d) `Error`
Answer: b) `B`

17. What is the correct syntax of an `if-else` statement?
    a) `if (condition) {}`
    b) `if condition:`
    c) `if condition {} else {}`
    d) `if (condition): {}`
    Answer: b) `if condition:`

---

4. Nested Conditionals & Return Statement (15 Questions)

18. What will be the output of this code?

```
x = 10
if x > 5:
    if x < 15:
        print("Nested")
```

a) `Nested`
b) `Error`
c) `No output`
d) `15`
Answer: a) `Nested`

19. What will be the output of this function?

```
def func():
    return 5
    print("Hello")
print(func())
```

a) `Hello`
b) `5 Hello`
c) `5`
d) Error
Answer: c) `5`

20. What happens if a function does not have a return statement?
    a) Returns `None`
    b) Causes an error
    c) Returns `0`
    d) Infinite loop
    Answer: a) Returns `None`
21. What is the correct way to return two values from a function?
    a) `return a, b`
    b) `return [a, b]`
    c) `return (a, b)`
    d) All of the above
    Answer: d) All of the above

1. Errors and Exceptions in Python

1. What type of error occurs when a variable is used before it is defined?
   a) SyntaxError
   b) NameError
   c) TypeError
   d) IndexError
   Answer: b) NameError
2. What will happen when the following code is executed?

   ```
   print(10 / 0)
   ```

   a) 0
   b) Infinity
   c) ZeroDivisionError
   d) None
   Answer: c) ZeroDivisionError

3. Which of the following is NOT a built-in exception in Python?
   a) ValueError
   b) MemoryError
   c) KeyError
   d) FileNotFound
   Answer: d) FileNotFound (Correct exception is `FileNotFoundError`)
4. Which keyword is used to handle exceptions in Python?
   a) `try`
   b) `catch`

c) `except`
d) Both a and c
Answer: d) Both a and c

5. What does the `finally` block do in exception handling?
   a) Executes only if an exception occurs
   b) Executes only if no exception occurs
   c) Always executes, regardless of exceptions
   d) None of the above
   Answer: c) Always executes, regardless of exceptions

---

2. Iteration in Python

6. Which loop is preferred when the number of iterations is known?
   a) `for`
   b) `while`
   c) `do-while`
   d) None of the above
   Answer: a) `for`
7. What is the output of the following loop?

   ```
   for i in range(3):
       print(i, end=" ")
   ```

   a) `0 1 2`
   b) `1 2 3`
   c) `0 1 2 3`
   d) `None`
   Answer: a) `0 1 2`

8. What happens if a `break` statement is used inside a loop?
   a) Exits the loop immediately
   b) Skips the next iteration
   c) Throws an error
   d) Repeats the loop
   Answer: a) Exits the loop immediately
9. What will be the output of this code?

   ```
   for i in range(3):
       if i == 1:
           continue
       print(i, end=" ")
   ```

   a) `0 1 2`
   b) `0 2`
   c) `1 2`

d) `None`
Answer: b) `0 2`

10. What is the output of the following code?

```
x = 0
while x < 3:
    print(x)
    x += 1
```

a) `0 1 2`
b) `1 2 3`
c) `0 1 2 3`
d) Infinite loop
Answer: a) `0 1 2`

---

3. Recursion in Python

11. What is recursion?
    a) A function calling another function
    b) A function calling itself
    c) A function that loops indefinitely
    d) A function with no return statement
    Answer: b) A function calling itself
12. What will happen if recursion is used without a base case?
    a) Program executes successfully
    b) RecursionError occurs
    c) Function stops execution automatically
    d) None of the above
    Answer: b) RecursionError occurs
13. What is the base case in recursion?
    a) The smallest input for which recursion stops
    b) The main function in a program
    c) The first call to a function
    d) None of the above
    Answer: a) The smallest input for which recursion stops
14. What is the output of this recursive function?

```
def func(n):
    if n == 0:
        return 1
    return n * func(n - 1)
print(func(3))
```

a) `3`
b) `6`

c) `9`
d) `None`
Answer: b) `6`

15. Which data structure is used internally in recursion?
    a) Queue
    b) Stack
    c) Linked List
    d) Array
    Answer: b) Stack

---

4. Conditional Execution

16. What will be the output of the following statement?

```
if 0:
    print("Hello")
```

    a) `Hello`
    b) Nothing
    c) `Error`
    d) `0`
    Answer: b) Nothing

17. What is the output of this code?

```
a = 5
if a < 10:
    print("Less than 10")
```

    a) `Less than 10`
    b) `Error`
    c) `None`
    d) `Nothing`
    Answer: a) `Less than 10`

---

5. Alternative Execution

18. What is alternative execution in Python?
    a) Multiple `if` statements
    b) `if-else` statements
    c) `for` loops

d) `while` loops
Answer: b) `if-else` statements

19. What is the output of this code?

```
num = 7
if num % 2 == 0:
    print("Even")
else:
    print("Odd")
```

a) `Even`
b) `Odd`
c) `None`
d) `Error`
Answer: b) `Odd`

---

6. Nested Conditionals

20. What is a nested conditional?
a) A loop inside an `if` statement
b) An `if` inside an `if`
c) A function inside a loop
d) A conditional inside a function
Answer: b) An `if` inside an `if`

21. What will be the output of this code?

```
x = 5
if x > 0:
    if x < 10:
        print("Between 0 and 10")
```

a) `Between 0 and 10`
b) `Nothing`
c) `Error`
d) `None`
Answer: a) `Between 0 and 10`

---

7. Return Statement in Python

22. What does the `return` statement do?
a) Terminates function execution
b) Returns a value to the caller
c) Both a and b

d) None of the above
Answer: c) Both a and b

23. What is the default return value of a function without a `return` statement?
a) `None`
b) `0`
c) `False`
d) `Error`
Answer: a) `None`

1. Errors & Exceptions (10 Questions)

1. What is the output of this code?

```
try:
    print(5 / 0)
except:
    print("Exception Occurred")
```

a) `Exception Occurred`
b) `ZeroDivisionError`
c) `None`
d) `5 / 0`
Answer: a) `Exception Occurred`

2. What type of error will the following code produce?

```
print("Hello)
```

a) `SyntaxError`
b) `NameError`
c) `IndentationError`
d) `TypeError`
Answer: a) `SyntaxError`

3. What will be the output of this code?

```
try:
    x = int("hello")
except ValueError:
    print("Invalid conversion")
```

a) `Invalid conversion`
b) `ValueError`
c) `TypeError`
d) `hello`
Answer: a) `Invalid conversion`

4. What does the `finally` block do in a try-except statement?
   a) Runs only if an exception occurs
   b) Always executes
   c) Executes only if no error occurs
   d) Skips error handling
   Answer: b) Always executes
5. What will be the output of this code?

```
try:
    print(1 / 0)
except ZeroDivisionError:
    print("Cannot divide by zero")
finally:
    print("Execution completed")
```

   a) `Cannot divide by zero Execution completed`
   b) `Execution completed Cannot divide by zero`
   c) `ZeroDivisionError`
   d) `None`
   Answer: a) `Cannot divide by zero Execution completed`

6. What keyword is used to define a custom exception?
   a) `def`
   b) `raise`
   c) `except`
   d) `try`
   Answer: b) `raise`
7. What is the correct syntax for handling multiple exceptions?
   a) `except (TypeError, ValueError):`
   b) `except TypeError, ValueError:`
   c) `except TypeError or ValueError:`
   d) `except TypeError and ValueError:`
   Answer: a) `except (TypeError, ValueError):`
8. What happens when an exception is raised inside a function?
   a) The program stops immediately
   b) The function execution stops and control moves to the caller
   c) The program ignores the exception
   d) The function continues execution
   Answer: b) The function execution stops and control moves to the caller
9. What will the following code output?

```
try:
    num = int(input("Enter a number: "))
except ValueError:
    print("Invalid input")
else:
    print("Valid input")
```

If the user enters `"abc"`, what is printed?
a) `Invalid input`
b) `Valid input`
c) `None`
d) `Error`
Answer: a) `Invalid input`

10. Which error occurs if you try to open a file that does not exist?
    a) `FileNotFoundError`
    b) `IOError`
    c) `SyntaxError`
    d) `TypeError`
    Answer: a) `FileNotFoundError`

---

2. Iteration & Recursion (10 Questions)

11. Which of the following is NOT a loop in Python?
    a) `for`
    b) `while`
    c) `do-while`
    d) `nested`
    Answer: c) `do-while`
12. What is the output of this loop?

```
x = 0
while x < 3:
    print(x)
    x += 1
```

a) `0 1 2`
b) `1 2 3`
c) `0 1 2 3`
d) `Infinite loop`
Answer: a) `0 1 2`

13. What will happen if the base condition is not defined in a recursive function?
    a) Infinite recursion
    b) Syntax error
    c) Logical error
    d) Stops execution
    Answer: a) Infinite recursion
14. What will this recursive function print?

```
def count(n):
    if n == 0:
        return
    print(n)
    count(n-1)
count(3)
```

a) `3 2 1`
b) `1 2 3`
c) `0`
d) `None`
Answer: a) `3 2 1`

15. Which of the following is NOT a termination condition for recursion?
    a) `if` condition
    b) Stack overflow
    c) Break statement
    d) Return statement
    Answer: c) Break statement

---

3. Conditional Execution, Alternative Execution, Nested Conditionals, Return Statement (10 Questions)

16. What is the output of this conditional statement?

```
if 5 > 3:
    print("Yes")
```

a) `Yes`
b) `No`
c) `Error`
d) `None`
Answer: a) `Yes`

17. What is the correct syntax of an `if-else` statement?
    a) `if (condition) {}`
    b) `if condition:`
    c) `if condition {} else {}`
    d) `if (condition): {}`
    Answer: b) `if condition:`
18. Which of the following returns `None`?
    a) `print()`
    b) `return`
    c) `input()`
    d) `len("test")`
    Answer: a) `print()`

19. What is the correct way to return multiple values from a function?
    a) `return a, b`
    b) `return [a, b]`
    c) `return (a, b)`
    d) All of the above
    Answer: d) All of the above
20. What will this function return?

```
def add(a, b):
    return a + b
print(add(2, 3))
```

a) `5`
b) `2, 3`
c) `None`
d) `Error`
Answer: a) `5`

Unit – 3

1. What is recursion in Python?

A) A function calling itself
B) A function calling another function
C) A loop inside a function
D) A function without return statement

✅ **Answer:** A) A function calling itself

---

2. What is the base case in recursion?

A) The part where function returns itself
B) The condition where recursion ends
C) The first call of the function
D) A loop condition

✅ **Answer:** B) The condition where recursion ends

---

3. What happens if the base case is not defined in recursion?

A) Infinite loop
B) Error occurs
C) Function will execute once
D) Code will not compile

✅ **Answer:** A) Infinite loop

---

4. What is the maximum recursion depth in Python by default?

A) 100
B) 500
C) 1000
D) 2000

✅ **Answer:** C) 1000

---

5. Which module can be used to change the recursion limit in Python?

A) sys
B) os
C) math
D) random

✅ **Answer:** A) sys

---

6. What is the output of the following code?

```
def fun(n):
    if n == 0:
        return 1
    return n * fun(n-1)
print(fun(4))
```

