

AI 210 Prerequisite Test *

Beaver-Edge AI Institute

If you are not sure whether you are ready to take AI 210, please complete this prerequisites test. There is no time limit on solving problems below.

If you can solve 70% of the problems, you are ready to take AI 210.

1. Variables and Data Types

1. Assign the value 42 to a variable `x` and print its type.
2. Swap the values of `a` and `b` without using a third variable.
3. Check if a variable `x` is an instance of `float`.
4. Convert the integer 99 to a float and print it.
5. Add an integer and a float and print the result.
6. Demonstrate the difference between `/` and `//` using two integers.
7. Print the result of `"Hi" * 3`.
8. Assign a complex number `3+4j` to a variable and print its type.
9. Convert `3.99` to an integer and print it.
10. Print the remainder of `17 % 5`.
11. Demonstrate exponentiation using `2 ** 5`.
12. Show how to use the `+=` operator with an integer.
13. Print the type of the result of `5 + 3.2`.
14. Assign `None` to a variable and print its type.
15. Create two variables, `x=10` and `y=2`, and print `x/y` and `x//y`.
16. Print the absolute value of `-15`.

*Copyright © Beaver-Edge AI Institute. All Rights Reserved. No part of this document may be copied or reproduced without the written permission of Beaver-Edge AI Institute.

2. Strings

17. Extract the substring "thon" from "Python".
18. Reverse the string "abcdef".
19. Count the occurrences of 'a' in "banana".
20. Convert "hello world" to uppercase.
21. Check if "Python" starts with "Py".
22. Concatenate "Hello" and "World" with a space in between.
23. Find the index of 'W' in "Hello World".
24. Replace "Python" with "Java" in "I love Python!".
25. Print only the even-indexed characters of "abcdefg".
26. Split the string "apple,banana,cherry" by comma.
27. Join a list ["Red","Green","Blue"] into a single string with commas.
28. Print the last character of "Hello".
29. Check if the substring "world" is in "hello world".
30. Print the length of "Mississippi".
31. Use an f-string to print "My name is John and I am 30 years old."

3. Lists

32. Create a list of numbers [1, 2, 3, 4, 5] and print its length.
33. Extract the last three elements of [1, 2, 3, 4, 5].
34. Append "Python" to an empty list and print the list.
35. Remove "banana" from a list of fruits if it exists.
36. Sort the list [3, 1, 4, 2, 5] in ascending order.
37. Insert 100 at index 2 in [10, 20, 30, 40].
38. Create a new list that is the concatenation of [1,2] and [3,4].
39. Multiply a list [1, 2] by 3 and print the result.
40. Find the index of 4 in [2, 4, 6, 8].
41. Reverse the list [10, 20, 30, 40] in place.
42. Print the sum of all elements in [1, 2, 3, 4, 5].

43. Create a list from a string "hello" using `list()`.
44. Count how many times 2 appears in `[2, 3, 2, 5, 2]`.
45. Remove all elements from a list using `.clear()`.
46. Slice `[1, 2, 3, 4, 5]` to get `[2, 3, 4]`.
47. Create a list comprehension that squares each element in `[1, 2, 3]`.
48. Filter out even numbers from `[1, 2, 3, 4, 5, 6]` using a list comprehension.
49. Extend `[1, 2]` with `[3, 4, 5]` using `.extend()`.
50. Create a 2D list `[[1,2],[3,4]]` and print `[3,4]`.

4. Tuples & Sets

51. Create a tuple `(10, 20, 30)` and print its second element.
52. Attempt to modify the first element of `(1, 2, 3)` and note what happens.
53. Convert the list `[1, 2, 3]` into a tuple.
54. Unpack `(100, 200, 300)` into three variables `a, b, c`.
55. Check if 2 is in the tuple `(1, 2, 3)`.
56. Create a set from `[1, 2, 2, 3, 4, 4, 5]` and print it.
57. Find the union of `{1, 2}` and `{2, 3}`.
58. Find the intersection of `{10, 20, 30}` and `{20, 30, 40}`.
59. Check if `{1, 2}` is a subset of `{1, 2, 3}`.
60. Remove 3 from `{1, 2, 3, 4}`.
61. Find the difference between `{1, 2, 3}` and `{2, 3, 4}`.
62. Find the symmetric difference between `{1, 2, 3}` and `{2, 3, 4}`.
63. Check if `{1, 2}` is disjoint from `{3, 4}`.
64. Add 100 to the set `{10, 20, 30}`.
65. Clear all elements from a set.
66. Convert the string "hello" into a set and print it.
67. Create a tuple from `range(5)` and print it.
68. Find the length of the tuple `("a", "b", "c")`.
69. Check if "x" is in the tuple `("x", "y", "z")`.

