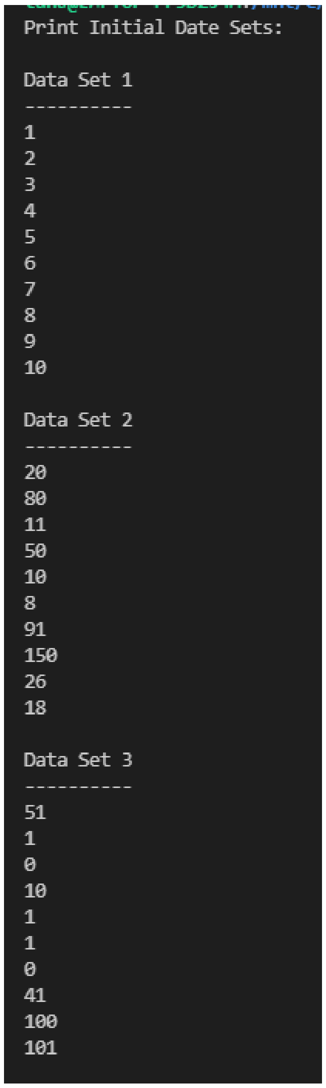
Assignment 5
Comp 2231
T00665076
Moein Sharifi Moghaddam

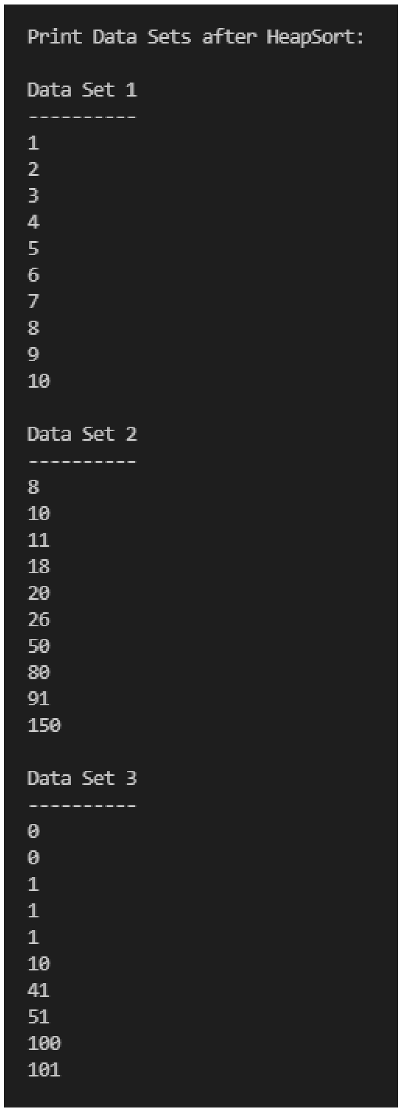
Part 1

a. Data ses
   i. Data set 1 = sorted
   ii. Data set 2 = unsorted
   iii. Data set 3 = unsorted with repeated element



```
Print Initial Date Sets:

Data Set 1
----------
1
2
3
4
5
6
7
8
9
10

Data Set 2
----------
20
80
11
50
10
8
91
150
26
18

Data Set 3
----------
51
1
0
10
1
1
0
41
100
101
```