A) 24
B) 10
C) 4
D) 120

✅ **Answer:** A) 24

---

7. Which of the following problems is solved using recursion?

A) Fibonacci Series
B) Factorial Calculation
C) Tower of Hanoi
D) All of the above

✅ **Answer:** D) All of the above

---

8. What is the output of the following code?

```
def fun(n):
    if n <= 0:
        return 0
    return n + fun(n-1)
```

```
print(fun(3))
```

A) 6
B) 3
C) 5
D) 4

✅ **Answer:** A) 6

9. What is the purpose of the base case in recursion?

A) To stop the recursive calls
B) To print the result
C) To increase recursion depth
D) To repeat the loop

✅ **Answer:** A) To stop the recursive calls

10. What is tail recursion?

A) When the recursive call is the last statement
B) When recursion happens at the start
C) When recursion uses two functions
D) When recursion happens in loops

✅ **Answer:** A) When the recursive call is the last statement

11. What is the output of the following code?

```
def fun(n):
    if n == 1:
        return 1
    return n * fun(n-1)
print(fun(5))
```

A) 15
B) 20
C) 120
D) 25

✅ **Answer:** C) 120

---

12. Which keyword is used to define a function in Python?

A) def
B) function
C) func
D) lambda

✅ **Answer:** A) def

---

13. What happens if recursion depth exceeds the limit in Python?

A) Memory error
B) RecursionError
C) TypeError
D) ValueError

✅ **Answer:** B) RecursionError

---

14. What is the smallest possible recursion in Python?

A) 1
B) 0
C) -1
D) None

✅ **Answer:** B) 0

---

15. What is the output of the following code?

```
def fun(n):
    if n <= 0:
        return n
    return fun(n-1)
print(fun(5))
```

A) 0
B) 5
C) 1
D) Error

✅ **Answer:** A) 0

16. What is indirect recursion?

A) When two or more functions call each other
B) When a function calls itself directly
C) When recursion happens without stopping
D) When recursion happens in loops

✅ **Answer:** A) When two or more functions call each other

17. What is the output of the following code?

```
def fun(n):
    if n == 0:
        return 1
    else:
        return n * fun(n-1)
print(fun(3))
```

A) 6
B) 3
C) 9
D) 1

✅ **Answer:** A) 6

18. Which function is used to set the maximum recursion depth in Python?

A) sys.setrecursionlimit()
B) os.setrecursionlimit()
C) math.setrecursionlimit()
D) recursion.setlimit()

✅ **Answer:** A) sys.setrecursionlimit()

19. Which of the following is not an example of recursion?

A) Fibonacci series
B) Factorial
C) Loop iteration
D) Tower of Hanoi

✅ **Answer:** C) Loop iteration

20. What is the advantage of recursion?

A) Simplifies code
B) Faster execution
C) Uses less memory
D) No advantage

✅ **Answer:** A) Simplifies code

21. What is a stack diagram in recursion?

A) A diagram showing variables
B) A diagram showing how functions call each other
C) A diagram showing loops
D) A diagram showing memory usage

✅ **Answer:** B) A diagram showing how functions call each other

22. What does each stack frame represent in a stack diagram?

A) A variable
B) A function call
C) A loop
D) A return statement

✅ **Answer:** B) A function call

23. What happens to the stack when a function is called?

A) A new frame is added
B) A frame is removed
C) The stack remains the same
D) The stack is cleared

✅ **Answer:** A) A new frame is added

---

24. What happens to the stack when a function returns?

A) A new frame is added
B) The top frame is removed
C) The stack doubles in size
D) The stack becomes empty

✅ **Answer:** B) The top frame is removed

---

25. What does the base case represent in a recursive function's stack diagram?

A) The first function call
B) The middle function call
C) The last function call before returning
D) A loop

✅ **Answer:** C) The last function call before returning

---

26. What happens if there is no base case in recursion?

A) The program executes normally
B) Infinite recursion occurs
C) The function ends quickly
D) The program skips the function

✅ **Answer:** B) Infinite recursion occurs

---

27. What is the first function call at the bottom of the stack called?

A) Base case
B) Recursive call
C) Initial call
D) Loop

✅ **Answer:** C) Initial call

28. How is the stack diagram processed in recursion?

A) Top to bottom
B) Bottom to top
C) Left to right
D) Randomly

✅ **Answer:** B) Bottom to top

29. What is the purpose of a stack diagram?

A) To show loops
B) To understand how recursion works
C) To remove recursion
D) To print output

✅ **Answer:** B) To understand how recursion works

30. Which data structure is used internally to handle recursion?

A) Queue
B) List
C) Stack
D) Array

✅ **Answer:** C) Stack

31. What is stored in a stack frame?

A) Function name, arguments, and return address
B) Only variables
C) Only arguments
D) Loops

✅ **Answer:** A) Function name, arguments, and return address

---

32. What is the maximum depth of the stack in Python by default?

A) 500
B) 1000
C) 2000
D) Unlimited

✅ **Answer:** B) 1000

---

33. What happens if the recursion depth exceeds the maximum limit?

A) Recursion stops
B) StackOverflowError
C) RecursionError
D) Infinite loop

✅ **Answer:** C) RecursionError

---

34. Which function can be used to set the recursion depth in Python?

A) sys.setlimit()
B) recursion.limit()
C) sys.setrecursionlimit()
D) recursiondepth()

✅ **Answer:** C) sys.setrecursionlimit()

---

35. What happens first in a recursive stack?

A) Base case is reached
B) Function starts returning
C) Function calls itself
D) Stack becomes empty

✅ **Answer:** C) Function calls itself

---

36. How many stack frames are created for this code?

```
def fun(n):
    if n == 0:
        return
    fun(n-1)
fun(3)
```

A) 1
B) 2
C) 3
D) 4

✅ **Answer:** D) 4

---

37. What is the output of the following code?

```
def fun(n):
    if n == 0:
        return
    print(n)
    fun(n-1)
print(fun(3))
```

A) 3 2 1
B) 1 2 3
C) 3 2 1 None
D) None

✅ **Answer:** C) 3 2 1 None

---

38. Which case stops the recursion process?

A) Base Case
B) Recursive Case
C) Infinite Case
D) Loop Case

✅ **Answer:** A) Base Case

---

39. How does the stack diagram look when a function returns?

A) Frames are added
B) Frames are removed
C) Stack remains the same
D) Stack becomes empty

✅ **Answer:** B) Frames are removed

---

40. What happens to the stack when recursion ends?

A) Stack becomes empty
B) Stack remains full
C) Stack holds one frame
D) Stack doubles

✅ **Answer:** A) Stack becomes empty

---

41. What is the main purpose of the stack diagram?

A) To optimize recursion
B) To visualize the function calls
C) To speed up recursion
D) To store loops

✅ **Answer:** B) To visualize the function calls

---

42. What is the return value of this recursive function?

```
def fun(n):
    if n == 0:
        return 1
    return n + fun(n-1)
print(fun(3))
```

A) 3
B) 6
C) 5
D) 4

✅ **Answer:** B) 6

---

43. What will happen if a recursive function does not have a return statement?

A) It will return None
B) It will print nothing
C) It will cause an error
D) It will run indefinitely

✅ **Answer:** A) It will return None

---

44. What will happen if recursion depth is not controlled?

A) RecursionError
B) Infinite Loop
C) StackOverflowError
D) Program Crash

✅ **Answer:** A) RecursionError

---

45. What is the last frame in the stack diagram?

A) Base Case
B) First Function Call
C) Last Function Call
D) Initial Case

✅ **Answer:** A) Base Case

---

46. What type of recursion is shown when each recursive call happens at the end of the function?

A) Head Recursion
B) Tail Recursion
C) Indirect Recursion
D) Infinite Recursion

✅ **Answer:** B) Tail Recursion

---

47. What will be the stack height for this code?

```
def fun(n):
    if n == 0:
        return
    fun(n-1)
fun(5)
```

A) 5
B) 6
C) 4
D) 0

✅ **Answer:** B) 6

---

48. Which method is not used to visualize recursion?

A) Stack Diagram
B) Tree Diagram
C) Flowchart
D) Pie Chart

✅ **Answer:** D) Pie Chart

---

49. How does the system know which function call to return to next?

A) Base Case
B) Stack Order
C) Loop
D) Random Call

✅ **Answer:** B) Stack Order

50. What is the advantage of stack diagrams?

A) Easy to debug recursive functions
B) Speed up execution
C) Reduce memory usage
D) Prevent infinite loops

✅ **Answer:** A) Easy to debug recursive functions

51. What is multiple assignment in Python?

A) Assigning multiple values to a single variable
B) Assigning a single value to multiple variables
C) Assigning multiple values to multiple variables in one line
D) Assigning a list to a variable

✅ **Answer:** C) Assigning multiple values to multiple variables in one line

52. Which of the following is the correct example of multiple assignment in Python?

A) `x, y = 5, 10`
B) `x = y = 10`
C) `x = 5, 10`
D) `x = y, z`

✅ **Answer:** A) `x, y = 5, 10`

53. What will be the output of the following code?

```
x, y, z = 1, 2, 3
print(y)
```

A) 1
B) 2
C) 3
D) Error

✅ **Answer:** B) 2

54. What happens if the number of variables does not match the number of values in multiple assignment?

A) Error
B) First variable gets all values
C) Last variable gets all values
D) Python ignores extra values

✅ **Answer:** A) Error

55. What is the output of the following code?

```
x = y = z = 100
print(x, y, z)
```

A) 100 100 100
B) 100 0 0
C) 100 100 0
D) Error

✅ **Answer:** A) 100 100 100

56. Which type of assignment is used here?

```
x, y = 5, 5
```

A) Single Assignment
B) Multiple Assignment
C) Chained Assignment
D) Tuple Assignment

✅ **Answer:** B) Multiple Assignment

57. What is the output of this code?

```
a, b = 10, 20
```

```
a, b = b, a
print(a, b)
```

A) 10 20
B) 20 10
C) 10 10
D) Error

✅ **Answer:** B) 20 10

---

58. What will the following code produce?

```
x, y = (5, 10)
print(x + y)
```

A) 15
B) 510
C) 10
D) Error

✅ **Answer:** A) 15

---

59. Which of the following is chained assignment?

A) `x, y = 10, 20`
B) `x = y = z = 50`
C) `x, y, z = 5, 5, 5`
D) `x = 10, y = 20`

✅ **Answer:** B) `x = y = z = 50`

---

60. What is the output of this code?

```
python
CopyEdit
a, b, c = 2, 4, 6
print(a + c)
```

A) 10
B) 12

C) 8
D) Error

✅ **Answer:** C) 8

---

61. How many values can be assigned in multiple assignment?

A) Only 2
B) Only 3
C) Unlimited
D) Only 1

✅ **Answer:** C) Unlimited

---

62. What is the output of the following code?

```
x, y = 5, 10
x, y = y, x
print(x)
```

A) 5
B) 10
C) Error
D) None

✅ **Answer:** B) 10

---

63. Which of the following is correct for swapping two variables using multiple assignment?

A) `x, y = y, x`
B) `x = y, y = x`
C) `x = y`
D) `swap(x, y)`

✅ **Answer:** A) `x, y = y, x`

---

64. What happens if you assign fewer values to more variables?

A) Error
B) Remaining variables are set to `None`
C) Remaining variables are set to `0`
D) Python ignores extra variables

