**BASIC LOGIC BUILDING (Beginner Level)**

Start with these to understand control flow, loops, conditions.

**✅ Numbers**

* Check whether a number is **even or odd**
* Check if a number is **positive, negative, or zero**
* Check if a number is **prime**
* Print **first N prime numbers**
* Find **factorial** of a number
* Find **GCD** and **LCM** of two numbers
* Check if number is **perfect**, **armstrong**, or **palindrome**
* Count the number of **digits** in a number
* Reverse a number
* Find the **sum of digits** of a number
* Print **Fibonacci series**

**🔹 PATTERN PRINTING (Star / Number Patterns)**

These enhance your **loop control** and nested logic.

**✅ Star Patterns**

* Left triangle
* Right triangle
* Pyramid
* Inverted pyramid
* Diamond shape
* Hollow pyramid
* Floyd's Triangle

**✅ Number Patterns**

* 123
* 1  
  12  
  123
* Pascal’s triangle
* Palindromic triangle
* Number pyramid

**🔹 STRINGS & CHARACTER LOGIC**

Very important for logic building.

**✅ Basic String Problems**

* Count **vowels/consonants**
* Check for **palindrome string**
* Reverse a string
* Check if two strings are **anagrams**
* Convert string to **uppercase/lowercase**
* Count number of **words**
* Remove **duplicate characters**
* Remove **spaces** from a string

**🔹 ARRAY LOGIC**

Learn **iteration, storage, searching, sorting.**

**✅ 1D Array**

* Find the **maximum and minimum** elements
* Sort the array (bubble, selection, insertion)
* Find **sum/average**
* Search an element (linear/binary search)
* Reverse array
* Find duplicate elements
* Merge two arrays
* Rotate array left/right

**✅ 2D Array**

* Matrix addition, subtraction
* Matrix multiplication
* Transpose of matrix
* Find sum of diagonal elements

**🔹 BITWISE & MATHS**

Bit-level logic and efficient computation.

* Check if a number is a power of 2
* Swap two numbers without third variable
* Count number of 1s in binary
* Convert decimal to binary/octal/hex
* Find square root without sqrt()
* Set/Clear/Toggle a specific bit
* XOR of array elements

**🔹 LOGICAL THINKING / PUZZLE-LIKE**

These strengthen real problem-solving skills.

* Find missing number in array 1 to N
* Find duplicate in array where one element occurs twice
* Check if brackets are balanced (stack logic)
* Detect palindrome number/string
* Implement simple calculator
* Convert Roman to integer
* Implement atoi() or string to integer
* N-Queens problem (Recursion)
* Sudoku solver (Backtracking)

**🔹 INTERMEDIATE CHALLENGES**

Start applying **algorithms** and combinations.

* Find **longest palindrome** in a string
* Print all **permutations** of a string
* Merge two sorted arrays
* Implement binary search
* Find pair with sum equal to K
* Print Pascal triangle
* Count frequency of elements
* Run-Length Encoding (RLE compression)
* Find leaders in an array
* Zigzag matrix print

**🔹 ADVANCED PROBLEM SOLVING**

Use recursion, dynamic programming, backtracking.

* Nth Fibonacci using recursion and DP
* Subset sum problem
* Tower of Hanoi
* Rat in a maze
* Snake and Ladder game logic
* Generate valid parenthesis combinations
* Knapsack problem (0/1 knapsack)
* Minimum number of coins for a value
* Word break problem
* Longest common subsequence / substring

**🔹 BONUS CHALLENGES (Fun + Smart)**

* Generate magic square
* Spiral matrix print
* Count the number of vowels in each word
* Tic Tac Toe logic
* Ludo dice roll simulator
* Snake game in console
* Basic chatbot using string matching
* Detect prime using Sieve of Eratosthenes
* Caesar cipher encode/decode

**✅ Tips to Improve Logic Building:**

1. **Start with dry runs** (pen and paper trace).
2. Write your own version **before seeing solutions**.
3. Debug with print statements (or breakpoints).
4. Solve the same problem in **multiple ways**.
5. Move from **brute force to optimized** approach.
6. Participate in platforms: HackerRank, LeetCode, Codeforces.