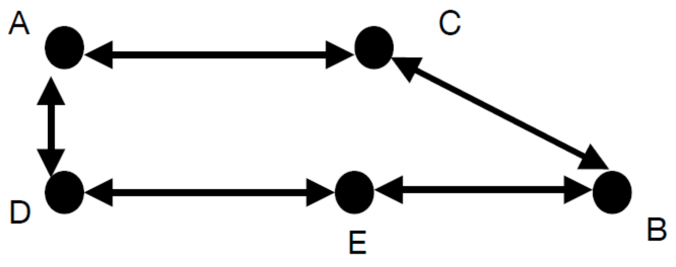
b. Data Sets after HeapSort



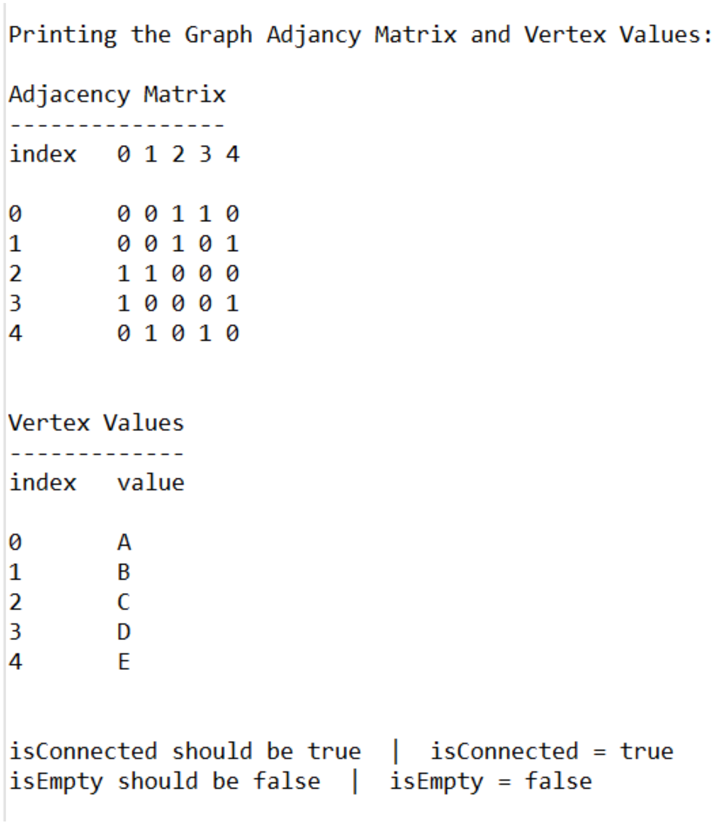
```
Print Data Sets after HeapSort:

Data Set 1
----------
1
2
3
4
5
6
7
8
9
10

Data Set 2
----------
8
10
11
18
20
26
50
80
91
150

Data Set 3
----------
0
0
1
1
1
10
41
51
100
101
```

Part 2

Test Data: Graph [A <-> C <-> B <-> E <-> D <-> A]