✅ **Answer:** A) Error

---

65. What is the output of this code?

```
x, y, z = 1, 2, 3
print(z)
```

A) 1
B) 2
C) 3
D) Error

✅ **Answer:** C) 3

---

66. Can multiple assignment be used with lists?

A) Yes
B) No
C) Only with tuples
D) Only with integers

✅ **Answer:** A) Yes

---

67. What is the output of this code?

```
a, b, c = [10, 20, 30]
print(a, c)
```

A) 10 20
B) 10 30
C) 20 30
D) Error

✅ **Answer:** B) 10 30

---

68. Which operator is used for multiple assignment in Python?

A) `=`
B) `==`
C) `,`
D) `+=`

✅ **Answer:** A) `=`

---

69. What happens if more values are assigned to fewer variables?

A) Error
B) Remaining values are ignored
C) Remaining values are set to `None`
D) Python splits the values equally

✅ **Answer:** A) Error

---

70. What will the following code output?

```
a, b = 5, 6
a, b = b, a
print(a, b)
```

A) 5 6
B) 6 5
C) Error
D) None

✅ **Answer:** B) 6 5

---

71. Which data type is mostly used in multiple assignment?

A) List
B) Tuple

C) Dictionary
D) All of the above

✅ **Answer:** D) All of the above

---

72. What is the output of this code?

```
x, y = 10, 20
x, y = y, x + y
print(x, y)
```

A) 10 20
B) 20 30
C) 10 30
D) 20 10

✅ **Answer:** B) 20 30

---

73. What is the output of the following code?

```
a, b = 5, 6
a, b, c = a + b, a, b
print(a, b, c)
```

A) 11 5 6
B) 5 6 11
C) Error
D) 6 5 11

✅ **Answer:** A) 11 5 6

---

74. Can multiple assignment be used with strings?

A) Yes
B) No
C) Only with integers
D) Only with lists

✅ **Answer:** A) Yes

---

75. What is the output of this code?

```
x, y, z = "ABC"
print(x, z)
```

A) A C
B) B C
C) A B
D) Error

✅ **Answer:** A) A C

---

76. What will happen if we assign the same value to two variables?

A) Both variables point to the same memory
B) Error
C) Different memory locations
D) None

✅ **Answer:** A) Both variables point to the same memory

---

77. Which of the following supports multiple assignment?

A) List
B) Tuple
C) String
D) All of the above

✅ **Answer:** D) All of the above

---

78. What will be the output?

```
a, b, c = "abc"
print(b)
```

A) a
B) b

C) c
D) Error

✅ **Answer:** B) b

---

79. Can multiple assignment be used for unpacking lists?

A) Yes
B) No
C) Only with integers
D) Only with tuples

✅ **Answer:** A) Yes

---

80. What is the output?

```
x, y, z = range(3)
print(x, y, z)
```

A) 0 1 2
B) 1 2 3
C) Error
D) 0 2 1

✅ **Answer:** A) 0 1 2

81. What is the primary purpose of the while statement in Python?

- a) To execute a block of code a fixed number of times
- b) To execute a block of code as long as a condition is true ✅
- c) To iterate over a list
- d) To define a function

---

82. What will be the output of the following code?

```
x = 5
while x > 0:
    print(x)
    x -= 1
```

- a) 5 4 3 2 1 ✅
- b) 5 4 3 2
- c) 4 3 2 1
- d) Infinite loop

---

83. What happens if the condition in the while loop is False at the start?

- a) The loop executes once
- b) The loop executes infinitely
- c) The loop does not execute ✅
- d) The loop throws an error

---

84. How can you exit a while loop before the condition becomes False?

- a) Using the **continue** statement
- b) Using the **break** statement ✅
- c) Using the **return** statement
- d) Using the **exit()** function

---

85. What is the output of the following code?

```
i = 1
while i < 6:
    if i == 3:
        break
    print(i)
    i += 1
```

- a) 1 2 ✅
- b) 1 2 3
- c) 1 2 3 4 5
- d) 1 2 4 5

---

86. What is the role of the else clause in a while loop?

- a) Executes when the loop is terminated by **break**
- b) Executes when the loop condition is **False** ✅
- c) Executes only when the loop starts

- d) It is not allowed in a **while** loop

---

87. What will happen if the condition in the while loop always evaluates to True?

- a) The loop will run infinitely ✅
- b) The loop will run once
- c) The loop will throw an error
- d) The loop will not execute

---

88. What is the correct syntax for a while loop in Python?

- a) `while condition:` ✅
- b) `while (condition)`
- c) `while: condition`
- d) `while {condition}`

---

89. What will be the output of the following code?

```
x = 0
while x < 3:
    print(x)
    x += 1
else:
    print("Done")
```

- a) 0 1 2
- b) 0 1 2 Done ✅
- c) 0 1 2 3 Done
- d) Done

---

90. Which statement skips the current iteration and moves to the next iteration in a while loop?

- a) break
- b) continue ✅
- c) pass
- d) return

91. What is the initial value of the loop control variable in the following code?

```
i = 0
while i < 5:
    print(i)
    i += 1
```

- a) 5
- b) 0 ✅
- c) 1
- d) None

92. How many times will the following loop execute?

```
x = 10
while x > 0:
    print(x)
    x -= 2
```

- a) 5 ✅
- b) 10
- c) Infinite
- d) 2

93. What will be the output of the following code?

```
x = 3
while x > 0:
    x -= 1
    if x == 1:
        continue
    print(x)
```

- a) 3 2
- b) 2 0 ✅
- c) 3 2 1
- d) 2 1 0

94. Which keyword is used to stop the execution of a while loop?

- a) return
- b) continue

- c) break ✅
- d) stop

---

95. What is the output of this code?

```
i = 1
while i <= 5:
    print(i, end=" ")
    i += 1
else:
    print("End")
```

- a) 1 2 3 4 5
- b) 1 2 3 4 5 End ✅
- c) End 1 2 3 4 5
- d) 1 2 3 4

---

96. Which of the following will NOT cause a while loop to stop?

- a) The condition becomes **False**
- b) The **break** statement is used
- c) The **continue** statement is used ✅
- d) The loop finishes its iterations

---

97. What will be the output of this code?

```
count = 0
while count < 3:
    print("Hello")
    count += 1
else:
    print("Bye")
```

- a) Hello Hello Hello
- b) Hello Hello Hello Bye ✅
- c) Bye Hello Hello
- d) Infinite loop

---

98. Which statement is used to do nothing inside a while loop?

- a) break
- b) continue
- c) pass ✅
- d) skip

---

99. What will happen if there is no increment or decrement in the loop body?

```
x = 5
while x > 0:
    print(x)
```

- a) The loop will execute 5 times
- b) The loop will execute once
- c) Infinite loop ✅
- d) Error

---

100. Which loop structure is best used when the number of iterations is unknown?

- a) for loop
- b) while loop ✅
- c) do-while loop
- d) switch loop

Part 1: Two-Dimensional Tables in Python

---

101. How is a two-dimensional list represented in Python?

- a) List of tuples
- b) List of lists ✅
- c) Tuple of lists
- d) Dictionary

---

102. What is the output of the following code?

```
matrix = [[1, 2], [3, 4]]
print(matrix[1][0])
```

- a) 1
- b) 2
- c) 3 ✅
- d) 4

---

103. How can you initialize a 3x3 matrix with zeros?

- a) `matrix = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]` ✅
- b) `matrix = [0, 0, 0]`
- c) `matrix = [[0]*3]*3`
- d) `matrix = [[0]*3 for i in range(3)]` ✅

---

104. How can you access the second row, third column of the matrix?

```
matrix = [[5, 6, 7], [8, 9, 10], [11, 12, 13]]
```

- a) `matrix[1][2]` ✅
- b) `matrix[2][1]`
- c) `matrix[2][3]`
- d) `matrix[1][3]`

---

105. What is the result of this code?

```
matrix = [[2, 4], [6, 8]]
print(matrix[0][-1])
```

- a) 2
- b) 4 ✅
- c) 6
- d) 8

---

106. How can you append a row to a 2D list?

- a) `matrix.insert([4, 5, 6])`
- b) `matrix.append([4, 5, 6])` ✅
- c) `matrix.add([4, 5, 6])`
- d) `matrix.extend([4, 5, 6])`

107. What is the default value of elements when creating a 2D list using:

```
matrix = [[0]*3 for i in range(3)]
```

- a) 0 ✅
- b) None
- c) 1
- d) -1

Part 2: Strings in Python

108. What is the output of the following code?

```
string = "Python"
print(string[1:4])
```

- a) Pyt
- b) yth ✅
- c) ytho
- d) hon

109. Strings in Python are:

- a) Mutable
- b) Immutable ✅
- c) Dynamic
- d) Constant

110. What is the result of this code?

```
text = "Python"
print(text.upper())
```

- a) python
- b) PYTHON ✅

- c) Python
- d) pYTHON

111. Which method is used to replace a substring in Python?

- a) `replace()` ✅
- b) `update()`
- c) `change()`
- d) `swap()`

112. What will the following code output?

```
string = "hello world"
print(string.split())
```

- a) ['hello', 'world'] ✅
- b) ['helloworld']
- c) ['hello', ' ', 'world']
- d) ['h', 'e', 'l', 'l', 'o', 'w', 'o', 'r', 'l', 'd']

113. What does the strip() method do in Python?

- a) Removes spaces from the start only
- b) Removes spaces from the end only
- c) Removes spaces from both start and end ✅
- d) Converts string to lowercase

114. What will be the output?

```
s = "Python"
print(s[::-1])
```

- a) nohtyP ✅
- b) Python
- c) Pytho
- d) Error

115. What is the correct syntax to concatenate two strings in Python?

- a) `string1.join(string2)`
- b) `string1 + string2` ✅
- c) `string1.append(string2)`
- d) `string1.extend(string2)`

Part 3: Lists in Python

116. How can you create a list in Python?

- a) `{}`
- b) `[]` ✅
- c) `()`
- d) `<>`

117. What is the output of this code?

```
lst = [1, 2, 3]
lst.append(4)
print(lst)
```

- a) [1, 2, 3, 4] ✅
- b) [4, 1, 2, 3]
- c) [1, 2, 3]
- d) [1, 2, 3, [4]]

118. How can you remove the last element from a list?

- a) `list.delete()`
- b) `list.remove()`
- c) `list.pop()` ✅
- d) `list.cut()`

119. What will be the result of this code?

```
lst = [1, 2, 3]
lst.insert(1, 100)
print(lst)
```

- a) [100, 1, 2, 3]
- b) [1, 100, 2, 3] ✅
- c) [1, 2, 100, 3]
- d) [1, 2, 3, 100]

120. How do you sort a list in ascending order?

- a) `list.sort()` ✅
- b) `list.sorted()`
- c) `list.reverse()`
- d) `list.order()`

