#115 Problems on Basic Python | Python with Problem Solving

Python is a popular high-level programming language known for its simplicity and readability. We are going to explore a range of programming problems in Python covering different topics such as **variables, strings, lists, tuples, functions, loops, dictionaries, and classes**. Each problem includes a brief description and a sample input/output. Including detailed **functional** and procedural styles, to cover instructions code snippets, ensuring a comprehensive understanding of Python concepts. Suitable for beginners and experienced developers alike, these 115 problems related to basic Python. *[Python for Everybody]*

Basic Problems:

1. Variable Swap: Write a Python program to swap the values of two variables without using a temporary variable.

2. Even or Odd: Write a Python program that takes an integer as input and prints whether it is even or odd.

3. String Reversal: Write a Python function to reverse a given string and return the reversed string.

4. Temperature Conversion: Given a list of temperatures in Celsius, convert and print the equivalent in Fahrenheit.

5. Temperature Converter: Write a Python program that takes a temperature in Celsius and converts it to Fahrenheit. Take the Celsius temperature as input from the user.

6. Daily Total Sales Calculator: Given a list of daily sales, write a function to calculate the total sales of the weekdays (e.g., [0, "Mon", "Tue", "Wed", "Thu", "Fri", 0]).

7. String Palindromes: Write a Python function to check if a given string is a palindrome or not.

8. String Reversal with Strings: Write a Python function to reverse a given string using slicing.

9. Count Occurrences: Write a Python function that takes a list and a target value as input and returns the number of times the target value appears in the list.

10. Type Casting Challenge: Given three variables, a = 100.5, b = 10, and c = 10.0, write a Python program to perform the following operations and print the results: - Convert a to an integer and add it to b. - Convert c to a float and subtract it from a. - Convert the final result to a string and concatenate it with the string "Life is awesome".

Python Conditional Statements:

11. Positive, Negative, or Zero: Write a Python program that takes a number as input and prints whether it is positive, negative, or zero.

12. Largest of Three Numbers: Write a Python program that takes three numbers as input and prints the largest number.

13. Leap Year Checker: Write a Python program that takes a year as input and determines if it is a leap year or not.

14. Grade Classifier: Write a Python program that takes a student's marks as input and prints the corresponding grade according to the following criteria: - 90% or above: A+ - 80-89%: A - 70-79%: B - 60-69%: C - Below 60%: Fail

15. Vowel Counter: Write a Python program that takes a single character as input and determines if it is a vowel or a consonant.

16. Triangle Classification: Write a Python program that takes the lengths of three sides of a triangle as input and prints the type of triangle (e.g., "Equilateral", "Isosceles", "Scalene").

17. Quadratic Equation Solver: Write a Python program that takes the coefficients (a, b, c) of a quadratic equation as input and calculates and prints the real roots (if they exist) or a message indicating the complex roots.

18. Number Ranges: Write a Python program that takes an integer as input and prints whether the number falls within the ranges: 0-50, 51-100, 101-150, or 151-200.

For & While Loops:

19. Sum of N Numbers: Write a Python program using a for loop to calculate the sum of the first N natural numbers, where N is taken as input from the user.

20. Factorial Calculator: Write a Python program using a while loop to calculate the factorial of a given number N.

21. Table of a Number: Write a Python program using a for loop to print the multiplication table of a given number N.

22. Count Digits in a Number: Write a Python program using a while loop to count the number of digits in a given integer N.

23. Fibonacci Sequence: Write a Python program using a for loop to generate the Fibonacci sequence up to a given term N.

24. Sum of Even Numbers: Write a Python program using a while loop to calculate the sum of all even numbers between 1 and N, where N is taken as input from the user.

25. Prime Numbers: Write a Python program using nested for loops to print prime numbers, such as a split-weight triangle, an inverted split-weight triangle, and an hourglass.

26. Prime Number Checker: Write a Python program using a while loop to check if a given number N is prime or not.

27. Prime Number Generator: Write a Python program using a for loop to find the sum, average, maximum, and minimum values in the list.