Would you like:

* Sample programs with code (C, C++, Python)?
* Practice sets with increasing difficulty?
* Interview-style logic puzzles?

Let me know your preferred language and level to continue.

**You said:**

python

**ChatGPT said:**

Updated saved memory

Great! Since you prefer **Python**, here’s a curated path of **logic-building problems with examples** to improve your programming skills step by step.

**🔹 BASIC PYTHON LOGIC PROGRAMS**

**1. Check if number is even or odd**

python

CopyEdit

num = int(input("Enter a number: "))

print("Even" if num % 2 == 0 else "Odd")

**2. Check prime number**

python

CopyEdit

num = int(input("Enter a number: "))

if num > 1:

for i in range(2, num):

if num % i == 0:

print("Not Prime")

break

else:

print("Prime")

else:

print("Not Prime")

**3. Factorial using loop**

python

CopyEdit

n = int(input("Enter number: "))

fact = 1

for i in range(1, n + 1):

fact \*= i

print("Factorial:", fact)

**4. Fibonacci series up to N terms**

python

CopyEdit

n = int(input("Enter number of terms: "))

a, b = 0, 1

for \_ in range(n):

print(a, end=" ")

a, b = b, a + b

**5. Sum of digits**

python

CopyEdit

n = int(input("Enter a number: "))

total = 0

while n > 0:

total += n % 10

n //= 10

print("Sum of digits:", total)

**🔹 STRING LOGIC IN PYTHON**

**6. Palindrome string**

python

CopyEdit

s = input("Enter a string: ")

print("Palindrome" if s == s[::-1] else "Not Palindrome")

**7. Count vowels and consonants**

python

CopyEdit

s = input("Enter a string: ").lower()

vowels = 'aeiou'

vc = cc = 0

for char in s:

if char.isalpha():

if char in vowels:

vc += 1

else:

cc += 1

print("Vowels:", vc, "Consonants:", cc)

**8. Anagram check**

python

CopyEdit

s1 = input("Enter first string: ")

s2 = input("Enter second string: ")

print("Anagram" if sorted(s1) == sorted(s2) else "Not Anagram")

**🔹 PATTERN LOGIC IN PYTHON**

**9. Right-angled triangle pattern**

python

CopyEdit

n = int(input("Enter rows: "))

for i in range(1, n + 1):

print("\*" \* i)

**10. Pyramid of numbers**

python

CopyEdit

n = int(input("Enter rows: "))

for i in range(1, n + 1):

print(" " \* (n - i) + str(i) \* (2 \* i - 1))

**🔹 ARRAY / LIST LOGIC**

**11. Find max and min in a list**

python

CopyEdit

lst = list(map(int, input("Enter numbers: ").split()))

print("Max:", max(lst))

print("Min:", min(lst))

**12. Remove duplicates**

python

CopyEdit

lst = list(map(int, input("Enter numbers: ").split()))

unique = list(set(lst))

print("Unique List:", unique)

**13. Search in list**

python

CopyEdit

lst = [4, 7, 2, 9, 5]

x = int(input("Enter number to search: "))

print("Found" if x in lst else "Not Found")

**14. Reverse list**

python

CopyEdit

lst = [1, 2, 3, 4, 5]

print("Reversed:", lst[::-1])

**🔹 INTERMEDIATE LOGIC**

**15. Fibonacci using recursion**

python

CopyEdit

def fib(n):

if n <= 1:

return n

return fib(n - 1) + fib(n - 2)

for i in range(10):

print(fib(i), end=" ")

**16. Binary to decimal**

python

CopyEdit

binary = input("Enter binary number: ")

decimal = int(binary, 2)

print("Decimal:", decimal)

**17. Palindrome number**

python

CopyEdit

n = int(input("Enter a number: "))

rev = int(str(n)[::-1])

print("Palindrome" if n == rev else "Not Palindrome")

Would you like me to:

* Continue with **pattern programs**?
* Provide **practice sets** (with questions only)?
* Create a **PDF of 100 Python logic programs**?
* Focus on **data structure-based logic**?

