

**Given a File of N employee records with a set K of Keys(4- digit) which uniquely determine the records in file F. Assume that file F is maintained in memory by a Hash Table (HT) of m memory locations with L as the set of memory addresses (2-digit) of locations in HT. Let the keys in K and addresses in L are integers. Design and develop a Program in C that uses Hash function  $H: K \rightarrow L$  as  $H(K)=K \bmod m$  (remainder method), and implement hashing technique to map a given key K to the address space L. Resolve the collision (if any) using linear probing.**

```
#include <stdio.h>
```

```
#define EMPTY -1
```

```
int main() {
```

```
    int m, n, key;
```

```
    int i, index;
```

```
    printf("Enter size of hash table (m): ");
```

```
    scanf("%d", &m);
```

```
    int hashTable[m];
```

```
    // Initialize hash table
```

```
    for (i = 0; i < m; i++) {
```

```
        hashTable[i] = EMPTY;
```

```
    }
```

```
    printf("Enter number of employee records (N): ");
```

```
    scanf("%d", &n);
```

```

for (i = 0; i < n; i++) {
    printf("Enter 4-digit key %d: ", i + 1);
    scanf("%d", &key);

    index = key % m; // Hash function

    // Linear probing for collision resolution
    while (hashTable[index] != EMPTY) {
        index = (index + 1) % m;
    }

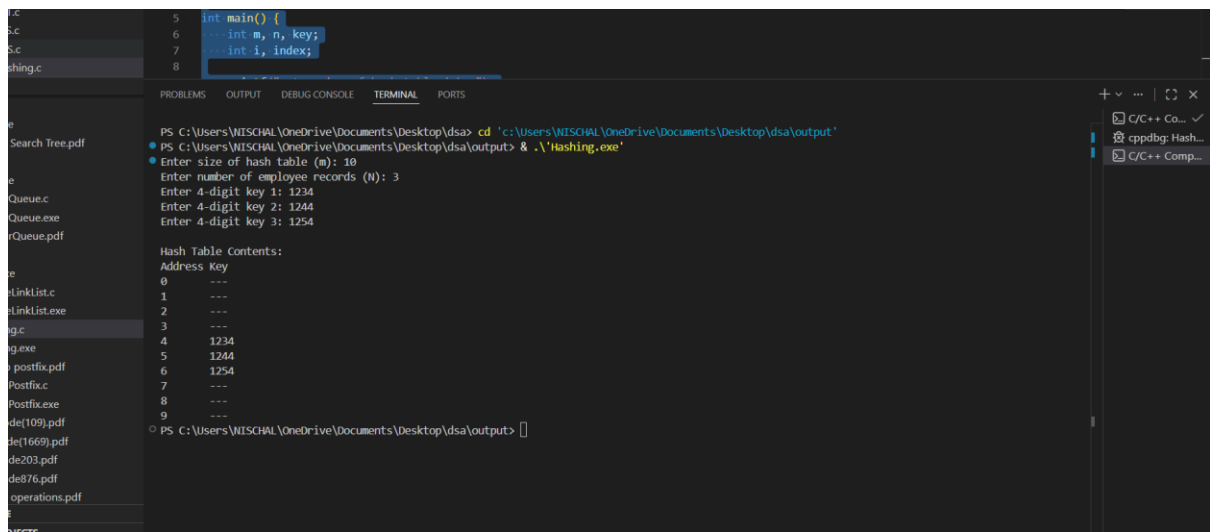
    hashTable[index] = key;
}

// Display hash table
printf("\nHash Table Contents:\n");
printf("Address\tKey\n");
for (i = 0; i < m; i++) {
    if (hashTable[i] != EMPTY)
        printf("%d\t%d\n", i, hashTable[i]);
    else
        printf("%d\t---\n", i);
}

return 0;
}

```

## OUTPUT:



The screenshot shows a Visual Studio Code editor with a C++ file named `hashing.c` open. The code defines a `main` function that takes three arguments: `m` (size of hash table), `n` (number of records), and `key` (4-digit keys). The program uses a hash table to store the keys. The terminal output shows the execution of the program, where the user enters the size of the hash table (10), the number of records (3), and three 4-digit keys (1234, 1244, 1254). The output then displays the contents of the hash table.

```
5 int main() {
6     int m, n, key;
7     int i, index;
8
```

Terminal Output:

```
PS C:\Users\WISCHAL\OneDrive\Documents\Desktop\dsa> cd 'c:\Users\WISCHAL\OneDrive\Documents\Desktop\dsa\output'
PS C:\Users\WISCHAL\OneDrive\Documents\Desktop\dsa\output> & .\hashing.exe
Enter size of hash table (m): 10
Enter number of employee records (N): 3
Enter 4-digit key 1: 1234
Enter 4-digit key 2: 1244
Enter 4-digit key 3: 1254

Hash Table Contents:
Address Key
0 ---
1 ---
2 ---
3 ---
4 1234
5 1244
6 1254
7 ---
8 ---
9 ---

PS C:\Users\WISCHAL\OneDrive\Documents\Desktop\dsa\output>
```