CSE220: Data Structures (Lab)
Fall 2024
Lab Quiz - 04
Duration: 40 Minutes

B

Name:	ID:	Section:

Question 1 [15 Points]

You are tasked with implementing a **hash table** that stores student records. Each record contains **a student name (a string) and a student ID (an integer)**. You will implement a hash table that uses **forward chaining** (a linked list) to handle collisions. Implement the following methods:

Hash Function: You are given a string representing the student name. You need to calculate a hash index for this string using the following rules: Take the sum of the ASCII values of the characters in the string. If the sum is odd, return the sum modulo the size of the hash table. If the sum is even, return the sum divided by 2, modulo the size of the hash table.

Search: Implement the search() method that takes a student name (string) and returns the corresponding student ID (integer). If the student is found in the hash table, return the student ID. If the student is not found, return None.

Delete: Implement the delete() method that takes a student name (string) and removes the corresponding record from the hash table. If the student is not found, return None.

[No built-in function except len(). Assume the display method and Node class are already implemented]

Sample Input:	**Sample Output:**	**Explanation:**

`ht = HashTable(5)`
`// assume the insert method is called multiple times and some entries are already in the hashtable`
 	
`print(‘Hash table after insertions:\n’)`
`ht.display()`
 	
`ht.search("David")`
 	
`ht.delete("Alice", 123456)`
 	
`print(‘Hash table after deletions:\n’)`
`ht.display()`
	`Hash table after insertions:`
	`Index 0: ("Bob”, 135927)`
	`Index 1: None`
	`Index 2: None`
	`Index 3:("Charlie", 348247)`
	`Index 4: ("Alice", 123456)->("David", 124382)`
	
	`ID of David is 124382`
	
	`Hash table after deletions:`
	`Index 0: ("Bob”, 135927)`
	`Index 1: None`
	`Index 2: None`
	`Index 3:("Charlie", 348247)`
	`Index 4: ("David", 124382)`
		For ‘Alice’, hash function calculation, A → 65, l → 108, i → 105, c → 99, e → 101. The sum of ASCII values = 65 + 108 + 105 + 99 + 101 = 478. Since 478 is even, the hash index = 478 / 2 % 5 = 4.
		For ‘Bob’, B → 66, o → 111, b → 98. The sum of ASCII values = 66 + 111 + 98 = 275. Since 275 is odd, the hash index = 275 % 5 = 0.
		David is found at index 4 and his ID returned.