Part 4: Advanced Concepts

121. What is the output of this code?

```
lst = [1, 2, [3, 4], 5]
print(lst[2][1])
```

- a) 3
- b) 4 ✅
- c) [3, 4]
- d) Error

122. Which method is used to copy a list?

- a) `copy()` ✅
- b) `clone()`
- c) `duplicate()`
- d) `append()`

123. How can you reverse a list in Python?

- a) `list.reverse()` ✅
- b) `list[::-1]`
- c) `reversed(list)`
- d) All of the above ✅

---

124. What will this code produce?

```
lst = [1, 2, 3]
print(len(lst))
```

- a) 2
- b) 3 ✅
- c) 4
- d) Error

---

125. Which method is used to remove elements from a list by index?

- a) `pop()` ✅
- b) `remove()`
- c) `delete()`
- d) `discard()`

---

126. How do you create a deep copy of a list?

- a) `list.copy()`
- b) `copy.deepcopy(list)` ✅
- c) `list.clone()`
- d) `deepcopy(list)`

---

127. What is the default sorting order of `sort()`?

- a) Descending
- b) Ascending ✅
- c) Random
- d) None

128. How can you join list elements into a string?

- a) `join()` ✅
- b) `concat()`
- c) `merge()`
- d) `append()`

129. What is the output of this code?

```
lst = ['a', 'b', 'c']
print(lst.index('b'))
```

- a) 0
- b) 1 ✅
- c) 2
- d) Error

130. What is the result of `max([4, 8, 2])`?

- a) 4
- b) 8 ✅
- c) 2
- d) None

131. What is a string in Python?

- a) Mutable data type
- b) Immutable data type ✅
- c) Numeric data type
- d) List

132. What does the `len()` function return when applied to a string?

```
s = "Hello World"
print(len(s))
```

- a) 10

- b) 11 ✅
- c) 12
- d) 9

---

133. How are strings stored in Python?

- a) List of integers
- b) List of characters ✅
- c) Dictionary
- d) Tuples

---

134. What is the output of the following code?

```
s = "Python"
print(s[3])
```

- a) t
- b) h ✅
- c) y
- d) n

---

135. What will happen if you try to modify a string in Python?

- a) It will modify successfully
- b) It will throw an error ✅
- c) It will add the new value
- d) It will replace the old value

---

136. What is the output of this code?

```
s = "Python"
print(s[-1])
```

- a) P
- b) n ✅
- c) o
- d) y

---

137. How can you access the first character of a string `s`?

- a) `s[1]`
- b) `s[0]` ✅
- c) `s[-1]`
- d) `s[2]`

---

138. What is the output of the following code?

```
s = "Python"
print(s[::2])
```

- a) Pto ✅
- b) yhn
- c) Pyth
- d) Ptn

---

139. What will be the output of this code?

```
s = "Hello World"
print(s[:5])
```

- a) Hello ✅
- b) World
- c) Hello World
- d) ello

---

140. How can you reverse a string `s` in Python?

- a) `s.reverse()`
- b) `s[::-1]` ✅
- c) `reverse(s)`
- d) `s.flip()`

---

141. What will be the output of this code?

```
s = "Python"
print(s[1:4])
```

- a) yth ✅
- b) Pyt
- c) tho
- d) thon

---

142. Which method is used to count the occurrences of a substring in a string?

- a) `len()`
- b) `count()` ✅
- c) `find()`
- d) `replace()`

---

143. How can you traverse a string in Python?

- a) Using a for loop ✅
- b) Using a while loop ✅
- c) Using if-else
- d) Both a and b ✅

---

144. What is the output of the following code?

```
s = "Python"
for i in s:
    print(i, end=" ")
```

- a) P y t h o n ✅
- b) Python
- c) n o h t y P
- d) Error

---

145. What is the result of this code?

```
s = "Hello"
print(s[1:4:2])
```

- a) el
- b) eo
- c) Hl
- d) el ✅

---

146. Which method converts the string into uppercase letters?

- a) `upper()` ✅
- b) `lower()`
- c) `capitalize()`
- d) `title()`

---

147. What will be the output of this code?

```
s = "Python Programming"
print(s[7:18])
```

- a) Programming ✅
- b) Python
- c) rammi
- d) on Pro

---

148. What is the correct method to remove whitespace from both ends of a string?

- a) `strip()` ✅
- b) `split()`
- c) `remove()`
- d) `clean()`

---

149. What will `len("Welcome")` return?

- a) 6
- b) 7 ✅
- c) 8
- d) 5

150. How do you concatenate two strings `s1` and `s2`?

- a) `s1.append(s2)`
- b) `s1.join(s2)`
- c) `s1 + s2` ✅
- d) `s1.merge(s2)`

151. What is the output of the following code?

```
s = "Python"
print(len(s[1:4]))
```

- a) 2
- b) 3 ✅
- c) 4
- d) 6

152. Which method returns the index of the first occurrence of a substring?

- a) `index()` ✅
- b) `find()`
- c) `search()`
- d) `replace()`

153. What is the output of this code?

```
s = "Python"
print(s.upper())
```

- a) python
- b) PYTHON ✅
- c) Python
- d) pYTHON

154. How can you iterate through each character in a string?

- a) Using a `for` loop ✅
- b) Using a `while` loop ✅
- c) Both a and b ✅
- d) None

---

155. What is the result of this code?

```
s = "python"
print(s.capitalize())
```

- a) Python ✅
- b) PYTHON
- c) python
- d) pYTHON

---

156. What will this code print?

```
s = "hello world"
print(s.title())
```

- a) Hello World ✅
- b) Hello world
- c) HELLO WORLD
- d) hello world

---

157. Which method is used to check if a string starts with a specific substring?

- a) `startswith()` ✅
- b) `endswith()`
- c) `isalpha()`
- d) `find()`

---

158. What will be the output of this code?

```
s = "Hello World"
print(s.split())
```

- a) ['Hello', 'World'] ✅

- b) ['H', 'e', 'l', 'l', 'o', ' ', 'W', 'o', 'r', 'l', 'd']
- c) Hello World
- d) ['Hello World']

---

159. Which method joins a list of strings into one string?

- a) `join()` ✅
- b) `concat()`
- c) `merge()`
- d) `combine()`

---

160. What will be the output of this code?

```
s = "Python Programming"
print(s.replace("Python", "Java"))
```

- a) Python Programming
- b) Java Programming ✅
- c) Programming Java
- d) Error

161. What is the primary purpose of the `for` loop in Python?

- a) Infinite loop
- b) Conditional execution
- c) Iterating over sequences ✅
- d) Defining functions

---

162. Which keyword is used to start a `for` loop in Python?

- a) while
- b) loop
- c) for ✅
- d) if

---

163. What will be the output of the following code?

```
for i in range(3):
    print(i)
```

- a) 0 1 2 ✓
- b) 1 2 3
- c) 0 1 2 3
- d) 1 2

---

164. What is the default starting value of `range()` in Python?

- a) 0 ✓
- b) 1
- c) -1
- d) None

---

165. How many times will this loop execute?

```
for i in range(5):
    print("Hello")
```

- a) 4
- b) 5 ✓
- c) Infinite
- d) 0

---

166. What is the correct syntax for iterating through a list using a `for` loop?

- a) `for i in range(list):`
- b) `for i in list:` ✓
- c) `for i = list:`
- d) `for i from list:`

---

167. What is the output of the following code?

```
for i in "Python":
    print(i, end=" ")
```

- a) Python

- b) P y t h o n ✅
- c) Pyt hon
- d) Error

---

168. What will the following code print?

```
for i in range(1, 6, 2):
    print(i)
```

- a) 1 2 3 4 5
- b) 1 3 5 ✅
- c) 2 4 6
- d) 1 4 6

---

169. What is the output of this code?

```
numbers = [10, 20, 30]
for num in numbers:
    print(num)
```

- a) 10 20 30 ✅
- b) [10, 20, 30]
- c) num num num
- d) Error

---

170. Which function is commonly used with `for` loops to generate numbers?

- a) `generate()`
- b) `range()` ✅
- c) `count()`
- d) `numbers()`

---

171. What will be the output of this code?

```
for i in range(4, 0, -1):
    print(i)
```

- a) 4 3 2 1 ✅

- b) 0 1 2 3 4
- c) 4 0 -1 -2
- d) Error

---

172. What is the output of this loop?

```
for i in range(2, 10, 3):
    print(i)
```

- a) 2 5 8 ✅
- b) 2 4 6 8
- c) 3 6 9
- d) 2 3 4

---

173. What will the following code print?

```
for i in range(5):
    if i == 3:
        break
    print(i)
```

- a) 0 1 2 ✅
- b) 0 1 2 3
- c) 0 1 2 3 4
- d) 1 2 3

---

174. Which statement is used to skip the current iteration in a `for` loop?

- a) break
- b) continue ✅
- c) pass
- d) return

---

175. What will this code print?

```
for i in range(5):
    if i == 3:
        continue
    print(i)
```

- a) 0 1 2 3 4
- b) 0 1 2 4 ✅
- c) 0 1 3 4
- d) 1 2 3 4

---

176. What does the `else` block in a `for` loop do?

- a) Executes when the loop condition is false
- b) Executes when the loop completes normally ✅
- c) Executes only if the loop is infinite
- d) Executes before the loop starts

---

177. What will be the output of this code?

```
for i in range(3):
    print(i)
else:
    print("Done")
```

- a) 0 1 2 Done ✅
- b) 0 1 2
- c) Done 0 1 2
- d) Error

---

178. How can you iterate through both keys and values of a dictionary?

- a) `for k, v in dict:`
- b) `for k, v in dict.items():` ✅
- c) `for k, v in dict.values():`
- d) `for k in dict.keys():`

---

179. What will be the output of the following code?

```
for i in range(3):
    pass
print("End")
```

- a) 0 1 2

- b) End ✅
- c) Error
- d) None

---

180. What will happen if the `range()` function is given only one argument?

- a) It starts from 1
- b) It starts from 0 ✅
- c) It starts from the given number
- d) It gives an error

181. What is string slicing in Python?

- a) Breaking a string into two parts
- b) Extracting a portion of the string ✅
- c) Adding two strings
- d) Comparing two strings

---

182. What is the output of the following code?

```
s = "Python Programming"
print(s[0:6])
```

- a) Python ✅
- b) Pytho
- c) Pyth
- d) Python Programming

---

183. How do you extract the last 4 characters from the string `"Programming"`?

- a) `s[-4:]` ✅
- b) `s[4:-1]`
- c) `s[4:8]`
- d) `s[0:4]`

---

184. What will be the output of this code?

```
s = "Python"
print(s[:4])
```

- a) Pyth ✅
- b) on
- c) Pyt
- d) ytho

---

185. What is the result of the following code?

```
s = "Python"
print(s[2:5])
```

- a) yth
- b) tho ✅
- c) ytho
- d) on

---

186. How can you compare two strings `s1` and `s2` in Python?

- a) `s1 == s2` ✅
- b) `s1.compare(s2)`
- c) `s1.equal(s2)`
- d) `s1 = s2`

---

187. What is the result of this code?

```
s1 = "Hello"
s2 = "hello"
print(s1 == s2)
```

- a) True
- b) False ✅
- c) Error
- d) None

---

188. Which method is used to compare two strings ignoring case sensitivity?

- a) `lower()` ✅
- b) `find()`
- c) `slice()`
- d) `title()`

---

189. What is the correct way to find the index of the first occurrence of `"o"` in `"Hello World"`?

- a) `s.index("o")`
- b) `s.find("o")` ✅
- c) `s.locate("o")`
- d) `s.search("o")`