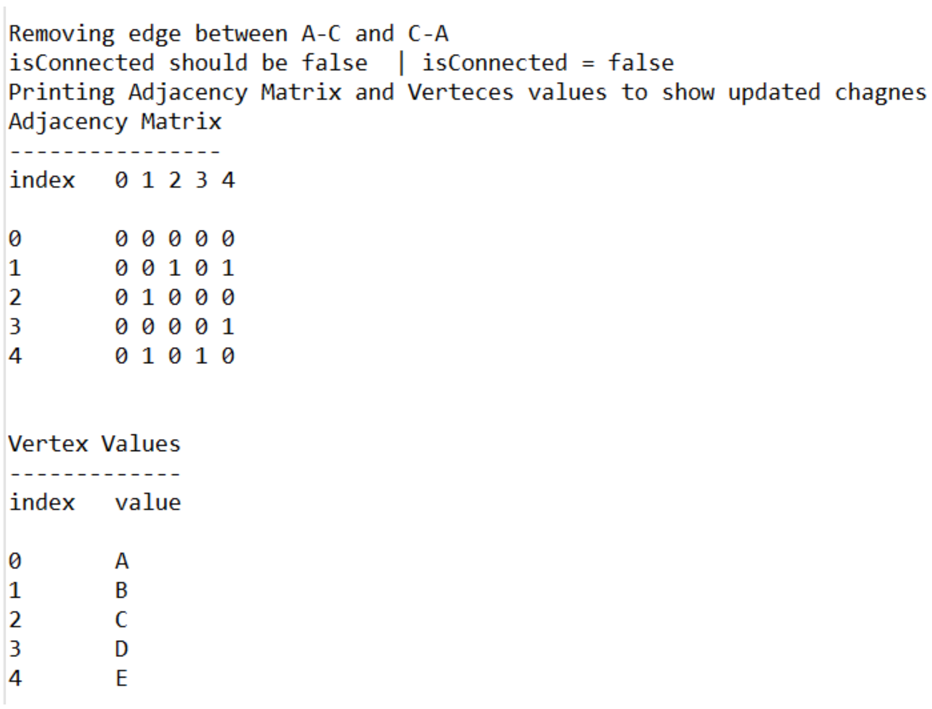


a. Testing Initialization of graph



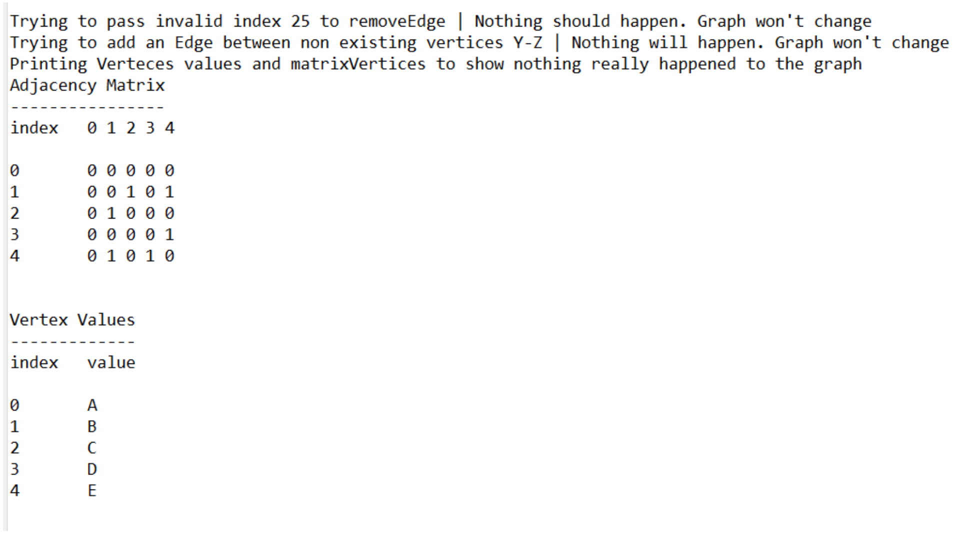
```
Printing the Graph Adjancy Matrix and Vertex Values:

Adjacency Matrix
----------------
index   0 1 2 3 4

0       0 0 1 1 0
1       0 0 1 0 1
2       1 1 0 0 0
3       1 0 0 0 1
4       0 1 0 1 0


Vertex Values
-------------
index   value

0       A
1       B
2       C
3       D
4       E


isConnected should be true  |  isConnected = true
isEmpty should be false  |  isEmpty = false
```

b. Removing some edges to make isConnected false.



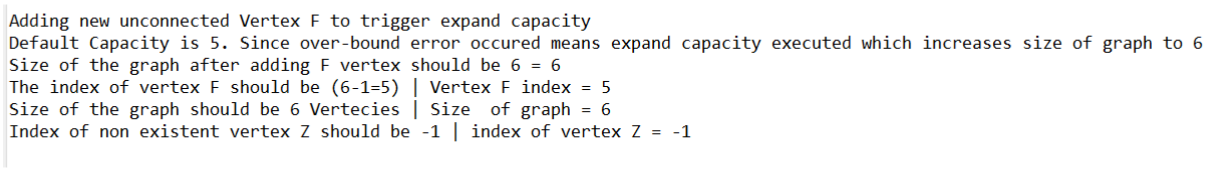
```
Removing edge between A-C and C-A
isConnected should be false  | isConnected = false
Printing Adjacency Matrix and Verteces values to show updated chagnes
Adjacency Matrix
----------------
index   0 1 2 3 4

0       0 0 0 0 0
1       0 0 1 0 1
2       0 1 0 0 0
3       0 0 0 0 1
4       0 1 0 1 0


Vertex Values
-------------
index   value

0       A
1       B
2       C
3       D
4       E
```

c. Removing and adding non-existent edges should result in no change in the graph.



