Table of Contents

## 1 FunctionDef num(a, b)

## 2 Function num

**num**: The function of num is to compute and return the sum of two input arguments.

**parameters**: The parameters of this Function. · a: The first value to be added. · b: The second value to be added.

**Code Description**: This function, named num, accepts two parameters, a and b. It performs an addition operation on these two parameters using the + operator and returns the resulting sum.

**Note**: The function will work with any data types that support the addition (+) operator, such as integers, floats, or even strings (where it will perform concatenation). Passing incompatible types will result in a TypeError.

**Output Example**:

# If the function is called with num(2, 3), the output will be:  
5

## 3 FunctionDef generate\_random\_integers(count, start, end)

## 4 Function generate\_random\_integers

**generate\_random\_integers**: The function of generate\_random\_integers is to return a list of a specified number of pseudo-random integers within a given inclusive range.

**parameters**: The parameters of this Function. · count: An integer specifying the number of random integers to generate. · start: An optional integer representing the inclusive lower bound for the random values, with a default value of 0. · end: An optional integer representing the inclusive upper bound for the random values, with a default value of 100.

**Code Description**: The function first validates the count parameter. If count is a negative number, it raises a ValueError. Next, it checks if the start value is greater than the end value. If it is, the function swaps the two values to ensure the range is always valid. Finally, it uses a list comprehension to generate the list of random integers. It iterates count times, and in each iteration, it calls random.randint(start, end) to produce a single random integer within the inclusive range of [start, end]. The function then returns the complete list of generated integers.

**Note**: This function relies on the random module. If the start parameter is provided with a value greater than the end parameter, the function will automatically correct the order and not raise an error. A negative value for count will result in a ValueError.

**Output Example**:

[42, 1, 88, 23, 99]

## 5 FunctionDef choose\_random\_item(items)

## 6 Function choose\_random\_item

**choose\_random\_item**: The function of choose\_random\_item is to choose a single random item from a non-empty sequence of strings.

**parameters**: The parameters of this Function. · items: A list of strings to choose from.

**Code Description**: The function first checks if the provided items list is empty. If it is, the function raises a ValueError with the message “items must not be empty”. If the list is not empty, it uses the random.choice() method to select a single element uniformly at random from the items list and returns that element.

**Note**: This function requires the input list items to be non-empty. Passing an empty list will result in a ValueError. The function relies on the random module, which must be imported for the code to execute successfully.

**Output Example**:

# Given the input list ["apple", "banana", "cherry"]  
# A possible return value is:  
"banana"

## 7 FunctionDef shuffle\_copy(items)

## 8 Function shuffle\_copy

**shuffle\_copy**: The function of shuffle\_copy is to return a new, randomly shuffled copy of a given list of integers without altering the original list.

**parameters**: The parameters of this Function. · items: A list of integers that will be copied and shuffled.

**Code Description**: The function begins by creating a shallow copy of the input list items and storing it in a new variable named copy. It then uses the random.shuffle() method to shuffle the elements of the copy list in place. The original items list remains unchanged. Finally, the function returns the shuffled copy.

**Note**: This function requires the random module to be imported. It is designed to be non-mutating, meaning the original list passed as an argument is not modified.

**Output Example**:

# Given an input list:  
[1, 8, 2, 7]  
  
# A possible return value could be:  
[7, 1, 8, 2]