Table of Contents

## 0.1 FunctionDef choose\_random\_item

# 1 Function: choose\_random\_item(items: List[str])

## 1.1 Overview

The choose\_random\_item function selects and returns a single random item from a given list of strings.

## 1.2 parameters

* **items** (List[str]): A non-empty list of strings from which a random item will be chosen.

## 1.3 Description

This function provides a safe way to pick a random element from a list. It first performs a validation check to ensure the input list items is not empty. If the list is empty, the function raises a ValueError to prevent runtime errors that would occur from attempting to choose an item from an empty sequence.

If the list contains one or more items, the function then utilizes the random.choice() method from Python’s random module. This method selects a single item uniformly at random from the input list, meaning each item has an equal probability of being chosen. The selected string is then returned as the result.

# Internal logic example  
import random  
  
# Sample list  
my\_options = ["option A", "option B", "option C"]  
  
# The function will return one of the three strings at random  
selected\_option = random.choice(my\_options)

## 1.4 Usage Notes

* The input list items must not be empty. Providing an empty list will result in a ValueError.
* The selection is uniformly random, meaning every item in the list has an equal chance of being chosen.
* This function depends on Python’s built-in random module, which must be imported in the script for random.choice() to be available.

**Output Example**: A single string from the provided list. For a list ['red', 'green', 'blue'], a possible output is 'green'.

## 1.5 Example

import random  
from typing import List  
  
def choose\_random\_item(items: List[str]) -> str:  
 """Choose a single random item from a non-empty sequence."""  
 if not items:  
 raise ValueError("items must not be empty")  
 return random.choice(items)  
  
# Example usage  
fruits = ["apple", "banana", "cherry", "date"]  
random\_fruit = choose\_random\_item(fruits)  
print(random\_fruit)

**Output:**

cherry

## 1.6 FunctionDef shuffle\_copy

# 2 Function: shuffle\_copy(items: List[int])

## 2.1 Overview

The shuffle\_copy function returns a shuffled copy of a given list of integers without modifying the original list.

## 2.2 parameters

* items (List[int]): A list of integers to be shuffled.

## 2.3 Description

This function provides a safe way to get a randomized version of a list while preserving the original data. The logic proceeds as follows:

1. A shallow copy of the input items list is created by calling list(items). This new list is stored in a copy variable, ensuring that the original list passed to the function remains unchanged.
2. The random.shuffle() function is then called on this copy. The random.shuffle() method shuffles the sequence of elements in the copy list in-place, rearranging them into a random order.
3. Finally, the function returns the shuffled copy.

The primary benefit of this function is its non-mutating behavior, which prevents unintended side effects in other parts of a program that might rely on the original, unshuffled list.

## 2.4 Usage Notes

* This function does not modify the original items list. It operates on and returns a new, separate list.
* The function depends on Python’s built-in random module. Ensure this module is imported (import random) in the file where shuffle\_copy is defined or used.
* Although the type hint specifies List[int], the underlying logic will work correctly with a list containing elements of any type (e.g., strings, floats, or objects).

**Output Example**: A possible return value for an input of [1, 2, 3, 4, 5].

[4, 1, 5, 3, 2]

## 2.5 Example

The following example demonstrates how to use shuffle\_copy and confirms that the original list is not altered.

import random  
from typing import List  
  
# The function must be defined or imported in your script  
def shuffle\_copy(items: List[int]) -> List[int]:  
 """Return a shuffled copy of the given list without mutating the input."""  
 copy = list(items)  
 random.shuffle(copy)  
 return copy  
  
# Example usage  
original\_numbers = [10, 20, 30, 40, 50]  
print(f"Original list before calling function: {original\_numbers}")  
  
# Get a shuffled copy  
shuffled\_numbers = shuffle\_copy(original\_numbers)  
print(f"Returned shuffled list: {shuffled\_numbers}")  
  
# Verify the original list is unchanged  
print(f"Original list after calling function: {original\_numbers}")

**Output:**

Original list before calling function: [10, 20, 30, 40, 50]  
Returned shuffled list: [30, 50, 10, 20, 40]  
Original list after calling function: [10, 20, 30, 40, 50]

*(Note: The actual order of elements in the “Returned shuffled list” will vary with each execution due to its random nature.)*