Name: Snehal Jayprakash Borji
UID: 2023510008
Course: F.Y.M.C.A.
Subject: DS

Practical 09

Aim: use Hashing Techniques to store the records. use any two methods

Coding:

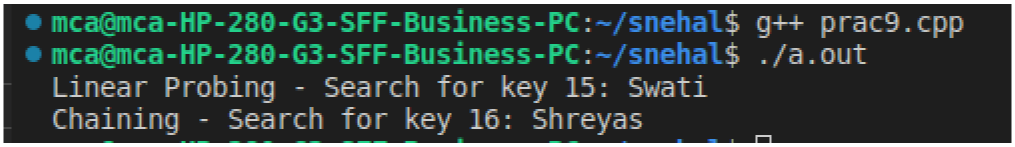
Prac9.cpp

```
#include <iostream>
#include <vector>
#include <list>

using namespace std;

// Structure to represent a record
struct Record {
   int key;
   string value;

   Record(int k, const string& v) : key(k), value(v) {}
};

// Hash table using linear probing
class LinearProbingHashTable {
private:
   int capacity;
   vector<Record*> table;

   int hashFunction(int key) {
       return key % capacity;
   }

public:
   LinearProbingHashTable(int size) : capacity(size), table(size, nullptr)
{}
```

```
   void insert(int key, const string& value) {
       int index = hashFunction(key);

       while (table[index] != nullptr) {
           // Linear probing: move to the next slot if current slot is
occupied
           index = (index + 1) % capacity;
       }

       table[index] = new Record(key, value);
   }

   string search(int key) {
       int index = hashFunction(key);

       while (table[index] != nullptr) {
           if (table[index]->key == key) {
               return table[index]->value;
           }

           // Linear probing: move to the next slot
           index = (index + 1) % capacity;
       }

       return "Record not found";
   }
};

// Hash table using chaining
class ChainingHashTable {
private:
   int capacity;
   vector<list<Record>> table;

   int hashFunction(int key) {
       return key % capacity;
   }

public:
   ChainingHashTable(int size) : capacity(size), table(size) {}
```

```
    void insert(int key, const string& value) {
        int index = hashFunction(key);
        table[index].emplace_back(key, value);
    }

    string search(int key) {
        int index = hashFunction(key);

        for (const auto& record : table[index]) {
            if (record.key == key) {
                return record.value;
            }
        }

        return "Record not found";
    }
};

int main() {
    LinearProbingHashTable linearProbingTable(10);
    ChainingHashTable chainingTable(10);

    // Inserting records
    linearProbingTable.insert(5, "Snehal");
    linearProbingTable.insert(15, "Swati");
    linearProbingTable.insert(25, "Sanika");

    chainingTable.insert(6, "Omkar");
    chainingTable.insert(16, "Shreyas");
    chainingTable.insert(26, "Uday");

    // Searching for records
    cout << "Linear Probing - Search for key 15: " <<
linearProbingTable.search(15) << endl;
    cout << "Chaining - Search for key 16: " << chainingTable.search(16) <<
endl;

    return 0;
}
```

OUTPUT :



```
● mca@mca-HP-280-G3-SFF-Business-PC:~/snehal$ g++ prac9.cpp
● mca@mca-HP-280-G3-SFF-Business-PC:~/snehal$ ./a.out
  Linear Probing - Search for key 15: Swati
  Chaining - Search for key 16: Shreyas
```