Dictionary using hashing

Prathibha. R
1BM18CS074

// Search function.

```
void Dictionary :: search (int key) {
    int flag = 0
    int index = int (key % max);
    temp[index] = root[index];
    while (temp[index] != NULL) {
        if (temp[index] -> data == key) {
            cout << "\n search success";
            flag = 1;
            break;
        }
        else
            temp[index] = temp[index] -> next;
    }
    if (flag == 0)
        cout << "\n search unsuccessful";
}

Dictionary :: Dictionary () {
    index = -1;
    for (int i=0 ; i<max ; i++)
    {
        root[i] = NULL;
        ptr[i] = NULL;
        temp[i] = NULL;
    }
}
```

```
// Insert function.

void Dictionary :: insert (int key)
{
    index = int (key % max)
    ptr [index] = (node type)* malloc (sizeof (node type));
    ptr [index] -> data = key;
    if (root [index] == NULL)
    {
        root [index] = ptr [index];
        root [index] -> next = NULL;
        temp [index] = ptr [index];
    }
    else
    {
        temp [index] = root [index];
        while (temp [index] -> next) = NULL)
            temp [index] = temp [index] -> next;
        temp [index] -> next = ptr [index];
    }
}

// Delete function

void Dictionary :: word delete_ele (int key) {
    index = int (key % max);
    temp [index] = root [index];
    while (temp [index] -> data != key && temp [index] != NULL)
    {
        ptr [index] = temp [index];
        temp [index] = temp [index] -> next; }
    ptr [index] -> next = temp [index] -> next;
    cout << "\n" << temp [index] -> data << "has deleted";
    temp [index] -> data = -1;
    temp [index] = NULL;
    free (temp [index]);
}
```