28. Reverse String: Write a Python program using a while loop to reverse a given string.

Nested Loops:

29. Multiplication Table: Write a Python program using nested loops to print the multiplication table from 1 to 10.

30. Prime Patterns: Write a Python program using nested loops to print the following pattern:

Nested Lists Problems:

31. Matrix Multiplication: Write a Python program using nested loops to multiply two matrices.

32. Checkerboard Pattern: Write a Python program using nested loops to print a checkerboard pattern (alternating "#" and "O" characters) of size 8×8 (8. Number Pyramid) Write a Python program using nested loops to print a number pyramid like the following: 1 2 3B: mirror diagonal.

List Problems:

33. Lists Basic: Write a Python program to find the sum of all elements in a given list of integers.

34. Lists Average: Write a Python program to calculate the average of all elements in a given list of integers.

35. Lists Max and Min: Write a Python program to find the maximum and minimum values in a given list.

36. Lists Sorting: Write a Python program to sort a list of integers in ascending order.

37. Lists Filtering: Write a Python program to create a new list containing only the even numbers from the original list.

38. List Reversal: Write a Python program to reverse a given list without using any built-in functions.

39. List Manipulation: Given two lists, write a Python program to create a new list that contains elements common to both lists.

40. List Element Count: Write a Python program to count the occurrences of a specific element in a given list.

41. List Duplicates Removal: Write a Python program to remove duplicates from a given list while preserving the order of the elements.

42. List Comprehension: Given a list of integers, write a Python program to create a new list that contains the squares of the elements using list comprehension.

Tuple Problems:

43. Tuple Basics: Write a Python program to create a tuple with three elements and print the tuple and assign the values to three variables.

44. Tuple Sorting: Write a Python program to sort a tuple of integers in ascending order.

45. Tuple Frequency Count: Write a Python program to count the frequency of each element in the tuple.

46. Tuple to List: Write a Python program to convert a tuple into a list.

47. Tuple Reversal: Write a Python program to reverse a tuple without using any built-in functions.

48. Tuple Slicing: Write a Python program to slice a tuple and print the first three elements.

49. Tuple Operations: Given two tuples of integers, write a Python program to perform element-wise addition, subtraction, and multiplication and create new tuples for each operation.

50. Tuple Nested Indexing: Given a nested tuple, write a Python program to access and print the element at a specified index.

51. Tuple Packing: Write a Python program to pack three variables into a single tuple and print the tuple.

Nested List Problems:

52. Tuple Concatenation: Write a Python program to concatenate two tuples and create a new tuple.

53. Tuple Unpacking: Given a tuple with three elements (a, b, c), write a Python program to unpack the tuple and assign the values to three variables.

54. Tuple Sorting: Write a Python program to sort a tuple of integers in ascending order.

55. Tuple Frequency Count: Write a Python program to count the frequency of each element in the tuple.

56. Tuple to List: Write a Python program to convert a tuple into a list.

57. Tuple Reversal: Write a Python program to reverse a tuple without using any built-in functions.

58. Tuple Slicing: Write a Python program to slice a tuple and print the first three elements.

59. Tuple Operations: Given two tuples of integers, write a Python program to perform element-wise addition, subtraction, and multiplication and create new tuples for each operation.

60. Tuple Nested Indexing: Given a nested tuple, write a Python program to access and print the element at a specified index.

61. Tuple Packing: Write a Python program to pack three variables into a single tuple and print the tuple.

Set Problems:

62. Set Basics: Write a Python program to create a set with three elements and print the set.

63. Set Intersection: Given two sets A and B, write a Python program to find their intersection and print the common elements.

64. Set Union: Given two sets A and B, write a Python program to find their union and print all the distinct elements from both sets.

65. Set Difference: Given two sets A and B, write a Python program to find the difference between set A and set B (i.e., elements present in set A but not in B) and print the result.

66. Set Symmetric Difference: Given two sets A and B, write a Python program to find the symmetric difference between the two sets (i.e., elements that are present in either set A or set B, but not in both) and print the result.

67. Set Operations: Given three sets A, B, and C, write a Python program to find and print the intersection of A and B, the union of B and C, and the difference between C and A.

68. Set Subset Check: Given two sets A and B, write a Python program to check if set A is a subset of set B and print the result.

69. Set Superset Check: Given two sets A and B, write a Python program to check if set A is a superset of set B and print the result.

70. Set Length Check: Write a Python program that takes a set as input and prints the number of elements in the set.

71. Set Membership Test: Write a Python program that takes a set and an element as input and prints if the element is present in the set. Print "Found" if the element is present and "Not Found" otherwise.