---

190. What is the output of this code?

```
s = "Python Programming"
print(s.find("g"))
```

- a) 10 ✅
- b) 11
- c) 12
- d) 13

---

191. What will the `find()` method return if the substring is not found?

- a) 0
- b) -1 ✅
- c) 1
- d) None

---

192. How can you search for the substring `"World"` in `"Hello World"`?

- a) `s.index("World")`
- b) `s.find("World")` ✅
- c) `s.locate("World")`
- d) `s.search("World")`

---

193. What is the output of this code?

```
s = "Hello World"
print(s.find("l", 4))
```

- a) 2
- b) 3
- c) 9 ✅
- d) 4

---

194. What is the result of this code?

```
s = "Programming"
print(s[3:7])
```

- a) gramm
- b) gram ✅
- c) gramming
- d) ming

---

195. What is the output of this code?

```
s = "Python"
print(s[-3:])
```

- a) hon ✅
- b) Pyt
- c) yth
- d) tho

---

196. Which function is used to find the last occurrence of a substring?

- a) `rfind()` ✅
- b) `find()`
- c) `index()`
- d) `count()`

---

197. What will be the output of the following code?

```
s = "Python Programming"
print(s.find("z"))
```

- a) -1 ✅
- b) 0
- c) 1
- d) Error

---

198. What is the result of this code?

```
s = "Hello World"
print(s.find("o", 5))
```

- a) 4
- b) 7 ✅
- c) 5
- d) 6

---

199. Which method is used to search for the first index of a substring in a string?

- a) `index()` ✅
- b) `find()`
- c) `locate()`
- d) `match()`

---

200. What will be the output of this code?

```
s = "Hello World"
print(s.find("World"))
```

- a) -1
- b) 6 ✅
- c) 5
- d) 7

Topic: Looping and Counting

1. What is the output of the following code?

```
count = 0
while count < 3:
    print(count)
    count += 1
```

A) 0 1 2
B) 1 2 3
C) 0 1 2 3
D) Infinite loop
Answer: A

2. What does the `range(5)` produce?

A) [0, 1, 2, 3, 4, 5]
B) [1, 2, 3, 4, 5]
C) [0, 1, 2, 3, 4]
D) [1, 2, 3, 4]
Answer: C

3. Which loop is guaranteed to execute at least once?

A) for loop
B) while loop
C) do-while loop
D) None of the above
Answer: C (Note: Python doesn't have native do-while, but conceptually correct)

4. What keyword is used to skip the current iteration in a loop?

A) skip
B) stop
C) continue
D) pass
Answer: C

5. What is the output?

```
for i in range(3):
    print(i, end=" ")
```

A) 0 1 2
B) 1 2 3

C) 0 1 2 3
D) Error
Answer: A

6. Which loop is used when the number of iterations is not known in advance?

A) for
B) while
C) do-while
D) All of the above
Answer: B

7. What is the output?

```
i = 1
while i < 5:
    i += 1
print(i)
```

A) 1
B) 4
C) 5
D) Error
Answer: C

8. How many times will the loop run?

```
for i in range(2, 10, 2):
    print(i)
```

A) 4
B) 5
C) 6
D) 8
Answer: B

9. What is the default start value of `range(n)`?

A) 0
B) 1
C) n
D) None
Answer: A

10. What is the output?

```
for i in range(1, 5):
    print(i * "*")
```

A) * * * *
B) *
**

C) ****
D) Error
Answer: B

11. What does `break` do in a loop?

A) Ends the loop
B) Skips one iteration
C) Restarts the loop
D) Causes an error
Answer: A

12. What will this print?

```
for i in range(3):
    if i == 1:
        break
    print(i)
```

A) 0
B) 0 1
C) 1
D) None
Answer: A

13. Which is the correct syntax of a for loop?

A) `for(i=0; i<n; i++)`
B) `for i in range(n):`
C) `loop i to n:`
D) `repeat i until n:`
Answer: B

14. What happens when the condition in a while loop is false?

A) Loop repeats
B) Loop ends
C) Error
D) Skips one iteration
Answer: B

15. What is the output?

```
x = 10
while x > 0:
    x -= 3
print(x)
```

A) 0
B) -2
C) -1
D) 1
Answer: B

16. Which of the following is a counting variable?

A) `for`
B) `i`
C) `if`
D) `def`
Answer: B

17. Which built-in function is often used for counting iterations in a loop?

A) count()
B) len()
C) range()
D) enumerate()
Answer: C

18. What is the output?

```
for i in range(3):
    for j in range(2):
        print(i, j)
```

A) (0,0) (1,1)
B) All combinations of i and j
C) Error
D) (0,1) (1,2)
Answer: B

19. Which of the following is an infinite loop?

A) `while True:`
B) `for i in range(1, 10):`
C) `while i == 5:`
D) `for i in range(0):`
Answer: A

20. How to iterate through a string character by character?

A) `for ch in string:`
B) `for string in ch:`
C) `loop ch string:`
D) `foreach ch in string:`
Answer: A

21. What is the output?

```
for i in range(5):
    if i == 3:
        continue
    print(i)
```

A) 0 1 2 3 4
B) 0 1 2 4
C) 1 2 3 4
D) Error
Answer: B

22. Which function can be used with a loop to get both index and value?

A) zip()
B) enumerate()
C) list()
D) map()
Answer: B

23. What is the output?

```
for i in range(3, 0, -1):
    print(i)
```

A) 1 2 3
B) 3 2 1
C) 3 2
D) Error
Answer: B

24. What happens if you forget to update the loop counter in a while loop?

A) Syntax error
B) Infinite loop
C) Loop executes once
D) Nothing
Answer: B

25. What does `pass` do in a loop?

A) Skips current iteration
B) Exits loop
C) Does nothing
D) Causes error
Answer: C

26. What is the result?

```
count = 0
for i in range(5):
    count += i
print(count)
```

A) 5
B) 10
C) 15
D) 20
Answer: B

27. Which loop can iterate over lists directly?

A) for
B) while
C) do-while
D) None
Answer: A

28. What is the output?

```
for i in range(1, 6):
    if i % 2 == 0:
        print(i)
```

A) 1 3 5
B) 2 4
C) 1 2 3 4 5
D) Error
Answer: B

29. How to make a loop run in reverse?

A) `for i in reverse(5)`
B) `for i in range(5, 0)`
C) `for i in range(5, 0, -1)`

D) `for i = 5 to 1`

Answer: C

30. What is printed?

```
i = 0
while i < 3:
    print("Loop", i)
    i += 1
```

A) Loop 0 Loop 1 Loop 2
B) Loop 1 Loop 2 Loop 3
C) Error
D) Nothing

Answer: A

Topics: List values, accessing elements, list length, list membership, lists and for loops, list operations, list deletion. Cloning lists, nested lists

1. Which of the following is a valid list?

A) `[1, 2, 3]`
B) `(1, 2, 3)`
C) `{1, 2, 3}`
D) `list(1, 2, 3)`
Answer: A

2. What is the output of `len([1, 2, 3, 4])`?

A) 3
B) 4
C) 5
D) Error
Answer: B

3. How do you access the last element of a list `lst`?

A) `lst[last]`
B) `lst[-1]`
C) `lst[len(lst)]`
D) `lst(1)`
Answer: B

4. What is the output of:

```
x = [1, 2, 3]
print(2 in x)
```

A) True
B) False
C) 2
D) Error
Answer: A

5. What will `list(range(3))` return?

A) `[0, 1, 2]`
B) `[1, 2, 3]`

C) `[0, 1, 2, 3]`
D) `(0, 1, 2)`
Answer: A

6. Which operator is used to concatenate two lists?

A) `+`
B) `*`
C) `%`
D) `&`
Answer: A

7. What does `lst.append(4)` do?

A) Adds 4 at beginning
B) Adds 4 at end
C) Adds 4 at index 1
D) Nothing
Answer: B

8. What is the output?

```
lst = [1, 2, 3]
lst[1] = 10
print(lst)
```

A) `[1, 2, 3]`
B) `[10, 2, 3]`
C) `[1, 10, 3]`
D) Error
Answer: C

9. What does `del lst[2]` do?

A) Removes the last item
B) Deletes the list
C) Removes the third element
D) Clears the list
Answer: C

10. Which method removes the first occurrence of a value?

A) `remove()`
B) `pop()`
C) `delete()`

D) `discard()`

Answer: A

11. What is the output?

```
lst = [1, 2, 3]
print(lst * 2)
```

A) `[1, 2, 3, 1, 2, 3]`
B) `[2, 4, 6]`
C) `[1, 4, 9]`
D) Error

Answer: A

12. Which method returns the index of an item?

A) `find()`
B) `locate()`
C) `index()`
D) `search()`

Answer: C

13. How do you clone a list `lst`?

A) `lst.copy()`
B) `list(lst)`
C) `lst[:]`
D) All of the above

Answer: D

14. What is the output?

```
a = [1, 2, 3]
b = a
b.append(4)
print(a)
```

A) `[1, 2, 3]`
B) `[1, 2, 3, 4]`
C) Error
D) None

Answer: B

15. How do you create a nested list?

A) `[[1, 2], [3, 4]]`
B) `[(1, 2), (3, 4)]`

C) `{{1, 2}, {3, 4}}`
D) All of the above
Answer: A

16. What is the output?

```
nested = [[1, 2], [3, 4]]
print(nested[1][0])
```

A) 1
B) 2
C) 3
D) 4
Answer: C

17. What is the result of `lst.pop(0)`?

A) Removes last item
B) Removes first item
C) Removes all
D) Adds element
Answer: B

18. What is the output?

```
lst = [1, 2, 3]
for i in lst:
    print(i, end=", ")
```

A) 123
B) 1, 2, 3,
C) 1 2 3
D) Error
Answer: B

19. What will `lst.clear()` do?

A) Deletes one element
B) Removes last item
C) Removes all elements
D) Gives error
Answer: C

20. What is the output?

```
lst = [1, [2, 3], 4]
print(lst[1][1])
```

A) 2
B) 3
C) [2, 3]
D) Error
Answer: B

21. `lst[::-1]` does what?

