**Practical No: 02**

**Implement all the functions of a dictionary (ADT) using hashing Data: Set of (key, value) pairs, Keys are mapped to values, Keys must be comparable, and Keys must be unique Standard Operations: Insert (key, value), Find (key), Delete (key)**

**Algorithm :**

1. **Initialization:**
   * Create a hash table hashTable using unordered\_map to store key-value pairs.
2. **Insert Operation:**
   * Input: key, value
   * If the key exists in hashTable, update the value:  
     hashTable[key] = value
   * If the key does not exist, insert the new key-value pair: hashTable.insert({key, value})
   * Output: Print a message:
     + "Key updated" if the key was updated,
     + "Key inserted" if the key was newly inserted.
3. **Find Operation:**
   * Input: key
   * If the key exists in hashTable, output the associated value:  
     cout << "Value: " << hashTable[key]
   * If the key does not exist, print:  
     "Key not found."
4. **Delete Operation:**
   * Input: key
   * If the key exists, remove the key-value pair:  
     hashTable.erase(key)
   * Output: Print "Key deleted" if successful, or "Key not found" if the key was not found.
5. **Display Operation:**
   * If hashTable is empty, print "Hash table is empty."
   * Otherwise, iterate through the hash table and display all key-value pairs.
6. **Exit:**
   * End the program when the user selects the exit option.

**Main Program Flow:**

1. Display a menu with options: Insert, Find, Delete, Display, and Exit.
2. Loop to handle user input:
   * Execute the corresponding operation based on the user's choice.
   * Repeat the process until the user selects "Exit."

**Program :**

#include <iostream>

#include <unordered\_map>

#include <string>

using namespace std;

class Dictionary {

private:

unordered\_map<string, string> hashTable; // Stores key-value pairs

public:

// Insert a key-value pair into the dictionary

void insert(const string &key, const string &value) {

if (hashTable.find(key) != hashTable.end()) {

cout << "Key '" << key << "' already exists. Updating its value.\n";

}

hashTable[key] = value;

cout << "Inserted/Updated: (" << key << ", " << value << ")\n";

}

// Find a value by its key

void find(const string &key) {

if (hashTable.find(key) != hashTable.end()) {

cout << "Found: (" << key << ", " << hashTable[key] << ")\n";

} else {

cout << "Key '" << key << "' not found.\n";

}

}

// Delete a key-value pair by its key

void remove(const string &key) {

if (hashTable.erase(key)) {

cout << "Key '" << key << "' deleted successfully.\n";

} else {

cout << "Key '" << key << "' not found.\n";

}

}

// Display all key-value pairs in the dictionary

void display() {

if (hashTable.empty()) {

cout << "The dictionary is empty.\n";

} else {

cout << "Current dictionary contents:\n";

for (const auto &pair : hashTable) {

cout << "(" << pair.first << ", " << pair.second << ")\n";

}

}

}

};

int main() {

Dictionary dict;

int choice;

string key, value;

do {

cout << "\nDictionary Operations:\n";

cout << "1. Insert\n";

cout << "2. Find\n";

cout << "3. Delete\n";

cout << "4. Display\n";

cout << "5. Exit\n";

cout << "Enter your choice: ";

cin >> choice;

switch (choice) {

case 1:

cout << "Enter key: ";

cin >> key;

cout << "Enter value: ";

cin >> value;

dict.insert(key, value);

break;

case 2:

cout << "Enter key to find: ";

cin >> key;

dict.find(key);

break;

case 3:

cout << "Enter key to delete: ";

cin >> key;

dict.remove(key);

break;

case 4:

dict.display();

break;

case 5:

cout << "Exiting program.\n";

break;

default:

cout << "Invalid choice. Please try again.\n";

}

} while (choice != 5);

return 0;

}

**Output:**

Dictionary Operations:

1. Insert

2. Find

3. Delete

4. Display

5. Exit

Enter your choice: 1

Enter key: name

Enter value: Alice

Inserted/Updated: (name, Alice)

Dictionary Operations:

1. Insert

2. Find

3. Delete

4. Display

5. Exit

Enter your choice: 1

Enter key: age

Enter value: 25

Inserted/Updated: (age, 25)

Dictionary Operations:

1. Insert

2. Find

3. Delete

4. Display

5. Exit

Enter your choice: 4

Current dictionary contents:

(name, Alice)

(age, 25)

Dictionary Operations:

1. Insert

2. Find

3. Delete

4. Display

5. Exit

Enter your choice: 2

Enter key to find: name

Found: (name, Alice)

Dictionary Operations:

1. Insert

2. Find

3. Delete

4. Display

5. Exit

Enter your choice: 2

Enter key to find: city

Key 'city' not found.

Dictionary Operations:

1. Insert

2. Find

3. Delete

4. Display

5. Exit

Enter your choice: 3

Enter key to delete: age

Key 'age' deleted successfully.

Dictionary Operations:

1. Insert

2. Find

3. Delete

4. Display

5. Exit

Enter your choice: 4

Current dictionary contents:

(name, Alice)

Dictionary Operations:

1. Insert

2. Find

3. Delete

4. Display

5. Exit

Enter your choice: 5

Exiting program.