Let me know your next step.

**You said:**

also give interest and compound interest

**ChatGPT said:**

Here are **Python programs** to calculate **Simple Interest** and **Compound Interest**, with formulas and examples.

**✅ 1. Simple Interest (SI)**

**🔹 Formula:**

cpp

CopyEdit

Simple Interest (SI) = (P × R × T) / 100

* **P** = Principal amount
* **R** = Rate of interest (annual)
* **T** = Time in years

**🔸 Python Program:**

python

CopyEdit

# Simple Interest Calculator

P = float(input("Enter Principal Amount: "))

R = float(input("Enter Rate of Interest (in %): "))

T = float(input("Enter Time (in years): "))

SI = (P \* R \* T) / 100

print("Simple Interest =", SI)

**🔹 Sample Input/Output:**

java

CopyEdit

Enter Principal Amount: 1000

Enter Rate of Interest (in %): 5

Enter Time (in years): 2

Simple Interest = 100.0

**✅ 2. Compound Interest (CI)**

**🔹 Formula:**

java

CopyEdit

Compound Interest (CI) = A – P

Where A = P \* (1 + R/100)^T

**🔸 Python Program:**

python

CopyEdit

# Compound Interest Calculator

P = float(input("Enter Principal Amount: "))

R = float(input("Enter Rate of Interest (in %): "))

T = float(input("Enter Time (in years): "))

A = P \* (1 + R / 100) \*\* T

CI = A - P

print("Compound Interest =", round(CI, 2))

print("Total Amount =", round(A, 2))

**🔹 Sample Input/Output:**

java

CopyEdit

Enter Principal Amount: 1000

Enter Rate of Interest (in %): 5

Enter Time (in years): 2

Compound Interest = 102.5

Total Amount = 1102.5

Would you also like:

* Monthly/quarterly compound interest?
* Programs to compare SI vs CI?
* All logic problems in a downloadable file?

Let me know how you'd like to continue.

**You said:**

give simple formula

**ChatGPT said:**

Here are the **simple formulas** for both **Simple Interest** and **Compound Interest**:

**✅ Simple Interest (SI)**

SI=P×R×T100\boxed{SI = \frac{P \times R \times T}{100}}SI=100P×R×T​​

Where:

* PPP = Principal amount
* RRR = Rate of interest (annual, in %)
* TTT = Time (in years)

**✅ Compound Interest (CI)**

CI=P×(1+R100)T−P\boxed{CI = P \times \left(1 + \frac{R}{100}\right)^T - P}CI=P×(1+100R​)T−P​

Or,

CI=A−PwhereA=P×(1+R100)T\boxed{CI = A - P} \quad \text{where} \quad A = P \times \left(1 + \frac{R}{100}\right)^TCI=A−P​whereA=P×(1+100R​)T

Where:

* PPP = Principal amount
* RRR = Rate of interest (annual, in %)
* TTT = Time (in years)
* AAA = Final amount after interest

Let me know if you want formulas for **monthly/quarterly compounding** too.

**You said:**

more andmore question

**ChatGPT said:**

Perfect! Here's a **huge list of logic-building questions in Python**, from easy to advanced. These will boost your logical thinking and prepare you for interviews, coding rounds, and Python mastery.

**🔹 LEVEL 1: Beginner Python Logic Questions**

**Numbers:**

1. Find whether a number is even or odd.
2. Find if a number is positive, negative, or zero.
3. Check if a number is a prime number.
4. Print all prime numbers in a range.
5. Check if a number is a perfect number.
6. Find factorial of a number (using loop and recursion).
7. Print Fibonacci series up to N terms.
8. Find the sum of digits of a number.
9. Reverse a given number.
10. Count the number of digits in a number.
11. Find GCD and LCM of two numbers.
12. Convert decimal to binary, octal, and hexadecimal.
13. Check if a number is an Armstrong number.
14. Check if a number is a palindrome.