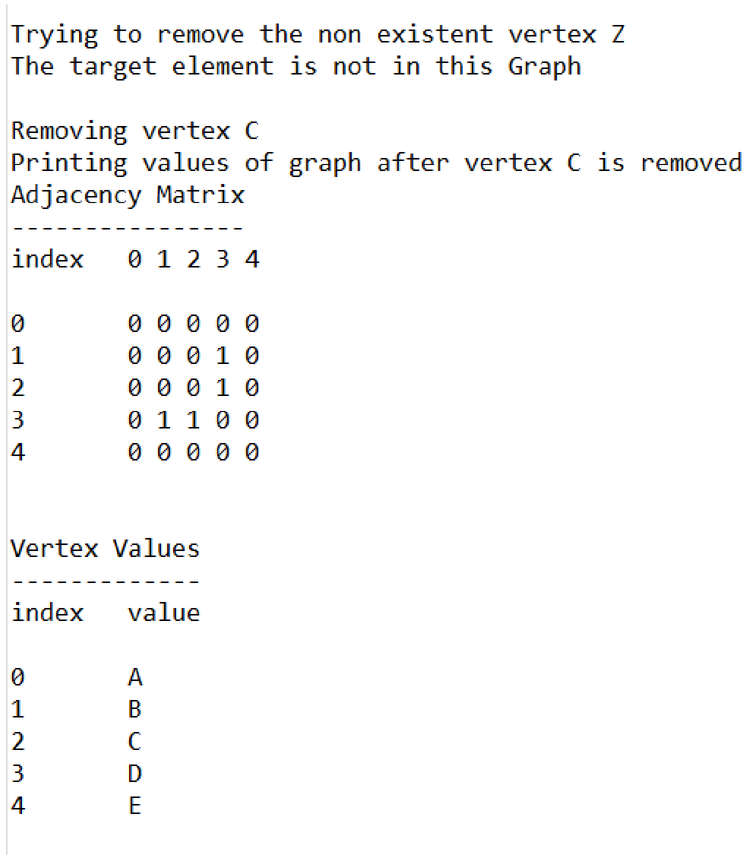
```
Trying to pass invalid index 25 to removeEdge | Nothing should happen. Graph won't change
Trying to add an Edge between non existing vertices Y-Z | Nothing will happen. Graph won't change
Printing Verteces values and matrixVertices to show nothing really happened to the graph
Adjacency Matrix
----------------
index   0 1 2 3 4

0       0 0 0 0 0
1       0 0 1 0 1
2       0 1 0 0 0
3       0 0 0 0 1
4       0 1 0 1 0


Vertex Values
-------------
index   value

0       A
1       B
2       C
3       D
4       E
```

d. Adding a new vertex trigger expands the original capacity of the graph. Then, test index of added vertex, update size and index of a non-existent Z.



```
Adding new unconnected Vertex F to trigger expand capacity
Default Capacity is 5. Since over-bound error occured means expand capacity executed which increases size of graph to 6
Size of the graph after adding F vertex should be 6 = 6
The index of vertex F should be (6-1=5) | Vertex F index = 5
Size of the graph should be 6 Vertecies | Size  of graph = 6
Index of non existent vertex Z should be -1 | index of vertex Z = -1
```

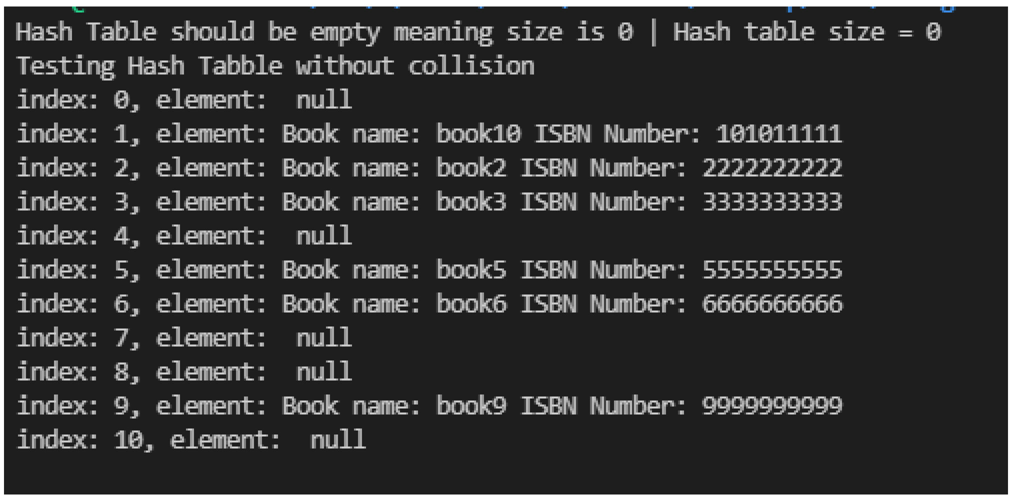
e. Removing non-existent vertex Z results in no change being made to the graph. Removing vertex C which exists in the graph.



```
Trying to remove the non existent vertex Z
The target element is not in this Graph

Removing vertex C
Printing values of graph after vertex C is removed
Adjacency Matrix
----------------
index   0 1 2 3 4

0       0 0 0 0 0
1       0 0 0 1 0
2       0 0 0 1 0
3       0 1 1 0 0
4       0 0 0 0 0


Vertex Values
-------------
index   value

0       A
1       B
2       C
3       D
4       E
```