A) Sorts the list
B) Clones the list
C) Reverses the list
D) None
Answer: C

22. Which of the following creates an empty list?

A) `[]`
B) `list()`
C) Both
D) Neither
Answer: C

23. What is the result of:

```
 [1, 2, 3].remove(4)
```

A) Removes 4
B) Error
C) Removes last item
D) Returns None
Answer: B

24. What is the output of:

```
x = [1, 2, 3]
print(x[3])
```

A) 3
B) Error
C) 4
D) None
Answer: B

25. What is `len([])`?

A) 0
B) 1

C) Error
D) None
Answer: A

26. What is the output?

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

A) [1, 2, 3, 4]
B) [1, 2, 3]
C) Error
D) None
Answer: B

27. What is the output?

```
x = [10, 20, 30]
print(20 in x)
```

A) True
B) False
C) 20
D) Error
Answer: A

28. What does `lst.insert(1, 100)` do?

A) Replaces index 1 with 100
B) Inserts 100 at index 1
C) Adds 100 at end
D) Gives error
Answer: B

29. Which method adds all elements from another list?

A) `extend()`
B) `append()`
C) `add()`
D) `concat()`
Answer: A

30. How to check if a list is empty?

```
if not lst:
```

A) Correct
B) Error
C) Not efficient
D) None
Answer: A

31. What is the output?

```
lst = ['a', 'b', 'c']
print(lst[0:2])
```

A) `['a', 'b']`
B) `['a', 'b', 'c']`
C) `['b', 'c']`
D) Error
Answer: A

32. What will this code do?

```
lst = [1, 2, 3]
lst += [4]
print(lst)
```

A) `[1, 2, 3, 4]`
B) `[5]`
C) Error
D) `[1, 2, 3, [4]]`
Answer: A

33. What is the result?

```
list1 = [1, 2]
list2 = list1
list1[0] = 100
print(list2)
```

A) `[1, 2]`
B) `[100, 2]`
C) `[1, 100]`
D) Error
Answer: B

34. What is the output?

```
lst = [10, 20, 30, 40]
print(lst[-2])
```

A) 20
B) 30
C) 40
D) Error
Answer: B

35. What does this return?

```
lst = [1, 2, 3]
print(lst.index(3))
```

A) 2
B) 3
C) 1
D) Error
Answer: A

36. What is the output?

```
lst = [[1, 2], [3, 4]]
print(len(lst))
```

A) 2
B) 4
C) 1
D) Error
Answer: A

37. Which will raise an IndexError?

A) `lst[0]` where `lst = []`
B) `lst = [1]; lst[0]`
C) `lst = [1]; lst[-1]`
D) `lst = [1,2,3]; lst[2]`
Answer: A

38. How do you copy only part of a list?

A) `lst.copy(1:3)`
B) `lst.slice(1,3)`
C) `lst[1:3]`
D) `lst.sub(1,3)`
Answer: C

39. What is the result of this loop?

```
for i in [10, 20, 30]:
    print(i, end=' ')
```

A) 10 20 30
B) [10, 20, 30]
C) i i i
D) Error
Answer: A

40. What is printed?

```
lst = [1, 2, 3]
lst.remove(2)
print(lst)
```

A) `[1, 2, 3]`
B) `[1, 3]`
C) `[2, 3]`
D) Error
Answer: B

41. Which one is not a valid list operation?

A) `append()`
B) `extend()`
C) `add()`
D) `insert()`
Answer: C

42. How do you flatten `[[1,2],[3,4]]` to `[1,2,3,4]`?

A) Use `sum(nested, [])`
B) Use a loop
C) Use list comprehension
D) All of the above
Answer: D

43. What is printed?

```
lst = [4, 5, 6]
print(6 in lst)
```

A) True
B) False
C) 6
D) Error
Answer: A

44. How to remove all elements from list `a`?

A) `a.delete()`
B) `a.remove()`
C) `a.clear()`
D) `a.empty()`
Answer: C

45. What happens if `list.pop()` is used on an empty list?

A) Returns `None`
B) Returns `0`
C) Raises `IndexError`
D) Does nothing
Answer: C

46. What does this do?

```
a = [1, 2]
b = a[:]
a[0] = 99
print(b)
```

A) `[99, 2]`
B) `[1, 2]`
C) `[99]`
D) Error
Answer: B

47. Which of the following is not allowed in a list?

A) Numbers
B) Strings
C) Dictionaries
D) All allowed
Answer: D

48. Which syntax creates a list of 5 zeros?

A) `[0] * 5`
B) `[0,0,0,0,0]`
C) `list(range(5))`
D) A and B
Answer: D

49. What is the output?

```
lst = ['a', 'b', 'c']
for i in range(len(lst)):
    print(lst[i])
```

A) a b c
B) abc
C) ['a', 'b', 'c']
D) Error
Answer: A

50. What is the output of:

```
list1 = [1, 2]
list2 = list1[:]
print(list1 is list2)
```

A) True
B) False
C) Error
D) None
Answer: B

Topics: Object oriented programming: introduction to classes, objects and methods

1. What is the basic building block of Object-Oriented Programming?

A) Function
B) Variable
C) Class
D) Module
Answer: C

2. What keyword is used to define a class in Python?

A) `object`
B) `class`
C) `def`
D) `method`
Answer: B

3. What is an object in Python?

A) A collection of functions
B) An instance of a class
C) A Python file
D) A module
Answer: B

4. What is the correct way to create an object from a class?

```
class Student:
    pass
```

A) `Student.create()`
B) `student = new Student()`
C) `student = Student()`
D) `create Student()`
Answer: C

5. Which method is automatically called when an object is created?

A) `__create__()`
B) `__object__()`
C) `__init__()`
D) `__new__()`
Answer: C

6. What is the output?

```
class A:
    def __init__(self):
        print("A created")
obj = A()
```

A) Nothing
B) Error
C) A created
D) Class A
Answer: C

7. Which of these defines an instance method?

A) `def method:`
B) `def method(self):`
C) `def method():`
D) `def method(obj):`
Answer: B

8. What does `self` refer to in a method?

A) The class
B) A function
C) The module
D) The object calling the method
Answer: D

9. Which is NOT a feature of OOP?

A) Inheritance
B) Encapsulation
C) Compilation
D) Polymorphism
Answer: C

10. How do you define a constructor in Python?

A) `def constructor(self):`
B) `def init(self):`
C) `def __init__(self):`
D) `constructor()`
Answer: C

11. What is method overloading in Python?

A) Using more than one method name
B) Defining methods with the same name but different parameters

C) Calling too many methods
D) It's not supported
Answer: D (Python does not support method overloading directly)

12. What is encapsulation?

A) Hiding the main logic
B) Protecting object data
C) Using many classes
D) Reusing code
Answer: B

13. Which keyword is used to inherit a class?

A) `extends`
B) `inherits`
C) `super`
D) None (inheritance is defined via parentheses)
Answer: D

14. Which is the parent class in:

```
class B(A):
    pass
```

A) B
B) A
C) self
D) None
Answer: B

15. What is a class variable?

A) Defined inside `__init__`
B) Specific to each object
C) Shared by all instances
D) Not allowed
Answer: C

16. How do you access a class variable?

A) `self.var`
B) `ClassName.var`
C) `obj.var`
D) All of the above
Answer: D

17. How can you define a class variable?

```
class A:
    x = 5
```

A) `self.x = 5`
B) `x = 5`
C) `A.x = 5`
D) `var x = 5`
Answer: B

18. What is the result?

```
class Test:
    def greet(self):
        return "Hello"
obj = Test()
print(obj.greet())
```

A) Hello
B) Test
C) Error
D) None
Answer: A

19. What is the output?

```
class A:
    def __init__(self, x):
        self.x = x
obj = A(10)
print(obj.x)
```

A) x
B) 10
C) Error
D) None
Answer: B

20. Which statement is true about class methods?

A) They use `self`
B) They use `cls`
C) They use `self` and `cls`
D) No arguments needed
Answer: B

21. How do you define a class method?

A) `@staticmethod`
B) `@classmethod`
C) `@init`
D) `@class`
Answer: B

22. What is the purpose of `@staticmethod`?

A) Binds method to class
B) Binds method to instance
C) Method that doesn't access class or instance
D) Forces static typing
Answer: C

23. Which of the following is correct?

```
class A:
    pass
a = A()
print(isinstance(a, A))
```

A) True
B) False
C) A
D) Error
Answer: A

24. What does `super()` do?

A) Calls constructor of parent class
B) Inherits class
C) Accesses local variables
D) Creates a new object
Answer: A

25. Which concept allows reusing code?

A) Polymorphism
B) Inheritance
C) Overloading
D) Constructor
Answer: B

26. Which is an example of polymorphism?

A) `len("abc")` and `len([1,2,3])`
B) `x = 5 + 3`

C) `print("Hello")`
D) `x = 10`
Answer: A

27. Which of the following can be used to restrict access to data?

A) public
B) private (with _ or __)
C) class
D) static
Answer: B

28. What is the output?

```
class A:
    def __init__(self):
        self.__x = 5
a = A()
print(a.__x)
```

A) 5
B) Error
C) None
D) __x
Answer: B (It's name mangled)

29. How to access a private variable?

A) Use `self.__var`
B) Use `object._ClassName__var`
C) Directly
D) Can't access
Answer: B

30. Which of these is not a magic method?

A) `__init__()`
B) `__str__()`
C) `__main__()`
D) `__len__()`
Answer: C

Topic: Standard Libraries

1. Which module provides access to mathematical functions?

A) `maths`
B) `math`
C) `cmath`
D) `numbers`
Answer: B

2. What is the output of `math.sqrt(16)`?

A) 4
B) 16
C) 8
D) Error
Answer: A

3. Which module is used for random number generation?

A) `randomize`
B) `math`
C) `random`
D) `numbers`
Answer: C

4. What does `random.randint(1, 5)` return?

A) Always 1
B) Any integer from 1 to 4
C) Any integer from 1 to 5 (inclusive)
D) Only float numbers
Answer: C

5. Which module can be used to get the current date and time?

A) `calendar`
B) `os`
C) `datetime`
D) `time`
Answer: C

6. What does `datetime.datetime.now()` return?

A) Only date
B) Only time

C) Date and time
D) Error
Answer: C

7. Which module is used to interact with the file system?

A) `sys`
B) `os`
C) `io`
D) `platform`
Answer: B

8. What does `os.getcwd()` return?

A) Python version
B) Directory path where Python is installed
C) Current working directory
D) List of all directories
Answer: C

9. What module helps to handle command-line arguments?

A) `cli`
B) `os`
C) `sys`
D) `argparse`
Answer: D

10. What does `sys.exit()` do?

A) Restarts the program
B) Exits from Python script
C) Clears memory
D) Displays OS name
Answer: B

11. Which module is used to compress files in Python?

A) `compress`
B) `shutil`
C) `zipfile`
D) `gzipfile`
Answer: C

12. What does `math.ceil(4.2)` return?

A) 4
B) 5
C) 4.0
D) 5.0
Answer: B

13. What module provides support for regular expressions?

A) `string`
B) `regex`
C) `re`
D) `pattern`
Answer: C

14. Which of these modules helps in serializing Python objects?

A) `pickle`
B) `zipfile`
C) `json`
D) Both A and C
Answer: D

15. What is the use of `time.sleep(2)`?

A) Prints time
B) Delays program for 2 seconds
C) Pauses output
D) Stops time module
Answer: B

16. Which module provides tools for working with iterators?

A) `looptools`
B) `functools`
C) `itertools`
D) `re`
Answer: C

17. Which function from `statistics` module gives the average of a list?

A) `statistics.avg()`
B) `statistics.mean()`
C) `math.mean()`
D) `numpy.mean()`
Answer: B

18. What module is used to interact with operating system environment variables?

A) `env`
B) `os`
C) `sys`
D) `platform`
Answer: B

19. What does `platform.system()` return?

A) Python version
B) System architecture
C) Operating system name (e.g., Windows, Linux)
D) RAM details
Answer: C