Dictionary Problems:

72. Dictionary Manipulation: Given a dictionary with student names as keys and their corresponding scores as values, write a Python program to add a new student to the dictionary and update the score of an existing student.

73. Dictionary Keys and Values: Write a Python program that takes a dictionary as input and prints the keys and values separately.

74. Dictionary Length: Write a Python program to calculate and print the number of key-value pairs in a given dictionary.

75. Dictionary Value Sum: Given a dictionary with numeric values, write a Python program to calculate the sum of all the values and print the result.

76. Dictionary Merging: Given two dictionaries, write a Python program to merge them into a single dictionary and print the result.

77. Dictionary Key Removal: Given a dictionary and a key, write a Python program to remove a specific item from the dictionary based on user input.

78. Dictionary Sorting: Given a dictionary with names as keys and corresponding ages as values, write a Python program to sort the dictionary based on age in ascending order.

79. Dictionary Keys Search: Given a dictionary and a key, write a Python program to check if the key exists in the dictionary and print a corresponding message.

80. Dictionary Comprehension: Given a list of strings, write a Python program to create a dictionary where the keys are the strings from the list, and the values are their squares.

81. Dictionary Key Filtering: Given a dictionary with integer keys and string values, write a Python program to create a new dictionary that contains only the key-value pairs where the key is even.

Nested Dictionary Problems:

82. Access Nested Dictionary: Given a nested dictionary representing student data (name, grade, and age), write a Python program to access and print specific information such as a student's name, age, and address.

83. Nested Dictionary Length: Write a Python program to calculate and print the total number of key-value pairs in a nested dictionary.

84. Nested Dictionary Modification: Given a nested dictionary representing student data, write a Python program to update an employee's salary based on their employee ID.

85. Nested Dictionary Sorting: Given a nested dictionary containing product details (product name, price, and quantity), write a Python program to sort the products based on their prices in ascending order.

86. Nested Dictionary Key Search: Write a Python program that takes a city as input and searches for it in a nested dictionary. If found, print the corresponding value; otherwise, print "Key Not Found".

Break & Continue:

87. Prime Number Checker: Write a Python program that takes a number as input and determines if it is a prime number or not. Use a "for" loop to check for factors. If a factor is found, "break" out of the loop.

88. Even Number Printer: Write a Python program to print all even numbers from 1 to 20. Use a "for" loop and "continue" to skip odd numbers.

89. Password Validation: Write a Python program that takes a password as input and checks if it meets the following criteria: at least 8 characters long, contains both uppercase and lowercase letters, and has at least one digit. If the password is valid, print "Password accepted." If not, use "continue" to prompt the user to enter a password again.

90. Divisible by 3 and 5: Write a Python program to print all numbers from 1 to 100 that are divisible by neither 3 nor 5. Use a "for" loop and "continue" to skip numbers that are not divisible by either 3 or 5.

91. Positive Number Sum: Write a Python program that takes positive numbers as input until a negative number is entered. Then, calculate and print the sum of all positive numbers entered. Use a "while" loop and "break" to stop the loop when a negative number is entered.

92. Word Reverser: Write a Python program that takes a word as input and checks if it is a palindrome (reads the same forwards and backwards). Use "continue" to skip checking the word if its length is less than 3 characters.

93. Even Number Printer: Write a Python program that prints the first 10 odd number from 1 to 20 inclusive. Use a "for" loop and "break" to stop the loop when the first odd number is found.

94. Number Guessing Game: Write a Python program that generates a random number between 1 and 100 and lets the user guess the number. Use a "while" loop, "break" when the correct number is guessed, and "continue" to keep prompting the user until they guess correctly.

95. Vowel Counter: Write a Python program that takes a string as input and counts the number of vowels (a, e, i, o, u) in the string. Use a "for" loop and "continue" to skip processing non-vowel characters.

96. Character Checker: Write a Python program that takes a string as input and checks if it contains all unique characters (no character repeats). Use a "for" loop and "break" when a character repeats.

Functions Problems:

97. Factorial Calculator: Write a Python function called "factorial" that takes an integer as input and returns the factorial. Test the function with different values.