**🔹 LEVEL 2: Strings and Character Logic**

1. Check if a string is a palindrome.
2. Count the number of vowels and consonants.
3. Count the number of words in a sentence.
4. Reverse a string without using slicing.
5. Find the frequency of each character.
6. Remove duplicate characters from a string.
7. Check if two strings are anagrams.
8. Convert lowercase to uppercase and vice versa.
9. Replace all spaces with hyphens in a string.
10. Remove all punctuation from a string.
11. Capitalize the first letter of each word.
12. Find the longest word in a sentence.

**🔹 LEVEL 3: Pattern Printing (Star and Number Patterns)**

1. Print a right-angled triangle of stars.
2. Print an inverted triangle.
3. Print a pyramid pattern.
4. Print a diamond pattern.
5. Print Floyd's triangle.
6. Print Pascal's triangle.
7. Print number pyramid.
8. Print palindromic triangle.
9. Hollow rectangle/star frame.

**🔹 LEVEL 4: Lists and Array-based Logic**

1. Find the maximum and minimum in a list.
2. Sort a list without using built-in sort().
3. Find sum and average of a list.
4. Remove duplicates from a list.
5. Count frequency of each element in a list.
6. Search for an element (linear/binary search).
7. Reverse a list.
8. Rotate a list left or right.
9. Merge two lists and sort the result.
10. Find the second largest number in a list.
11. Find pairs whose sum equals K.

**🔹 LEVEL 5: Matrix / 2D List Problems**

1. Add two matrices.
2. Multiply two matrices.
3. Transpose of a matrix.
4. Print diagonal elements.
5. Find the sum of upper triangle elements.
6. Spiral matrix print.
7. Snake pattern print.
8. Rotate matrix 90 degrees.

**🔹 LEVEL 6: Bitwise & Math Logic**

1. Check if a number is a power of 2.
2. Count number of set bits in an integer.
3. Swap two numbers using XOR.
4. Find unique element in list using XOR.
5. Multiply without using \* operator.
6. Find square root without math.sqrt().
7. Check if two numbers are equal without using ==.

**🔹 LEVEL 7: Intermediate Logic & Algorithms**

1. Find the missing number in a list from 1 to N.
2. Find the first repeating and non-repeating element.
3. Implement bubble sort / selection sort / insertion sort.
4. Check for balanced parentheses using stack.
5. Implement simple calculator using if-else.
6. Find longest substring without repeating characters.
7. Run-length encoding of a string.
8. Merge intervals.
9. Find duplicate number in array (Floyd’s cycle).
10. Count words starting with a vowel.
11. Reverse words in a sentence.

**🔹 LEVEL 8: Recursion, Backtracking, and Puzzles**

1. Print all permutations of a string.
2. Generate all subsets of a set.
3. Solve the N-Queens problem.
4. Solve Sudoku.
5. Tower of Hanoi.
6. Generate all balanced parentheses of N pairs.
7. Word break problem.
8. Rat in a Maze problem.
9. Knight's tour problem.

**🔹 LEVEL 9: Games and Simulations**

1. Rock, Paper, Scissors game.
2. Number guessing game.
3. Dice roll simulator.
4. Basic calculator with menu.
5. Tic Tac Toe game logic.
6. Snake game in console.
7. Ludo dice simulator.
8. Lottery number generator.

**🔹 LEVEL 10: Real-World Logic Problems**

1. Email validator using regex.
2. Password strength checker.
3. BMI Calculator.
4. Calendar generator.
5. Age calculator (from DOB).
6. Leap year checker.
7. Currency converter using rates.
8. Prime number checker API.
9. Student grade system.
10. Bank interest calculator.
11. Loan EMI calculator.