20. Which of the following is not a standard library?

A) `math`
B) `random`
C) `os`
D) `numpy`
Answer: D

21. Which module allows you to generate combinations and permutations?

A) `random`
B) `functools`
C) `itertools`
D) `math`
Answer: C

22. Which method is used to generate a random float between 0 and 1?

A) `random.integer()`
B) `random.float()`
C) `random.random()`
D) `math.random()`
Answer: C

23. What does `os.listdir()` return?

A) A string of files
B) A dictionary of files
C) A list of file and directory names

D) A list of only directories
Answer: C

24. What is the use of `shutil` module?

A) File compression
B) File and directory operations
C) Regular expressions
D) Random number generation
Answer: B

25. Which module provides accurate decimal floating point arithmetic?

A) `math`
B) `decimal`
C) `float`
D) `fraction`
Answer: B

26. What does `math.floor(3.9)` return?

A) 3
B) 4
C) 3.9
D) Error
Answer: A

27. Which module is used for parsing command-line arguments?

A) `sys`
B) `optparse`
C) `argparse`
D) `os`
Answer: C

28. Which module provides functions to manipulate dates and times?

A) `time`
B) `datetime`
C) Both A and B
D) `calendar`
Answer: C

29. Which of the following can be used to write JSON data to a file?

A) `json.write()`
B) `json.dump()`
C) `json.store()`
D) `json.output()`
Answer: B

30. What does `calendar.isleap(2024)` return?

A) `False`
B) `None`
C) `True`
D) `2024`
Answer: C

31. Which module is used to open and work with CSV files?

A) `csvfile`
B) `textio`
C) `csv`
D) `spreadsheet`
Answer: C

32. Which module provides tools for functional programming?

A) `functools`
B) `functional`
C) `toolz`
D) `utility`
Answer: A

33. What function from the `re` module matches patterns at the beginning of a string?

A) `re.search()`
B) `re.findall()`
C) `re.match()`
D) `re.compile()`
Answer: C

34. What will `math.pow(2, 3)` return?

A) 6
B) 8
C) 9
D) 2^3 as a string
Answer: B

35. Which module supports multithreading in Python?

A) `process`
B) `multiprocessing`
C) `concurrent`
D) `threading`
Answer: D

36. Which function is used to read environment variables?

A) `os.env()`
B) `os.environ.get()`
C) `sys.getenv()`
D) `platform.env()`
Answer: B

37. What does `sys.argv` provide?

A) OS version
B) Python version
C) Command-line arguments
D) System name
Answer: C

38. What is the output of `math.pi`?

A) 3.14
B) 3.141592653589793
C) pi
D) Error
Answer: B

39. Which module is used to display formatted time?

A) `calendar`
B) `datetime`
C) `time`
D) `os`
Answer: C

40. Which module can you use to serialize Python objects into byte streams?

A) `json`
B) `pickle`
C) `marshal`

D) `base64`

Answer: B

1. What is the output of `len([1, 2, 3])`?

A) 2
B) 3
C) Error
D) None
✅ **Answer: B**

2. Which of the following is a mutable data type?

A) list
B) tuple
C) string
D) int
✅ **Answer: A**

3. Which method adds an element at the end of a list?

A) append()
B) insert()
C) extend()
D) add()
✅ **Answer: A**

4. How can you remove all elements from a list?

A) del list
B) list.remove()
C) list.clear()
D) list.delete()
✅ **Answer: C**

5. Which of the following is used to create a tuple?

A) `{}`
B) `[]`
C) `()`

D) `<>`
✅ **Answer: C**

---

6. Which of these data types is immutable?

A) list
B) dictionary
C) set
D) tuple
✅ **Answer: D**

---

7. Which method is used to remove a particular element from a set?

A) remove()
B) discard()
C) pop()
D) All of the above
✅ **Answer: D**

---

8. Sets in Python are:

A) Ordered
B) Indexed
C) Mutable
D) Allow duplicate values
✅ **Answer: C**

---

9. Which of the following will create an empty dictionary?

A) `{}`
B) `dict()`
C) `[]`
D) Both A and B
✅ **Answer: D**

---

10. How do you access a value in a dictionary?

A) dict.key
B) dict.value
C) dict[key]
D) dict{key}
✅ **Answer: C**

---

11. What is the output of: `len({'a': 1, 'b': 2, 'c': 3})`**?**

A) 1
B) 2
C) 3
D) Error
✅ **Answer: C**

---

12. Which operation is not allowed on sets?

A) add
B) update
C) append
D) discard
✅ **Answer: C**

---

13. Which data structure uses LIFO?

A) Queue
B) Stack
C) List
D) Set
✅ **Answer: B**

---

14. Which module provides support for array data structure?

A) struct
B) array
C) list

D) collections
✅ **Answer: B**

15. What will `my_list = [1, 2, 3]; print(my_list[3])` produce?

A) 3
B) Error
C) None
D) 0
✅ **Answer: B**

16. Which keyword is used to define a function in Python?

A) function
B) def
C) fun
D) define
✅ **Answer: B**

17. Which data structure does not allow duplicates?

A) list
B) tuple
C) dictionary
D) set
✅ **Answer: D**

18. What will `list("abc")` return?

A) ['a', 'b', 'c']
B) ['abc']
C) 'a', 'b', 'c'
D) ('a', 'b', 'c')
✅ **Answer: A**

19. The default mode for the `pop()` method in lists removes:

A) First item
B) Last item
C) Random item
D) Middle item
✅ **Answer: B**

20. Which structure follows FIFO?

A) Stack
B) Array
C) Queue
D) List
✅ **Answer: C**

21. Which of the following can be used as a key in a dictionary?

A) List
B) Dictionary
C) Tuple
D) Set
✅ **Answer: C**

22. What is the output of `set([1, 2, 2, 3])`?

A) {1, 2, 2, 3}
B) {1, 2, 3}
C) [1, 2, 3]
D) Error
✅ **Answer: B**

23. Which method returns the number of times a value appears in a list?

A) count()
B) index()
C) find()

D) search()
✅ **Answer: A**

24. What does `my_dict.get('key', 'default')` do?

A) Returns ‘key’
B) Returns ‘default’ if ‘key’ is not found
C) Throws error
D) Adds key
✅ **Answer: B**

25. Which method combines two dictionaries in Python 3.9+?

A) merge()
B) update()
C) `|` operator
D) append()
✅ **Answer: C**

26. What is the output of `len(set([1, 2, 2, 3, 4]))`?

A) 5
B) 4
C) 3
D) Error
✅ **Answer: B**

27. Which of the following removes an item from a list by index?

A) del
B) remove()
C) pop()
D) All of the above
✅ **Answer: D**

28. Which is a Python built-in module for implementing queues with thread-safety?

A) queue
B) list
C) collections
D) os
✅ **Answer: A**

29. Which method returns the index of first matching item?

A) index()
B) find()
C) locate()
D) position()
✅ **Answer: A**

30. What does `reversed([1,2,3])` return?

A) [3, 2, 1]
B) reversed object
C) (3, 2, 1)
D) None
✅ **Answer: B**

31. What is the output of `list(range(3))`?

A) [1, 2, 3]
B) [0, 1, 2]
C) [0, 1, 2, 3]
D) [1, 2]
✅ **Answer: B**

32. Which of the following is the correct syntax to create an empty set?

A) `set = {}`
B) `set()`
C) `{}`
D) `[]`
✅ **Answer: B**

33. Which built-in function returns the largest item in an iterable?

A) largest()
B) max()
C) top()
D) big()
✅ **Answer: B**

34. Which method would you use to get all keys from a dictionary?

A) getkeys()
B) keys()
C) items()
D) values()
✅ **Answer: B**

35. Which of the following removes the first item with the specified value in a list?

A) del
B) pop()
C) remove()
D) discard()
✅ **Answer: C**

36. What is the output of `bool([])`?

A) True
B) False
C) Error
D) None
✅ **Answer: B**

37. What is the time complexity of accessing a value in a dictionary by key?

A) O(n)
B) O(log n)

C) O(1)
D) O(n log n)
✅ **Answer: C**

38. Which data structure uses curly braces `{}` and key-value pairs?

A) List
B) Tuple
C) Set
D) Dictionary
✅ **Answer: D**

39. What does the `update()` method do in a dictionary?

A) Changes values
B) Adds new key-value pairs or updates existing ones
C) Deletes items
D) Resets dictionary
✅ **Answer: B**

40. Which method removes a random element from a set?

A) discard()
B) remove()
C) pop()
D) delete()
✅ **Answer: C**

41. Which collection type is best for storing unique items?

A) List
B) Tuple
C) Set
D) Dictionary
✅ **Answer: C**

42. What is the output of `tuple("abc")`?

A) ('abc')
B) ['a', 'b', 'c']
C) ('a', 'b', 'c')
D) Error
✅ **Answer: C**

---

43. Which of the following methods adds elements from one list to another?

A) append()
B) insert()
C) extend()
D) join()
✅ **Answer: C**

---

44. What is the result of `[1, 2] + [3, 4]`?

A) [1, 2, 3, 4]
B) [1, 2, [3, 4]]
C) Error
D) [4, 6]
✅ **Answer: A**

---

45. Which method gives you a copy of a list?

A) clone()
B) copy()
C) duplicate()
D) replicate()
✅ **Answer: B**

---

46. What data structure is best suited for a Last-In-First-Out (LIFO) use case?

A) List
B) Set
C) Stack

D) Queue
✅ **Answer: C**

---

47. Which keyword is used to iterate over items in a collection?

A) repeat
B) loop
C) iterate
D) for
✅ **Answer: D**

---

48. How do you check if a key exists in a dictionary?

A) `key in dict`
B) `dict.has_key(key)`
C) `dict.exists(key)`
D) `key.exists()`
✅ **Answer: A**

---

49. What is the output of `len((1, 2, (3, 4)))`?

A) 2
B) 3
C) 4
D) Error
✅ **Answer: B**

---

50. Which of the following types is not ordered?

A) List
B) Tuple
C) Dictionary (before Python 3.7)
D) String
✅ **Answer: C**

1. What is the time complexity of linear search in the worst case?

A) O(1)
B) O(log n)
C) O(n)
D) O(n log n)
✅ **Answer: C**

2. Binary search can be applied only on:

A) Unsorted arrays
B) Sorted arrays
C) Any array
D) None
✅ **Answer: B**

3. What is the time complexity of binary search in the best case?

A) O(n)
B) O(log n)
C) O(1)
D) O(n log n)
✅ **Answer: C**

4. Which of the following is not true about linear search?

A) It is simple and easy to implement
B) It works on sorted and unsorted arrays
C) It is faster than binary search
D) It scans each element one by one
✅ **Answer: C**

5. What is the worst-case time complexity of binary search?

A) O(n)
B) O(log n)
C) O(n log n)
D) O(1)

✅ **Answer: B**

6. In binary search, if the element is not found, the search space becomes:

A) Half of the list
B) Entire list
C) Zero
D) Same

✅ **Answer: C**

7. In a list of 1024 elements, what is the maximum number of comparisons in binary search?

A) 10
B) 1024
C) 512
D) 11

✅ **Answer: D**

(Reason: $\log_2(1024) = 10$, but worst-case is $\log_2(n)+1$ comparisons)

