

PRACTICE WORKSHEET + FULL ANSWER KEY

Algorithms, Python and debugging

Secondary computer science. Work through the sections in order, or choose the one that matches this week's topic.

QUESTIONS	SECTIONS	ANSWER KEY	WORKING SPACE
104	10	At the back	Ruled

Name _____ Class _____ Date _____

Before you start

Trace a program before running it. Track how values change and test ordinary, boundary and invalid inputs. An algorithm should explain a repeatable method independently of a specific interface.

WORKED EXAMPLE

What does this Python code print? `total = 0; then for n in [2, 4, 6]: total = total + n; then print(total).`

- 1 Start at `total = 0`.
- 2 After each iteration the total is 2, then 6, then 12.
- 3 The program prints 12.

Use this method as your model. Set out your own working the same way.

Common mistakes to avoid

- Confusing assignment with equality
- Testing only one convenient input

There are 104 questions across 10 sections. Write in the ruled space, then mark your work against the answer key at the back. Topics covered: Variables and conditions, Loops, Testing and debugging.

SECTION A Values, variables and expressions

Trace the value of every variable before you run the code. Write down what you expect first.

1

What is printed by `print(3 * 4 + 2)`?

3

What is printed by `print(10 % 3)`?

2

What is printed by `print(10 // 3)`?

4

What is printed by `print(2 ** 5)`?

5

What is the value of x after `x = 5` then `x = x + 3`?

6

State the data type of 7, 7.0 and "7" in Python.

7

What is printed by `print("7" + "3")`?

8

What is printed by `print(int("7") + 3)`?

9

What is printed by `print(len("robot"))`?

10

Write a line that asks for a name and stores it in the variable name.

11

Explain the difference between `=` and `==` in Python.

SECTION B Selection and conditions

Read each condition aloud as a sentence. Check which branch runs when the first test is true.

12

Write a condition that checks whether age is at least 13.

13

Write a condition that is true when score is between 50 and 79 inclusive.

14

What is printed? `x = 4` then `if x > 5: print("big") else: print("small")`

15

What is printed? `x = 10` then `if x > 5: print("A") elif x > 8: print("B")`

16

Rewrite using `elif`: `if grade >= 80 print A; if grade >= 60 print B.`

17

Write a condition that is true when a number n is even.

18

What does not `(a and b)` evaluate to when a is True and b is False?

19

Write an `if` statement that prints valid only when a password is 8 or more characters.

20

Explain why the order of `elif` branches matters.

21

What is printed by `print(5 == "5")`?

22

Write a condition that checks whether a year is a leap year.

SECTION C Loops and lists

Write a trace table for the first three passes of each loop before you predict the output.

23 A loop runs for `n` in `range(4)`. List the values of `n`.

24 List the values produced by `range(2, 10, 3)`.

25 What is printed? `total = 0` then for `n` in `[2, 4, 6]`: `total = total + n` then `print(total)`

26 Write a loop that prints the numbers 1 to 5.

27 Write a loop that adds the numbers 1 to 100 and prints the total.

28 What is the value of `scores[0]` when `scores = [8, 3, 9]`?

29 What is the value of `scores[-1]` when `scores = [8, 3, 9]`?

30 Write a line that adds the value 5 to the end of the list `scores`.

31 What does `len(scores)` return when `scores = [8, 3, 9]`?

32 Write a while loop that keeps asking until the user types yes.

33 Explain one cause of an infinite loop.

SECTION D Functions and algorithms

Describe the method in words first. An algorithm should make sense without a specific language.

34 Write a function that returns the square of a number.

35 Write a function `average(numbers)` that returns the mean of a list.

36 Why is an empty list a useful test case for that average function?

37 Describe an algorithm to find the largest of three numbers.

38

Describe a linear search of a list.

39

State one advantage of binary search over linear search.

40

Write pseudocode that prints only the even numbers in a list.

41

What is returned by a function with no return statement?

42

Explain the difference between a parameter and an argument.

43

Explain why functions make a program easier to maintain.

44

Trace this: `def f(n): return n * 2` then `print(f(3) + f(4))`.

SECTION E Testing, debugging and computational thinking

Test the ordinary case, the boundary case and the invalid case. Record what you expected and what happened.

45

State the difference between a syntax error and a logic error.

48

Explain what a boundary test is and give an example.

46

Name the error type raised by `print(1/0)`.

49

A program prints nothing. Name two things to check first.

47

Write three test cases for a check that a password is at least 8 characters.

50 Explain why testing one convenient input is unsafe.

51 Describe how print statements help you find a bug in a loop.

52 Explain decomposition and give one example.

53 Explain abstraction with a programming example.

54 Trace $x = 1$ then while $x < 5$: $x = x * 2$. List the values x takes.

SECTION F Strings and text

Remember that indexing starts at 0 and a slice stops before its end value.

