**Hashing Implementation**

1. #include <stdio.h>
2. #include <string.h>
3. #include <stdlib.h>
5. struct hash \*hashTable= NULL;
6. int eleCount=0;
8. struct node {
9. int key, age;
10. char name[100];
11. struct node \*next;
12. };
14. struct hash {
15. struct node \*head;
16. int count;
17. };
19. struct node \*createNode(int key,char\*name,int age){
20. struct node \*newnode;
21. newnode=(struct node \*)malloc(sizeof(struct node));
22. newnode->key = key;
23. newnode->age = age;
24. strcpy(newnode->name, name);
25. newnode->next = NULL;
26. returnnewnode;
27. }
29. Void insertToHash(int key,char\*name,int age){
30. int hashIndex= key %eleCount;
31. struct node \*newnode=createNode(key, name, age);
32. */\* head of list for the bucket with index "hashIndex" \*/*
33. if(!hashTable[hashIndex].head){
34. hashTable[hashIndex].head=newnode;
35. hashTable[hashIndex].count=1;
36. return;
37. }
38. */\* adding new node to the list \*/*
39. newnode->next =(hashTable[hashIndex].head);
40. */\**
41. *\* update the head of the list and no of*
42. *\* nodes in the current bucket*
43. *\*/*
44. hashTable[hashIndex].head=newnode;
45. hashTable[hashIndex].count++;
46. return;
47. }
49. Void deleteFromHash(int key){
50. */\* find the bucket using hash index \*/*
51. Int hashIndex= key %eleCount, flag =0;
52. struct node \*temp,\*myNode;
53. */\* get the list head from current bucket \*/*
54. myNode=hashTable[hashIndex].head;
55. if(!myNode){
56. printf("Given data is not present in hash Table!!**\n**");
57. return;
58. }
59. temp =myNode;
60. while(myNode!= NULL){
61. */\* delete the node with given key \*/*
62. if(myNode->key == key){
63. flag =1;
64. if(myNode==hashTable[hashIndex].head)
65. hashTable[hashIndex].head=myNode->next;
66. else
67. temp->next =myNode->next;
69. hashTable[hashIndex].count--;
70. free(myNode);
71. **break**;
72. }
73. temp =myNode;
74. myNode=myNode->next;
75. }
76. if(flag)
77. printf("Data deleted successfully from Hash Table**\n**");
78. else
79. printf("Given data is not present in hash Table!!!!**\n**");
80. return;
81. }
83. Void searchInHash(int key){
84. Int hashIndex= key %eleCount, flag =0;
85. struct node \*myNode;
86. myNode=hashTable[hashIndex].head;
87. if(!myNode){
88. printf("Search element unavailable in hash table**\n**");
89. return;
90. }
91. while(myNode!= NULL){
92. if(myNode->key == key){
93. printf("VoterID : %d**\n**",myNode->key);
94. printf("Name : %s**\n**",myNode->name);
95. printf("Age : %d**\n**",myNode->age);
96. flag =1;
97. **break**;
98. }
99. myNode=myNode->next;
100. }
101. if(!flag)
102. printf("Search element unavailable in hash table**\n**");
103. return;
104. }
106. void display(){
107. struct node \*myNode;
108. inti;
109. for(i=0;i<eleCount;i++){
110. if(hashTable[i].count==0)
111. continue;
112. myNode=hashTable[i].head;
113. if(!myNode)
114. continue;
115. printf("**\n**Data at index %d in Hash Table:**\n**",i);
116. printf("VoterID Name Age **\n**");
117. printf("--------------------------------**\n**");
118. while(myNode!= NULL){
119. printf("%-12d",myNode->key);
120. printf("%-15s",myNode->name);
121. printf("%d**\n**",myNode->age);
122. myNode=myNode->next;
123. }
124. }
125. return;
126. }
128. int main(){
129. int n,ch, key, age;
130. char name[100];
131. printf("Enter the number of elements:");
132. scanf("%d",&n);
133. eleCount= n;
134. */\* create hash table with "n" no of buckets \*/*
135. hashTable=(struct hash \*)calloc(n,sizeof(struct hash));
136. while(1){
137. printf("**\n**1. Insertion**\t**2. Deletion**\n**");
138. printf("3. Searching**\t**4. Display**\n**5. Exit**\n**");
139. printf("Enter your choice:");
140. scanf("%d",&ch);
141. switch(ch){
142. case1:
143. printf("Enter the key value:");
144. scanf("%d",&key);
145. getchar();
146. printf("Name:");
147. fgets(name,100,stdin);
148. name[strlen(name)-1]='**\0**';
149. printf("Age:");
150. scanf("%d",&age);
151. */\*inserting new node to hash table \*/*
152. insertToHash(key, name, age);
153. **break**;
155. case2:
156. printf("Enter the key to perform deletion:");
157. scanf("%d",&key);
158. */\* delete node with "key" from hash table \*/*
159. deleteFromHash(key);
160. **break**;
162. case3:
163. printf("Enter the key to search:");
164. scanf("%d",&key);
165. searchInHash(key);
166. **break**;
167. case4:
168. display();
169. **break**;
170. case5:
171. exit(0);
172. default:
173. printf("U have entered wrong option!!**\n**");
174. **break**;
175. }
176. }
177. return0;
178. }

Output:

$ gccHashTablesLL.c

$ ./a.out

Enter the number of elements:3

1. Insertion 2. Deletion

3. Searching 4. Display

5. Exit

Enter your choice:1

Enter the key value:3

Name:Sally

Age:23

1. Insertion 2. Deletion

3. Searching 4. Display

5. Exit

Enter your choice:1

Enter the key value:33

Name:Harry

Age:25

1. Insertion 2. Deletion

3. Searching 4. Display

5. Exit

Enter your choice:1

Enter the key value:7

Name:Nick

Age:30

1. Insertion 2. Deletion

3. Searching 4. Display

5. Exit

Enter your choice:1

Enter the key value:35

Name:Raj

Age:28

1. Insertion 2. Deletion

3. Searching 4. Display

5. Exit

Enter your choice:4

Data at index 0 in Hash Table:

VoterID Name Age

--------------------------------

33 Harry 25

3 Sally 23

Data at index 1 in Hash Table:

VoterID Name Age

--------------------------------

7 Nick 30

Data at index 2 in Hash Table:

VoterID Name Age

--------------------------------

35 Raj 28

1. Insertion 2. Deletion

3. Searching 4. Display

5. Exit

Enter your choice:2

Enter the key to perform deletion:33

Data deleted successfully from Hash Table

1. Insertion 2. Deletion

3. Searching 4. Display

5. Exit

Enter your choice:4

Data at index 0 in Hash Table:

VoterID Name Age

--------------------------------

3 Sally 23

Data at index 1 in Hash Table:

VoterID Name Age

--------------------------------

7 Nick 30

Data at index 2 in Hash Table:

VoterID Name Age

--------------------------------

35 Raj 28

1. Insertion 2. Deletion

3. Searching 4. Display

5. Exit

Enter your choice:3

Enter the key to search:35

VoterID : 35

Name : Raj

Age : 28

1. Insertion 2. Deletion

3. Searching 4. Display

5. Exit

Enter your choice:5