8. What will a linear search return if the target is not found?

A) 0
B) -1
C) None
D) Error

✅ **Answer: B** *(commonly used convention)*

9. Which of the following is a recursive implementation of binary search likely to use?

A) Loop
B) Queue
C) Stack
D) Array
✅ **Answer: C** *(recursion uses stack)*

---

10. In binary search, after each comparison, how is the array divided?

A) Randomly
B) Into three equal parts
C) In half
D) It is not divided
✅ **Answer: C**

---

11. What is required before performing binary search?

A) Sorting the array
B) Checking length
C) Counting elements
D) Nothing
✅ **Answer: A**

---

12. What is the average-case time complexity of linear search?

A) O(n)
B) O(log n)
C) O(1)
D) $O(n^2)$
✅ **Answer: A**

---

13. Which search algorithm is better for a small number of elements?

A) Linear Search
B) Binary Search
C) Hash Search
D) None

✅ **Answer: A**

---

14. Which of the following is correct for binary search algorithm?

A) Starts at first element
B) Always checks the last element
C) Uses the middle element
D) Uses hash function

✅ **Answer: C**

---

15. Which search method guarantees finding the first occurrence of a value in an unsorted list?

A) Binary Search
B) Linear Search
C) Hash Search
D) Jump Search

✅ **Answer: B**

---

16. In which scenario is linear search preferable over binary search?

A) Sorted list
B) Large list
C) List with only a few elements
D) None

✅ **Answer: C**

---

17. Binary search is a type of:

A) Divide and Conquer algorithm
B) Dynamic Programming

C) Greedy Algorithm
D) Brute Force
✅ **Answer: A**

18. What will `binary_search([1,3,5,7,9], 4)` return (assuming proper implementation)?

A) 2
B) 3
C) -1
D) 1
✅ **Answer: C**

19. What is returned in a successful binary search?

A) Boolean value
B) Index of found element
C) All elements
D) Middle element
✅ **Answer: B**

20. Which of the following is true about both linear and binary search?

A) Both require sorted input
B) Both use recursion
C) Both can find a missing element
D) Both return the index of the element if found
✅ **Answer: D**

Let me know if you'd like:

- A **printable version** (PDF/Doc/Excel)
- **Code examples** of linear and binary search in Python
- Or **interactive quiz format** with scoring!

4o

You said:
10 more question

ChatGPT said:

Here are **10 more MCQs** on **"Searching: Linear and Binary Search"** in Python, continuing from the previous sets, with correct answers included:

🔍 More MCQs on Linear and Binary Search

---

21. What is the main limitation of binary search?

A) Can’t search large lists
B) Requires sorted data
C) Needs recursion
D) Returns Boolean only
✅ **Answer: B**

---

22. Which of the following is not true for linear search?

A) It can search in unsorted arrays
B) It compares each element one by one
C) It requires the list to be sorted
D) It works well with small datasets
✅ **Answer: C**

---

23. Which one of the following uses divide-and-conquer strategy?

A) Linear Search
B) Binary Search
C) Jump Search
D) Linear Probing
✅ **Answer: B**

---

24. Which statement is true about binary search?

A) Works faster than linear search on unsorted data
B) Needs the list to be sorted before searching
C) Always takes O(n) time
D) Searches each element sequentially
✅ **Answer: B**

25. Which Python module provides a built-in implementation of binary search on sorted sequences?

A) heapq
B) array
C) bisect
D) search
✅ **Answer: C**

26. In binary search, if the target is less than the middle element, what happens next?

A) Search right half
B) Search whole list again
C) Search left half
D) Return -1
✅ **Answer: C**

27. What will be the output of `linear_search([2, 4, 6, 8], 6)` if implemented correctly?

A) 2
B) 3
C) -1
D) None
✅ **Answer: A**

28. Which type of error will occur if you perform binary search on an unsorted array?

A) TypeError
B) Logical Error (wrong result)
C) Syntax Error
D) Runtime Error

✅ **Answer: B**

29. What is the maximum number of comparisons required in linear search for a list of 50 items?

A) 49
B) 50
C) 25
D) 10

✅ **Answer: B**

30. Which search algorithm is better suited for linked lists?

A) Binary Search
B) Linear Search
C) Interpolation Search
D) Jump Search

✅ **Answer: B**

31. What is the main limitation of binary search?

A) Can’t search large lists
B) Requires sorted data
C) Needs recursion
D) Returns Boolean only

✅ **Answer: B**

32. Which of the following is not true for linear search?

A) It can search in unsorted arrays
B) It compares each element one by one
C) It requires the list to be sorted

D) It works well with small datasets
✅ **Answer: C**

---

33. Which one of the following uses divide-and-conquer strategy?

A) Linear Search
B) Binary Search
C) Jump Search
D) Linear Probing
✅ **Answer: B**

---

34. Which statement is true about binary search?

A) Works faster than linear search on unsorted data
B) Needs the list to be sorted before searching
C) Always takes O(n) time
D) Searches each element sequentially
✅ **Answer: B**

---

35. Which Python module provides a built-in implementation of binary search on sorted sequences?

A) heapq
B) array
C) bisect
D) search
✅ **Answer: C**

---

36. In binary search, if the target is less than the middle element, what happens next?

A) Search right half
B) Search whole list again
C) Search left half
D) Return -1
✅ **Answer: C**

---

37. What will be the output of `linear_search([2, 4, 6, 8], 6)` if implemented correctly?

A) 2
B) 3
C) -1
D) None
✅ **Answer: A**

---

38. Which type of error will occur if you perform binary search on an unsorted array?

A) TypeError
B) Logical Error (wrong result)
C) Syntax Error
D) Runtime Error
✅ **Answer: B**

---

39. What is the maximum number of comparisons required in linear search for a list of 50 items?

A) 49
B) 50
C) 25
D) 10
✅ **Answer: B**

---

40. Which search algorithm is better suited for linked lists?

A) Binary Search
B) Linear Search
C) Interpolation Search
D) Jump Search
✅ **Answer: B**

21. Which sorting algorithm is similar to the method used for sorting playing cards in hands?

A) Selection Sort
B) Merge Sort
C) Insertion Sort
D) Bubble Sort
✅ **Answer: C**

---

22. What is the average-case time complexity of Bubble Sort?

A) $O(n)$
B) $O(\log n)$
C) $O(n^2)$
D) $O(n \log n)$
✅ **Answer: C**

---

23. Which of the following sorts is adaptive?

A) Bubble Sort
B) Insertion Sort
C) Both A and B
D) Selection Sort
✅ **Answer: C**

---

24. Which sorting algorithm performs better when the number of elements is large?

A) Bubble Sort
B) Selection Sort
C) Insertion Sort
D) None of these
✅ **Answer: D**

---

25. What happens if no swaps are made in a pass of bubble sort?

A) The algorithm continues
B) The list is reversed
C) The algorithm stops early

D) An error occurs
✅ **Answer: C**

---

26. Which is the most swap-efficient among the three: bubble, selection, insertion?

A) Bubble Sort
B) Selection Sort
C) Insertion Sort
D) All are same
✅ **Answer: B**

---

27. In Insertion Sort, which element is considered in the second pass?

A) First
B) Second
C) Third
D) Last
✅ **Answer: C**

---

28. Selection Sort finds the minimum element in:

A) First half only
B) Entire unsorted part
C) Whole list
D) Sorted part only
✅ **Answer: B**

---

29. What is the main purpose of the inner loop in Bubble Sort?

A) To find minimum element
B) To insert elements
C) To compare and swap adjacent elements
D) To sort using divide and conquer
✅ **Answer: C**

---

30. Which sort is most suitable for real-time systems where swaps are costly?

A) Bubble Sort
B) Insertion Sort
C) Selection Sort
D) Quick Sort
✅ **Answer: C**

---

31. Which sorting technique does not require checking every adjacent pair each time?

A) Bubble Sort
B) Insertion Sort
C) Selection Sort
D) Heap Sort
✅ **Answer: C**

---

32. In which sort does the number of comparisons always remain the same regardless of order?

A) Bubble Sort
B) Selection Sort
C) Insertion Sort
D) None
✅ **Answer: B**

---

33. Which sort is guaranteed to make at most (n-1) swaps in all cases?

A) Bubble Sort
B) Insertion Sort
C) Selection Sort
D) Merge Sort
✅ **Answer: C**

---

34. In Python, how do we insert elements in Insertion Sort?

A) Use `insert()` method
B) Swap until in position
C) Shift larger elements

D) Use slicing
✅ **Answer: C**

35. Which sorting algorithm keeps growing a sorted list from left to right?

A) Bubble Sort
B) Insertion Sort
C) Selection Sort
D) Radix Sort
✅ **Answer: B**

36. Which algorithm always compares all remaining elements to find the minimum?

A) Insertion Sort
B) Selection Sort
C) Bubble Sort
D) Heap Sort
✅ **Answer: B**

37. What type of sorting is Bubble Sort?

A) Divide and conquer
B) Exchanging sort
C) Insertion-based sort
D) Counting sort
✅ **Answer: B**

38. What happens if we do not use the swap condition in Bubble Sort?

A) It runs faster
B) It does not sort correctly
C) It sorts using insertion
D) Nothing changes
✅ **Answer: B**

39. Is Selection Sort a stable sort?

A) Yes
B) No
C) Sometimes
D) Only in Python
✅ **Answer: B**

---

40. In Selection Sort, how many comparisons are made in the first pass for `n` elements?

A) n
B) n-1
C) n-2
D) 1
✅ **Answer: B**

---

41. Which sorting algorithm requires the most number of swaps?

A) Bubble Sort
B) Selection Sort
C) Insertion Sort
D) None
✅ **Answer: A**

---

42. In worst case, how many comparisons does Insertion Sort make for `n` elements?

A) n
B) n log n
C) $n^2/2$
D) n(n-1)/2
✅ **Answer: D**

---

43. In Python, what happens when you call `.sort()` on a list?

A) Bubble Sort
B) Merge Sort (Timsort)
C) Selection Sort

D) Insertion Sort
✅ **Answer: B**

44. Which sort is least efficient on large datasets?

A) Selection Sort
B) Merge Sort
C) Bubble Sort
D) Quick Sort
✅ **Answer: C**

45. Which sorting algorithm checks all previous elements before inserting?

A) Selection Sort
B) Insertion Sort
C) Merge Sort
D) Bubble Sort
✅ **Answer: B**

46. What does `swap(arr, i, j)` function do in Bubble Sort?

A) Sorts the whole list
B) Replaces all i with j
C) Exchanges arr[i] with arr[j]
D) Copies arr[i] to arr[j]
✅ **Answer: C**

47. Which sort will finish early if the list becomes sorted before all passes?

A) Selection Sort
B) Insertion Sort
C) Bubble Sort
D) Both B and C
✅ **Answer: D**

48. What is the time complexity of selection sort in the average case?

A) $O(n \log n)$
B) $O(n^2)$
C) $O(n)$
D) $O(\log n)$
✅ **Answer: B**

49. Which sorting method uses the fewest comparisons in the best case?

A) Bubble Sort
B) Insertion Sort
C) Selection Sort
D) None
✅ **Answer: B**

50. What kind of algorithm is Insertion Sort?

A) Exchange sort
B) Divide and conquer
C) Incremental sort
D) Partition-based sort
✅ **Answer: C**