

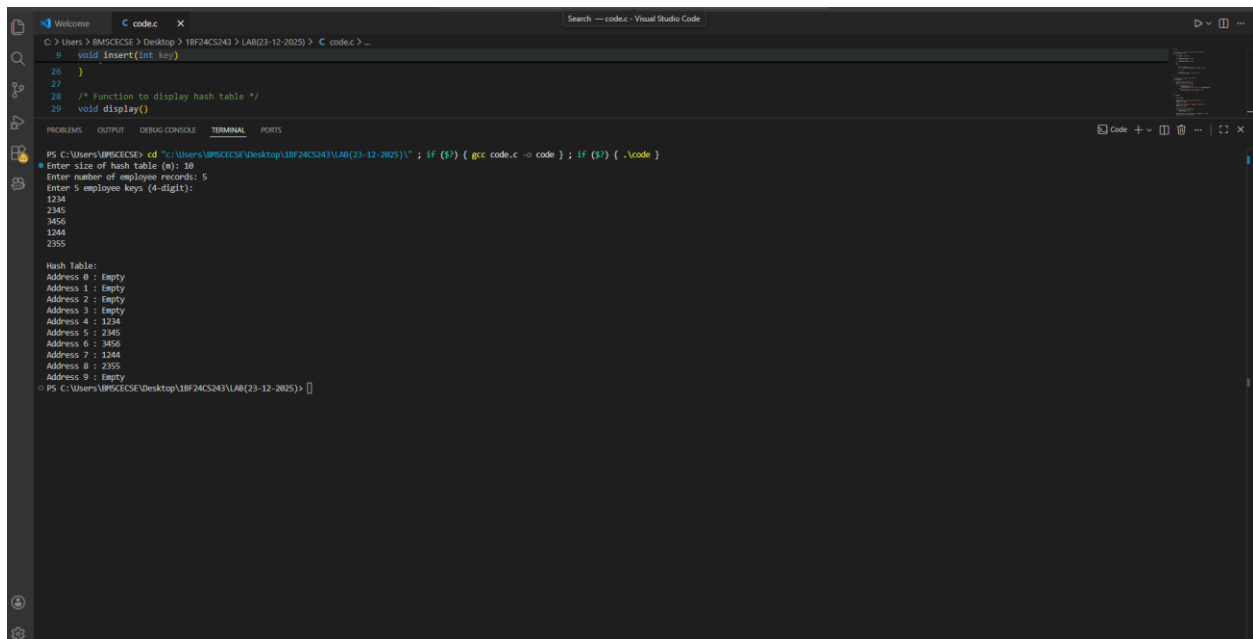
## LAB(23-12-2025)

INPUT:

```
1 #include <stdio.h>
2
3 #define MAX 20
4
5 int hashTable[MAX];
6 int m;
7
8 /* Function to Insert key using Linear Probing */
9 void insert(int key)
10 {
11     int index = key % m;
12     if (hashTable[index] == -1)
13     {
14         hashTable[index] = key;
15     }
16     else
17     {
18         int i = 1;
19         while (hashTable[(index + i) % m] != -1)
20         {
21             i++;
22         }
23         hashTable[(index + i) % m] = key;
24     }
25 }
26
27 /* Function to display hash table */
28 void display()
29 {
30     printf("\nHash Table:\n");
31     for (int i = 0; i < m; i++)
32     {
33         if (hashTable[i] != -1)
34             printf("Address %d : %d\n", i, hashTable[i]);
35         else
36             printf("Address %d : Empty\n", i);
37     }
38 }
39
40 int main()
41 {
42     int n, key;
43
44     printf("Enter size of hash table (m): ");
45     scanf("%d", &m);
46
47     printf("Enter number of employee records: ");
```

```
28 /* Function to display hash table */
29 void display()
30 {
31     printf("\nHash Table:\n");
32     for (int i = 0; i < m; i++)
33     {
34         if (hashTable[i] != -1)
35             printf("Address %d : %d\n", i, hashTable[i]);
36         else
37             printf("Address %d : Empty\n", i);
38     }
39 }
40
41 int main()
42 {
43     int n, key;
44
45     printf("Enter size of hash table (m): ");
46     scanf("%d", &m);
47
48     printf("Enter number of employee records: ");
49     scanf("%d", &n);
50
51     /* Initialize hash Table */
52     for (int i = 0; i < m; i++)
53         hashTable[i] = -1;
54
55     printf("Enter %d employee keys (4-digit):\n", n);
56     for (int i = 0; i < n; i++)
57     {
58         scanf("%d", &key);
59         insert(key);
60     }
61
62     display();
63
64     return 0;
65 }
```

OUTPUT:



The screenshot shows a Visual Studio Code editor with a C file named `code.c`. The code defines a hash table with 10 slots and a function to display its contents. The terminal output shows the program's execution, including prompts for the hash table size and the number of records, followed by the display of the hash table state.

```
9 void insert(int key)
26 }
27
28 /* Function to display hash table */
29 void display()

PS C:\Users\BMSGECSE> cd "C:\Users\BMSGECSE\Desktop\BIF24CS243\LAB(23-12-2025)" ; if ($?) { gcc code.c -o code } ; if ($?) { .code }
Enter size of hash table (n): 10
Enter number of employee records: 5
Enter 5 employee keys (4-digit):
1234
2345
3456
1244
2355

Hash Table:
Address 0 : Empty
Address 1 : Empty
Address 2 : Empty
Address 3 : Empty
Address 4 : 1234
Address 5 : 2345
Address 6 : 3456
Address 7 : 1244
Address 8 : 2355
Address 9 : Empty

PS C:\Users\BMSGECSE\Desktop\BIF24CS243\LAB(23-12-2025)>
```