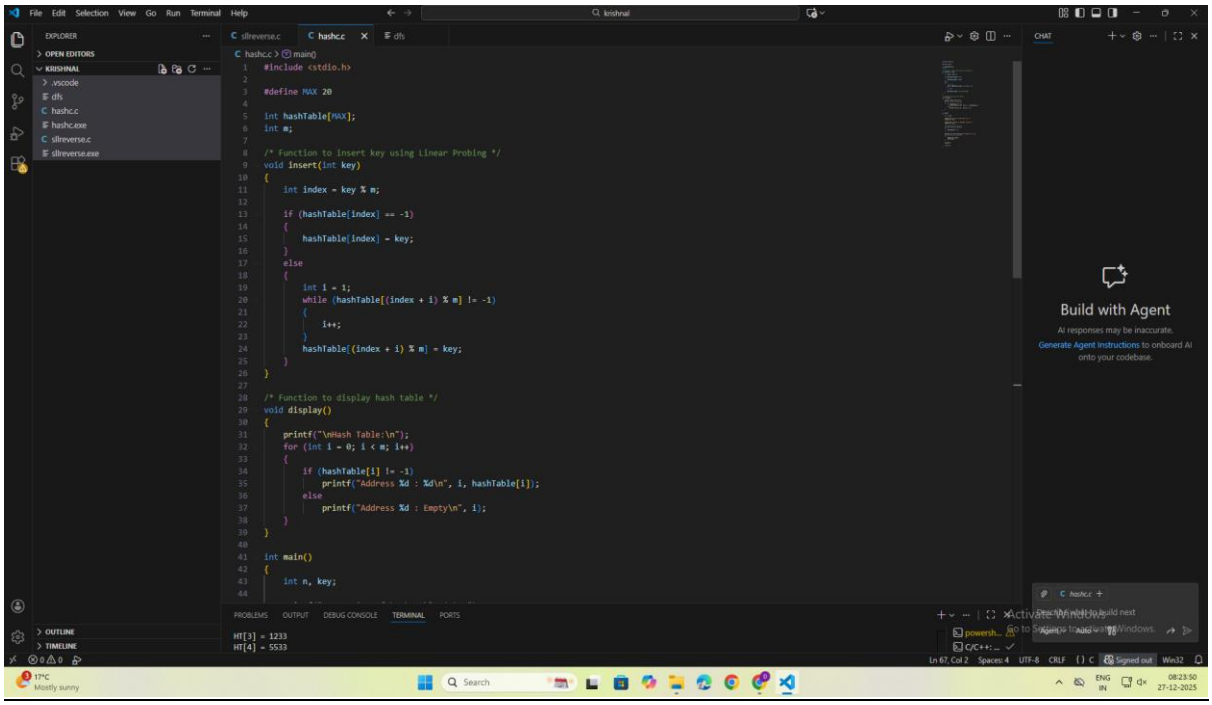
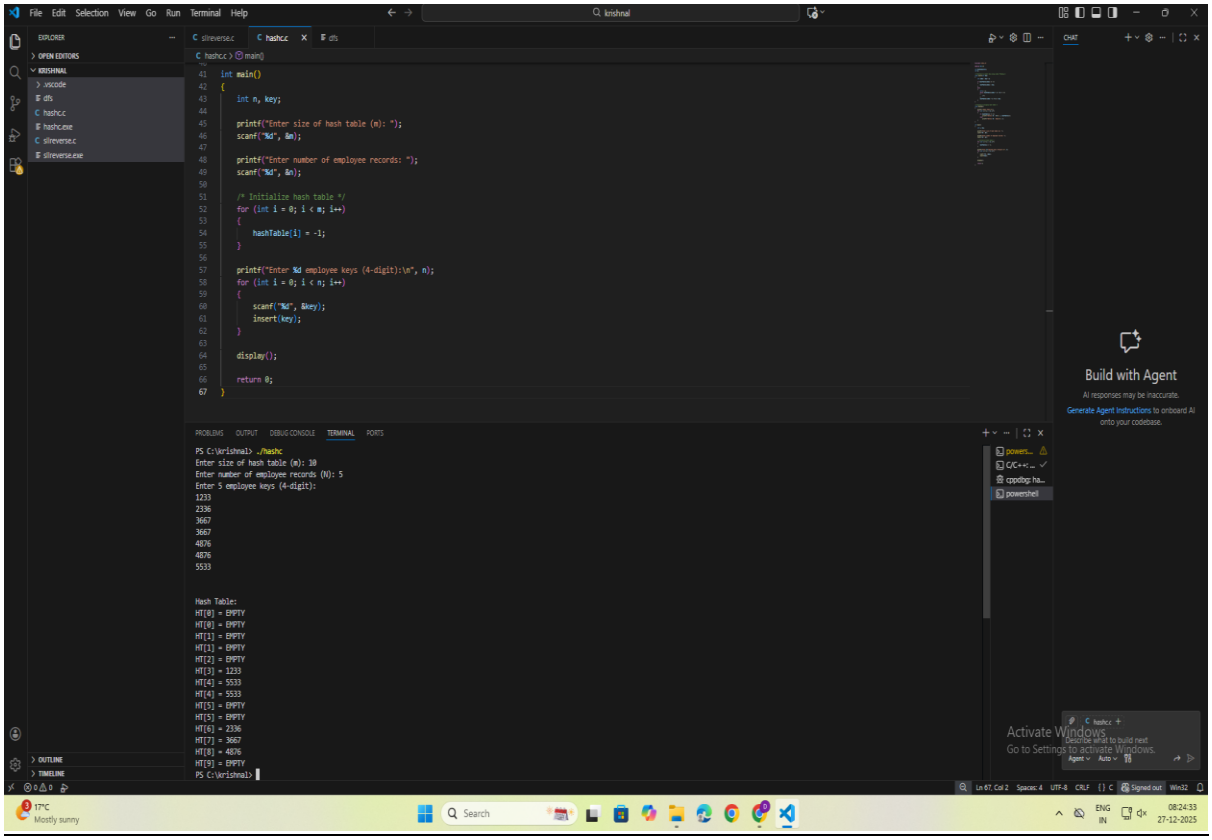
Hash Code


#include <stdio.h>
#define MAX 20
int hashTable[MAX];
int m;
/* Function to insert key using Linear Probing */
void insert(int key)
{
int index = key % m;
if (hashTable[index] == -1)
{
hashTable[index] = key;
}
else
{
int i = 1;
while (hashTable[(index + i) % m] != -1)
{
i++;
}
hashTable[(index + i) % m] = key;
}
}
/* Function to display hash table */
void display()
{
printf("\nHash Table:\n");
for (int i = 0; i < m; i++)
{
if (hashTable[i] != -1)
printf("Address %d : %d\n", i, hashTable[i]);
else
printf("Address %d : Empty\n", i);
}
}
int main()
{
int n, key;
HT[3] = 1233
HT[4] = 5533
Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.


int main()
{
int n, key;
printf("Enter size of hash table (m): ");
scanf("%d", &m);
printf("Enter number of employee records: ");
scanf("%d", &n);
/* Initialize hash table */
for (int i = 0; i < m; i++)
{
hashTable[i] = -1;
}
printf("Enter %d employee keys (4-digit):\n", n);
for (int i = 0; i < n; i++)
{
scanf("%d", &key);
insert(key);
}
display();
return 0;
}
PS C:\krishnal> ./hashc
Enter size of hash table (m): 10
Enter number of employee records (N): 5
Enter 5 employee keys (4-digit):
1233
2336
3667
3667
4876
4876
5533
Hash Table:
HT[0] = EMPTY
HT[0] = EMPTY
HT[1] = EMPTY
HT[1] = EMPTY
HT[2] = EMPTY
HT[3] = 1233
HT[4] = 5533
HT[4] = 5533
HT[5] = EMPTY
HT[5] = EMPTY
HT[6] = 2336
HT[7] = 3667
HT[8] = 4876
HT[9] = EMPTY
PS C:\krishnal>
Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.