Searching algorithms and performance comparison
Due April 24th 11:59pm
you'll write three functions: one for linear search and one for binary search, the last function to compare the performance with time module
You'll then compare the time complexity of both algorithms by running tests on large datasets.

Instructions:

Task1: (10pts) Linear Search:

- Define a function linear_search(arr, target) that:
  - Takes a list of integers arr and an integer target as input.
  - Iterates over each element in arr and checks if it matches target.
  - Returns the index of target if found, or -1 if target is not in arr.

Task2: (15pts) Binary Search:

- Define a function binary_search(arr, target) that:
  - Takes a sorted list of integers arr and an integer target as input.
  - Uses a binary search algorithm to check if target is in arr.
  - Returns the index of target if found, or -1 if target is not in arr.

Task3: (15pts)Performance Comparison:

- Define a function compare_search_performance() that:
  - Generates 3 large lists of sorted integers (e.g., from 1 to 100,000).
  - Calls both linear_search and binary_search on this list with a specific target (e.g., 10).
  - Measures the time taken by each function using Python's time module and prints the results.

Task4: (30pts) Hash Table for Gym Exercise shopping List

You are designing a simple shopping list for gym and fitness items using a **hash table**.
The hash table should store items like "protein powder", "resistance bands", "yoga mat", "dumbbells", etc.
You must:

- Implement a **hash table of size 51** using a Python list.
- Implement **linear probing** to resolve collisions.
- Implement the following **functions**:
  - hash_function(key: str, size: int) -> int
    (Create a simple hash for the item name.)
  - insert(table: list, key: str) -> None
    (Insert an item into the shopping list.)
  - search(table: list, key: str) -> bool
    (Check if an item is already in the shopping list.)
  - display(table: list) -> None
    (Display all items in the hash table with their index.)
- Starter Hints:
- Initialize the table as: table = [None] * 51
- Example hash function:

```
    def hash_function(key, size):
        total = 0
        for char in key:
            total += ord(char)
    return total % size
```

Sample Output of Task 4:

Gym Shopping List System

- Add item
- Search item
- Show shopping list
- Exit Enter choice: 1 Enter item to add: protein powder

Enter choice: 1 Enter item to add: yoga mat

Enter choice: 3 Shopping List: Index 15: protein powder Index 38: yoga mat ...

Enter choice: 2 Enter item to search: yoga mat Result: Found!

Enter choice: 2 Enter item to search: treadmill Result: Not

Answers

```
import time

# --------------------
# Task 1: Linear Search (10pts)
# --------------------
def linear_search(arr, target):
    for i in range(len(arr)):
        if arr[i] == target:
            return i
    return -1

# --------------------
# Task 2: Binary Search (15pts)
# --------------------
def binary_search(arr, target):
    left = 0
    right = len(arr) - 1
```

```
    while left <= right:
        mid = (left + right) // 2
        if arr[mid] == target:
            return mid
        elif arr[mid] < target:
            left = mid + 1
        else:
            right = mid - 1
    return -1

# --------------------
# Task 3: Compare Search Performance (15pts)
# --------------------
def compare_search_performance():
    test_cases = [
        list(range(1, 100001)),       # 100,000 elements
        list(range(1, 1000001)),      # 1,000,000 elements
        list(range(1, 10000001))      # 10,000,000 elements
    ]
    target = 10
    print("\n--- Performance Comparison ---")
    for i, arr in enumerate(test_cases):
        print(f"\nDataset {i+1} (Size: {len(arr)}):")

        start = time.time()
        linear_search(arr, target)
        end = time.time()
        print(f"Linear Search Time: {end - start:.6f} seconds")

        start = time.time()
        binary_search(arr, target)
        end = time.time()
        print(f"Binary Search Time: {end - start:.6f} seconds")

# --------------------
# Task 4: Hash Table Shopping List (30pts)
# --------------------
TABLE_SIZE = 51
hash_table = [None] * TABLE_SIZE

def hash_function(key, size):
    total = sum(ord(char) for char in key)
    return total % size
```

```
def insert(table, key):
    index = hash_function(key, TABLE_SIZE)
    original_index = index
    while table[index] is not None:
        if table[index] == key:
            print(f"'{key}' is already in the list at index {index}.")
            return
        index = (index + 1) % TABLE_SIZE
        if index == original_index:
            print("Hash table is full! Cannot insert more items.")
            return
    table[index] = key
    print(f"'{key}' added at index {index}.")

def search(table, key):
    index = hash_function(key, TABLE_SIZE)
    original_index = index
    while table[index] is not None:
        if table[index] == key:
            return True
        index = (index + 1) % TABLE_SIZE
        if index == original_index:
            break
    return False

def display(table):
    print("\n--- Shopping List ---")
    for i, item in enumerate(table):
        if item is not None:
            print(f"Index {i}: {item}")
    print()

def Task4_menu():
    while True:
        print("\nGym Shopping List System")
        print("------------------------")
        print("1. Add item")
        print("2. Search item")
        print("3. Show shopping list")
        print("4. Exit")
        choice = input("Enter choice: ")

        if choice == '1':
            item = input("Enter item to add: ").strip().lower()
```

```
            insert(hash_table, item)
        elif choice == '2':
            item = input("Enter item to search: ").strip().lower()
            found = search(hash_table, item)
            if found:
                print(f"Result: '{item}' is in the shopping list.\n")
            else:
                print(f"Result: '{item}' not found.\n")
        elif choice == '3':
            display(hash_table)
        elif choice == '4':
            print("Exiting program. Stay fit!")
            break
        else:
            print("Invalid choice. Try again.\n")



# Run the performance test:
# compare_search_performance()

# Run the gym shopping list hash table menu:
Task4_menu()
```