programiz.com

Online C Compiler | jeevanmuchakarla/ds.csa0344-Data-structures

Programiz C Online Compiler

C Certification ›

main.c

Run

```c
#192210211
#include<stdio.h>
#include<stdlib.h>
struct item
{
    int key;
    int value;
};
struct hashtable_item
{

    int flag;
    struct item *data;
};
struct hashtable_item *array;
int size = 0;
int max = 10;
void init_array()
{
    int i;
    for (i = 0; i < max; i++)
    {
    array[i].flag = 0;
    array[i].data = NULL;
    }
}
int hashcode(int key)
{
    return (key % max);
}
```

Output

Clear

```
/tmp/dKzoRx1qjh.o
dKzoRx1qjh.o: malloc.c:2379: sysmalloc: Assertion `(old_top == initial_top (av) &&
    old_size == 0) || ((unsigned long) (old_size) >= MINSIZE && prev_inuse (old_top)
    && ((unsigned long) old_end & (pagesize - 1)) == 0)' failed.
Aborted
```

main.c

Run

```c
void insert(int key, int value)
{
    int index = hashcode(key);
    int i = index;
    struct item *new_item = (struct item*) malloc(sizeof(struct item));
    new_item->key = key;
    new_item->value = value;
    while (array[i].flag == 1)
    {
    if (array[i].data->key == key)
        {
        printf("\n Key already exists, hence updating its value \n");
        array[i].data->value = value;
        return;
    }
    i = (i + 1) % max;
    if (i == index)
        {
        printf("\n Hash table is full, cannot insert any more item \n");
        return;}}
    array[i].flag = 1;
    array[i].data = new_item;
    size++;
    printf("\n Key (%d) has been inserted \n", key);}
void remove_element(int key)
{
    int index = hashcode(key);
    int  i = index;
    while (array[i].flag != 0)
    {
```

Output

Clear

```
/tmp/dKzoRx1qjh.o
dKzoRx1qjh.o: malloc.c:2379: sysmalloc: Assertion `(old_top == initial_top (av) &&
    old_size == 0) || ((unsigned long) (old_size) >= MINSIZE && prev_inuse (old_top)
    && ((unsigned long) old_end & (pagesize - 1)) == 0)' failed.
Aborted
```

main.c

Run

```
60      {
61      if (array[i].flag == 1  &&  array[i].data->key == key )
62          {
63          array[i].flag =  2;
64          array[i].data = NULL;
65          size--;
66          printf("\n Key (%d) has been removed \n", key);
67          return;
68      }
69      i = (i + 1) % max;
70      if (i == index)
71          {
72          break;
73      }
74
75      }
76
77      printf("\n This key does not exist \n");
78
79  }
80  void display()
81  {
82      int i;
83      for (i = 0; i < max; i++)
84      {
85      struct item *current = (struct item*) array[i].data;
86
87      if (current == NULL)
88          {
89          printf("\n Array[%d] has no elements \n", i);
```

Output

Clear

```
/tmp/dKzoRx1qjh.o
dKzoRx1qjh.o: malloc.c:2379: sysmalloc: Assertion `(old_top == initial_top (av) &&
    old_size == 0) || ((unsigned long) (old_size) >= MINSIZE && prev_inuse (old_top)
    && ((unsigned long) old_end & (pagesize - 1)) == 0)' failed.
Aborted
```

main.c

Run

```
89          printf("\n Array[%d] has no elements \n", i);
90      }
91      else
92          {
93          printf("\n Array[%d] has elements -: \n  %d (key) and %d(value) ", i,
                current->key, current->value);
94      }
95      }
96  }
97
98  int size_of_hashtable()
99  {
100     return size;
101 }
102
103 void main()
104 {
105     int choice, key, value, n, c;
106     array = (struct hashtable_item*) malloc(max * sizeof(struct
            hashtable_item*));
107     init_array();
108
109     do {
110         printf("Implementation of Hash Table in C with Linear Probing \n\n");
111         printf("MENU-: \n1.Inserting item in the Hashtable"
112                             "\n2.Removing item from the Hashtable"
113                             "\n3.Check the size of Hashtable"
114                             "\n4.Display Hashtable"
115                 "\n\n Please enter your choice-:");
116
```

Output

Clear

```
/tmp/dKzoRx1qjh.o
dKzoRx1qjh.o: malloc.c:2379: sysmalloc: Assertion `(old_top == initial_top (av) &&
    old_size == 0) || ((unsigned long) (old_size) >= MINSIZE && prev_inuse (old_top)
    && ((unsigned long) old_end & (pagesize - 1)) == 0)' failed.
Aborted
```

main.c

Run

```
117         scanf("%d", &choice);
118
119         switch(choice)
120                 {
121
122         case 1:
123               printf("Inserting element in Hashtable\n");
124               printf("Enter key and value-:\t");
125               scanf("%d %d", &key, &value);
126               insert(key, value);
127               break;
128         case 2:
129               printf("Deleting in Hashtable \n Enter the key to delete-:");
130               scanf("%d", &key);
131               remove_element(key);
132               break;
133         case 3:
134               n = size_of_hashtable();
135               printf("Size of Hashtable is-:%d\n", n);
136               break;
137         case 4:
138               display();
139               break;
140         default:
141                printf("Wrong Input\n");}
142         printf("\n Do you want to continue-:(press 1 for yes)\t");
143         scanf("%d", &c);
144     }while(c == 1);
145   return 0;
146 }
```

Output

Clear

```
/tmp/dKzoRx1qjh.o
dKzoRx1qjh.o: malloc.c:2379: sysmalloc: Assertion `(old_top == initial_top (av) &&
    old_size == 0) || ((unsigned long) (old_size) >= MINSIZE && prev_inuse (old_top)
    && ((unsigned long) old_end & (pagesize - 1)) == 0)' failed.
Aborted
```