55 What is printed by `print("robot"[0])`?

56 What is printed by `print("robot"[-1])`?

57 What is printed by `print("robot"[1:4])`?

58 What is printed by `print("ab" * 3)`?

59 What is printed by `print("Hello".upper())`?

60 What is printed by `print("Hello World".split())`?

61 Write a condition that checks whether the string word contains the letter e.

62 Write a line that prints first and last with a space between them.

63 What is printed by `print(len("a b c"))`?

64 Write a loop that prints each character of "code" on its own line.

SECTION G Dictionaries, files and validation

A dictionary looks a value up by key. Never trust input without checking it.

65 Write a dictionary storing the ages of Ana (13) and Ben (14).

66 What is the value of ages["Ben"] for that dictionary?

67 Write a line that adds Cara aged 12 to that dictionary.

68 Write a loop that prints every key in ages.

69 State one advantage of a dictionary over a list.

70 Write a line that opens data.txt for reading.

71 State why a file should be closed after use.

72 Write a loop that keeps asking until the user enters a number from 1 to 10.

73 Explain why user input should always be validated.

74 Name the error raised by int("abc").

SECTION H Sorting, searching and efficiency

Describe the method in words first, then count the comparisons it needs.

75 Describe a bubble sort in two sentences.

76 Sort 5, 2, 9, 1 into ascending order.

77 How many comparisons does a linear search of 100 items need in the worst case?

78 How many comparisons does a binary search of 100 sorted items need in the worst case?

79 State the precondition for a binary search.

80 Explain why a binary search halves the problem at each step.

81

Write a line that sorts the list scores in place.

82

What does `max(scores)` return for `scores = [4, 11, 7]`?

83

Explain why an algorithm that scales well matters for large data.

84

Describe how you would find duplicates in a list.

SECTION I Data representation

Write the place values above the digits before you convert.

85

Convert the binary number 1011 to denary.

86

Convert the denary number 20 to binary.

87

Convert the binary number 11111111 to denary.

88

How many different values can 8 bits represent?

89

Convert the denary number 200 to hexadecimal.

90

Convert the hexadecimal number 1F to denary.

91

How many bits are there in one byte?

92

How many bytes are there in 2 kilobytes, taking 1 kB as 1,000 bytes?

93

Explain why computers use binary.

94

State what a character set such as ASCII does.

SECTION J Mixed review and program design

Plan, then code, then test. Write the expected output before you run anything.

95

Write pseudocode for a program that reads 10 numbers and prints the largest.

96

Describe an algorithm that decides whether a number is prime.

97

Trace: `total = 0` then for `n` in `range(1, 5)`: `total = total + n * n`. State the final total.

98

Explain what variable scope means.

99

Explain the purpose of a comment in code.

100

Name two things that make code easier to read.

101

What is printed by `print(sum([1, 2, 3]) / len([1, 2, 3]))`?

102

Write a condition that checks the list `scores` is not empty.

103

Describe a test plan for a program that turns a percentage into a grade.

104

Explain the difference between iteration and recursion.

Answer key

Work each section before you check it. These are model answers: an equivalent method, or a different well-supported interpretation, can be just as correct.

SECTION A - VALUES, VARIABLES AND EXPRESSIONS

1. 14.
2. 3, because // is integer division.
3. 1, the remainder.
4. 32.
5. 8.
6. int, float and str.
7. 73, because the two strings are joined.
8. 10.
9. 5.
10. name = input("What is your name? ")
11. = assigns a value to a variable; == tests whether two values are equal.

SECTION B - SELECTION AND CONDITIONS

12. age >= 13
13. score >= 50 and score <= 79
14. small.
15. A, because the first true branch runs and the rest are skipped.
16. if grade >= 80: print("A") elif grade >= 60: print("B")
17. n % 2 == 0
18. True, because a and b is False.
19. if len(password) >= 8: print("valid")
20. The first true condition runs, so a broader test placed first can hide a later, more specific one.
21. False, because an int and a str are not equal.
22. (year % 4 == 0 and year % 100 != 0) or year % 400 == 0

SECTION C - LOOPS AND LISTS

23. 0, 1, 2, 3.
24. 2, 5, 8.
25. 12.
26. for n in range(1, 6): print(n)
27. total = 0; for n in range(1, 101): total = total + n; print(total). The total is 5050.
28. 8.
29. 9.
30. scores.append(5)
31. 3.
32. answer = ""; while answer != "yes": answer = input("Type yes: ")
33. The loop condition never becomes false, for example a counter that is never updated inside the loop.

SECTION D - FUNCTIONS AND ALGORITHMS

34. def square(n): return n * n
35. def average(numbers): return sum(numbers) / len(numbers)
36. It exposes a division-by-zero error or missing validation.
37. Store the first as largest, then compare the second and third in turn, updating largest whenever a greater number is found.
38. Check each item in order until the target is found or the list ends.
39. It is far faster on large data sets, but the list must already be sorted.
40. FOR each n IN list: IF n MOD 2 = 0 THEN OUTPUT n
41. None.
42. A parameter is the name in the function definition; an argument is the value passed in when it is called.
43. Logic is written once, named, tested separately and reused, so a change is made in only one place.
44. 14, because 6 + 8.

