The National School of Artificial Intelligence
المدرسة الوطنية العليا للذكاء الاصطناعي

2nd Year /Semester3 (2024-2025)

Data Structures & Algorithms 2
Lab 5
Hashing

Exercise 1

Create a basic hash table class that supports the following operations: insertion, search, and removal of values from the table.
You can implement the hash table as an array and use a simple hashing function, such as: key = value % TABLE SIZE, also for this exercise you don't need to find solutions to address the cases of collisions. You are only asked to compare the number of collisions produced by the hashing function following this experiment Setup.
Experiment Setup:

- Set TABLE SIZE to 1000.
- Perform insertion of random values, filling 20%, 50%, and 80% of the hash table capacity.
- Record and compare collision occurrences produced for each filling percentage.

Exercise 2

• Analyze the provided implementation of separate chaining hash table ***(SeparateChaining.h, SeparateChaining.cpp, TestSeparateChaining.cpp)*** and answer the following questions:
1. What are the data structures used to implement this hash table?
2. How are insert and delete operations implemented in the hash table?
3. Why is rehashing used, and how is it implemented?

• Reimplement the separate chaining hash table using a vector of singly linked lists.

Exercise 3

• Analyze the provided implementation of quadratic probing table ***(QuadraticProbing.h,QuadraticProbing.cpp, TestQuadraticProbing.cpp)*** and answer the following questions:
1. What are the data structures used to implement this hash table?
2. How are insert and delete operations implemented in the hash table?
3. How is collision handling implemented?
4. Why is rehashing used, and how is it implemented?

- Implement Linear probing and Double hashing.

Exercise 4

Implement the classic cuckoo hash table in which two separate tables are maintained.

The simplest way to do this is to use a single array and modify the hash function to access either the top half or the bottom half.

Exercise 5

Implement a hopscotch hash table and compare its performance with linear probing, separate chaining, and cuckoo hashing.