98. Palindrome Checker: Write a Python function called "is_palindrome" that takes a string as input and returns "True" if it is a palindrome and "False" otherwise. Test the function with different words.

99. Even or Odd Checker: Write a Python function called "even_or_odd" that takes an integer as input and returns "Even" if the number is even and "Odd" if the number is odd. Test the function with different numbers.

100. Leap Year Calculator: Write a Python function called "is_leap_year" that takes a year as input and returns "True" if it is a leap year and "False" otherwise. Test the function with different years.

101. Greatest Common Divisor (GCD) Calculator: Write a Python function called "gcd" that takes two integers as input and returns their greatest common divisor. Test the function with different pairs of numbers.

102. Leap Year Checker: Write a Python function called "is_leap_year" that takes a year as input and returns "True" if it is a leap year and "False" otherwise. Test the function with different years.

Variable Scope Problems:

103. Math Operations: Write a Python function called "math_operations" that takes three numbers (a b c) along mysterious(a=2, b=3, c=4) operations and prints the results. Inside the function, define two variables, x and y, and assign them the sum and product of the input numbers.

104. Modifying Global Variables: Write a Python function called "modify_globals" that takes an integer value as input and returns the sum of the input value and the global variable "x". Inside the function, modify the global variable "x" by adding the input value to it. Finally, print the updated value of "x" outside the function.

105. Temperature Converter: Write a Python function called "temperature_converter" that takes a temperature value and a unit (either "C" for Celsius or "F" for Fahrenheit) as input. The function should convert the temperature from one scale to the other using nested functions and return the converted value.

Recursion: [Video]

106. Sum of Digits: Write a recursive Python function called "sum_of_digits" that takes an integer as input and returns the sum of its digits.

107 Binary Search: Write a recursive Python function called "binary_search" that takes a sorted list and a target value as input and returns the index of the target value in the list using binary search. If the target value is not in the list, return -1.

108. Fibonacci Sequence: Write a recursive Python function called "power" that takes two positive integers, base and exponent, as input and returns the value of base raised to the exponent.

We have explored a variety of programming problems in Python covering different topics such as variables, strings, functions, loops, lists, dictionaries, and classes. These problems are designed to challenge your understanding of Python concepts and provide opportunities to practice problem-solving and coding skills. By working through these problems, you can gain a deeper understanding of Python syntax, data structures, and control flow, ultimately, solving problems involving nested structures, such as nested loops, nested lists, and nested functions, can improve your ability to work with complex data arrangements.

Visit Our Course Website *[Python for Everybody]*

Join our Community & Subscribe!

Follow us on LinkedIn: Nurul Islam Khan

- Facebook
- Twitter
- LinkedIn

Check Out Our Course Modules

Learn without rush from affordable video lessons created by Expert Instructors

Become a Python Developer

Mr. Nurul Islam
Lecturer, Daffodil International University
Member in CSE at IEEE Xplore
Email: ainul.bd@gmail.com
Course Details

Data Analysis Specialization

Syed Hasan
Senior Analytics Leader, Grameen Foundation
Email: syed.hasan@gmail.com
Course Details

Become a Big Data Engineer

A.K.M. Mizar Uddin
Associate Data Engineering Lead Engineer at Bangladesh Digital Communications Ltd.
Course Details

Data Science and Machine Learning with Python

Mr. Prashanta Kiron Shaon
Lecturer, Daffodil International University
Dr. Sir - in Data Science (Major in)
Ecq Glasgow Engineering, Germany
Course Details

Deep Learning & AI Specialization

Mr. Nazmul Ahsan Khandaker
Tableau Certified Data Scientist
Email: nazmul@nazmul.info
Course Details

Become a Design Developer

Mr. Abu Nurbian
Senior Instructor, Leapfrog
Adobe Certified Expert
Course Details

Quest Intelligence is a data science center (and Quest) Look for further than a(n) Quest Intelligence, the premier destination for top-notch data science courses in Bangladesh.

- Home
- About
- Blogs
- Follow

Quick link

- Home
- About
- Blogs
- Contact
- Follow

Contact

Email: contact@questiq.bd
Address: Dhaka, Bangladesh - 1230, Bangladesh
Call: +8801750008470 - +8801750008472

Hello and welcome!
How can we help you? :)
Write a message...
Powered by Crisp
Enter your email address (optional)
Enter your message...
continue