5. Dictionaries

70. Create a dictionary with keys "name" and "age", and print name's value.
71. Add "city": "New York" to the dictionary person.
72. Check if "name" is a key in person.
73. Remove the key "age" from person.
74. Create a dictionary from two lists ["a", "b", "c"] and [1,2,3].
75. Iterate over the dictionary person and print each key and value.
76. Update person with another dictionary new_info = {"age": 30}.
77. Print only the keys of person using .keys().
78. Print only the values of person using .values().
79. Create a nested dictionary family with keys "parent", "child" each mapping to another dict.
80. Increment the value of "age" in person by 1.
81. Check if "Alice" is one of the values in person.
82. Create a dictionary comprehension that maps each integer i in [1,2,3] to its square.
83. Clear the dictionary person completely.

6. Control Flow

84. Check if a number n is positive, negative, or zero using if-elif-else.
85. Print "Yes" if x is between 5 and 10 (inclusive).
86. Print "Divisible by 3" if x is divisible by 3, otherwise "Not divisible".
87. Check if y is None.
88. Write an if-elif-else chain for letter grades: A,B,C,D,F.
89. Print "Multiple of 10" if x is a multiple of 10, else print "Not a multiple".
90. Write a nested if to check if x > 0, then if x is even or odd.
91. Check if a number n is within [5, 15] using a single condition.
92. Print "x is 10" if x == 10, else print "x is not 10".
93. Print "Yes" if (a > 0 or b > 0) else "No".

7. Loops

94. Print numbers from 1 to 5 using a `for` loop.
95. Sum numbers from 1 to 100 using a `for` loop.
96. Print only even numbers between 1 and 10 using a loop.
97. Use a `while` loop to print "Hello" 3 times.
98. Use a loop to find the maximum value in [5, 3, 9, 2, 8].
99. Demonstrate the use of `break` by stopping a loop when `i == 3`.
100. Demonstrate the use of `continue` by skipping `i == 2`.
101. Print the indices and elements of ["a", "b", "c"] using `enumerate`.
102. Print numbers from 10 down to 1 using a `for` loop with a negative step.
103. Use a `while` loop to sum digits of a number `n = 123`.
104. Print the Fibonacci sequence up to `n=10` terms using a `for` loop.
105. Multiply all numbers in [1, 2, 3, 4] using a loop.
106. Print each character of "Python" in reverse order using a loop.
107. Use a `while` loop to print powers of 2 up to 128.
108. Use a `for` loop with `range` to print odd numbers from 1 to 9.
109. Print the index and square of each element in [1, 2, 3] using `enumerate`.

8. Functions

110. Define a function `square(n)` that returns `n**2`.
111. Define a function `add(a, b)` that returns their sum.
112. Define a function `string_length(s)` that returns `len(s)`.
113. Define a function `is_even(n)` that checks if `n` is even.
114. Define a function `multiply_list(lst)` that returns the product of elements.
115. Define a function `factorial(n)` that computes `n!`.
116. Define a function `count_vowels(s)` that counts vowels in a string.
117. Write a function `max_of_three(a,b,c)` that returns the largest.
118. Define a function `palindrome(s)` that checks if `s` is a palindrome.
119. Define a function `sum_digits(n)` that returns sum of digits of `n`.

120. Define a function `remove_duplicates(lst)` that returns a list with duplicates removed (preserving order).
121. Write a function `merge_dicts(d1, d2)` that merges two dictionaries (values in `d2` override `d1`).
122. Define a function `all_even(lst)` that returns `True` if all elements are even.

9. Classes

123. Create a class `Person` with attributes `name` and `age`.
124. Instantiate a `Person` object with `name="Alice"` and `age=30`, then print `person.name`.
125. Add a method `greet` in `Person` that prints "Hello, I am X".
126. Create a subclass `Student` inheriting from `Person`, adding `grade`.
127. Override `greet` in `Student` to print "Hi, I'm X and I'm in grade Y".
128. Create a class `Car` with attributes `brand` and `year`, and a method `info()`.
129. Create a `Car` object and call its `info()` method.
130. Implement a class `Rectangle` with `width` and `height`, and a method `area()`.
131. Create a class `BankAccount` with methods `deposit` and `withdraw`.
132. Define a class `Animal` with a method `make_sound()`, and a subclass `Dog` overriding it.
133. Implement a class `Stack` with methods `push`, `pop`, `is_empty`.