SECTION E - TESTING, DEBUGGING AND COMPUTATIONAL THINKING

45. A syntax error breaks the language rules and stops the program running; a logic error runs but produces the wrong result.
46. ZeroDivisionError, which is a runtime error.
47. A 7-character string (False), an 8-character string (True) and an empty string (False).
48. A test at the edge of a valid range, for example ages 12, 13 and 14 for a rule of 13 or over.
49. Whether the function is actually called, and whether the condition inside it is ever true.
50. It misses the invalid, boundary and unusual cases where bugs are most likely to hide.
51. Print the loop variable and the running total on each pass, then compare them with your expected trace.
52. Breaking a problem into smaller parts, for example separating input, calculation and output.
53. Hiding detail behind a name, for example calling average(scores) without needing to know how the mean is computed.
54. 1, 2, 4 and then 8, at which point the loop stops.

SECTION F - STRINGS AND TEXT

55. r, because indexing starts at 0.
56. t, because -1 counts back from the end.
57. obo

- 58. ababab
- 59. HELLO
- 60. ['Hello', 'World']
- 61. "e" in word
- 62. print(first + " " + last)
- 63. 5, because the two spaces are characters too.
- 64. for letter in "code": print(letter)

SECTION G - DICTIONARIES, FILES AND VALIDATION

- 65. ages = {"Ana": 13, "Ben": 14}
- 66. 14
- 67. ages["Cara"] = 12
- 68. for name in ages: print(name)
- 69. Values are looked up by a meaningful key rather than by numeric position.
- 70. file = open("data.txt", "r")
- 71. It releases the resource and makes sure buffered data is actually written to disk.
- 72. n = 0 then while n < 1 or n > 10: n = int(input("1 to 10: "))
- 73. Users make mistakes and can enter unexpected values, which would otherwise crash the program or corrupt the data.
- 74. ValueError.

SECTION H - SORTING, SEARCHING AND EFFICIENCY

- 75. Compare each neighbouring pair and swap them if they are in the wrong order. Repeat the passes until no swaps are needed.
- 76. 1, 2, 5, 9
- 77. 100
- 78. 7, because the range halves each time.
- 79. The list must already be sorted.
- 80. Comparing with the middle item rules out one half of the remaining range.
- 81. scores.sort()
- 82. 11
- 83. A poor algorithm takes time that grows much faster than the data, so it becomes unusable at scale.
- 84. Track the values already seen in a set, and report any value that appears a second time.

SECTION I - DATA REPRESENTATION

- 85. 11
- 86. 10100
- 87. 255
- 88. 256
- 89. C8
- 90. 31
- 91. 8 bits.

- 92. 2,000 bytes
- 93. Electronic components have two reliable states, on and off, which map directly to 1 and 0.
- 94. It maps each character to a unique binary code so text can be stored and transmitted.

SECTION J - MIXED REVIEW AND PROGRAM DESIGN

- 95. SET largest TO the first input; FOR each of the remaining 9 inputs: IF n > largest THEN SET largest TO n; OUTPUT largest
- 96. Treat any n below 2 as not prime, then test divisibility from 2 up to the square root of n; an exact divisor means not prime.
- 97. 30, from 1 + 4 + 9 + 16.
- 98. The region of a program where a name is visible; a variable created inside a function is normally local to it.
- 99. It explains why the code does something, for a human reader; it is ignored when the program runs.
- 100. Meaningful names and consistent indentation; small, well-named functions also help.
- 101. 2.0
- 102. len(scores) > 0
- 103. Test a normal value, each boundary between grades, 0, 100 and an invalid input such as -5 or text.
- 104. Iteration repeats using a loop; recursion repeats by a function calling itself, and needs a base case to stop.

Review and next steps

Mark your work against the answer key, record your score for each section, then answer the three questions below. Bring this page to your next lesson so your tutor can start from what actually blocked you.

Which question did you find hardest, and what exactly stopped you?

Rewrite the corrected working for that question here.

Which section do you want to practise again before your next lesson?

Score record

SECTION	QUESTIONS	NUMBER CORRECT	PRACTISE AGAIN?
A. Values, variables and expressions	11		
B. Selection and conditions	11		
C. Loops and lists	11		
D. Functions and algorithms	11		
E. Testing, debugging and computational thinking	10		
F. Strings and text	10		
G. Dictionaries, files and validation	10		
H. Sorting, searching and efficiency	10		
I. Data representation	10		
J. Mixed review and program design	10		
Total	104		

Keep going with one-to-one support.

This is an original Noble Educators topic sample, not a complete syllabus and not an official exam paper. Check the topic, level and any required methods with your school or tutor.

Book a free demo lesson at nobleeducators.com/contact