Objectives:

In this lab, students will practice:

1. Implementation of Hash Tables with collision resolution strategies

QUESTION 1: (HashMap using Quadratic Probing)

Create a class QHashMap which inherits the HashMap class implemented in the last lab. Override the getNextCandidateIndex(int key, int i) method so that it performs quadratic probing, i.e., add the square of i to the hash value of the key.

Create a main to test these functions properly.

```
int main()
{
cout<<"LINEAR PROBING\n";
HashMap<string> *map=new HashMap<string>;
map->insert(89, "hassan", map->getHashArray());
map->insert(18, "ali", map->getHashArray());
map->insert(49, "ayaan", map->getHashArray());
map->insert(58, "ahsan", map->getHashArray());
map->insert(69, "baber", map->getHashArray());
cout<<"QUADRATIC PROBING\n";
QHashMap<string> *Qmap=new QHashMap<string>;
Qmap->insert(89, "hassan", Qmap->getHashArray());
Qmap->insert(18, "ali", Qmap->getHashArray());
Qmap->insert(49, "ayaan", Qmap->getHashArray());
Qmap->insert(58, "ahsan", Qmap->getHashArray());
Qmap->insert(69, "baber", Qmap->getHashArray());
return 0;
}
```

Question 2: Double Hashing

- The Student class represents a student record with an ID and a name.
- The DoubleHashingHashTable class contains methods for inserting student records, searching for records, and displaying the contents of the hash table.
- The insert method uses double hashing to handle collisions and inserts a new student record into the hash table.
- The search method looks for a student record based on the provided key (student ID).
- The displayTable method is included to visualize the contents of the hash table.

```
int hash1(int key) {
```

```
        return key % size;
    }

    int hash2(int key) {
        // Using a simple secondary hash function for demonstration
        return 1 + (key % (size - 1));
    }
```

If collision not occur then used that hashing
int index = hash1(key);
If collision occur then used that hashing
int step = hash2(key);
index = (hash1(key) + i * step) % size;

```
int main() {
    DoubleHashingHashTable hashTable(10);

    // Insert student records
    hashTable.insert(101, "Alice");
    hashTable.insert(201, "Bob");
    hashTable.insert(301, "Charlie");
    hashTable.insert(401, "David");

    // Display the initial hash table
    hashTable.displayTable();

    return 0;
}
```