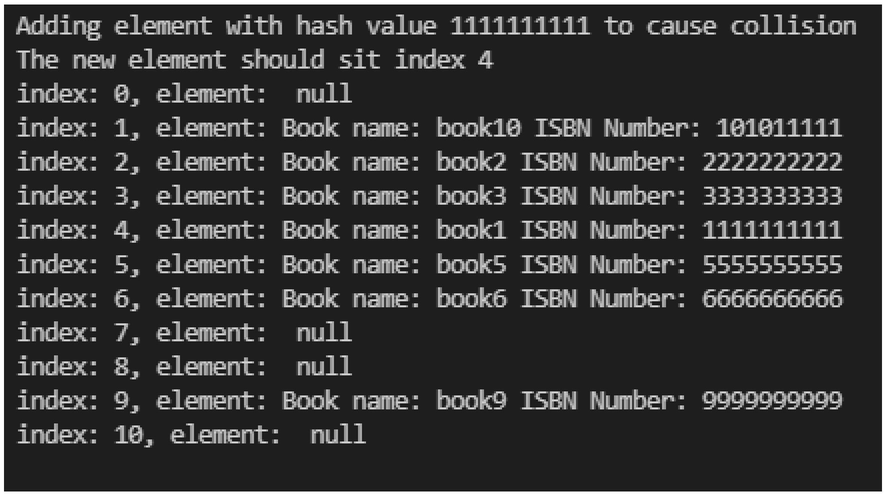
Part 3

a. Testing the size when empty which should be 0. Adding 7 elements with 7 different hash values which causes no collision.



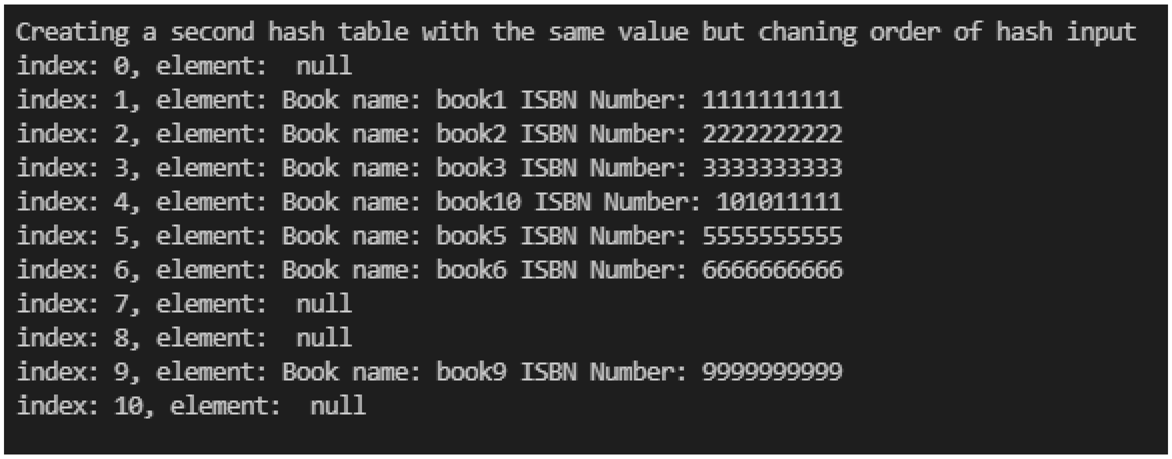
```
Hash Table should be empty meaning size is 0 | Hash table size = 0
Testing Hash Tabble without collision
index: 0, element:  null
index: 1, element: Book name: book10 ISBN Number: 101011111
index: 2, element: Book name: book2 ISBN Number: 2222222222
index: 3, element: Book name: book3 ISBN Number: 3333333333
index: 4, element:  null
index: 5, element: Book name: book5 ISBN Number: 5555555555
index: 6, element: Book name: book6 ISBN Number: 6666666666
index: 7, element:  null
index: 8, element:  null
index: 9, element: Book name: book9 ISBN Number: 9999999999
index: 10, element:  null
```

b. Add a new hash element with ISBN number 1111111111 which yields the same hash value as the element in index 1. This would cause collisions. The program looks for the next index room available. As index 2 and 3 are occupied. The new element with ISBN 1111111111 should sit on index 4.



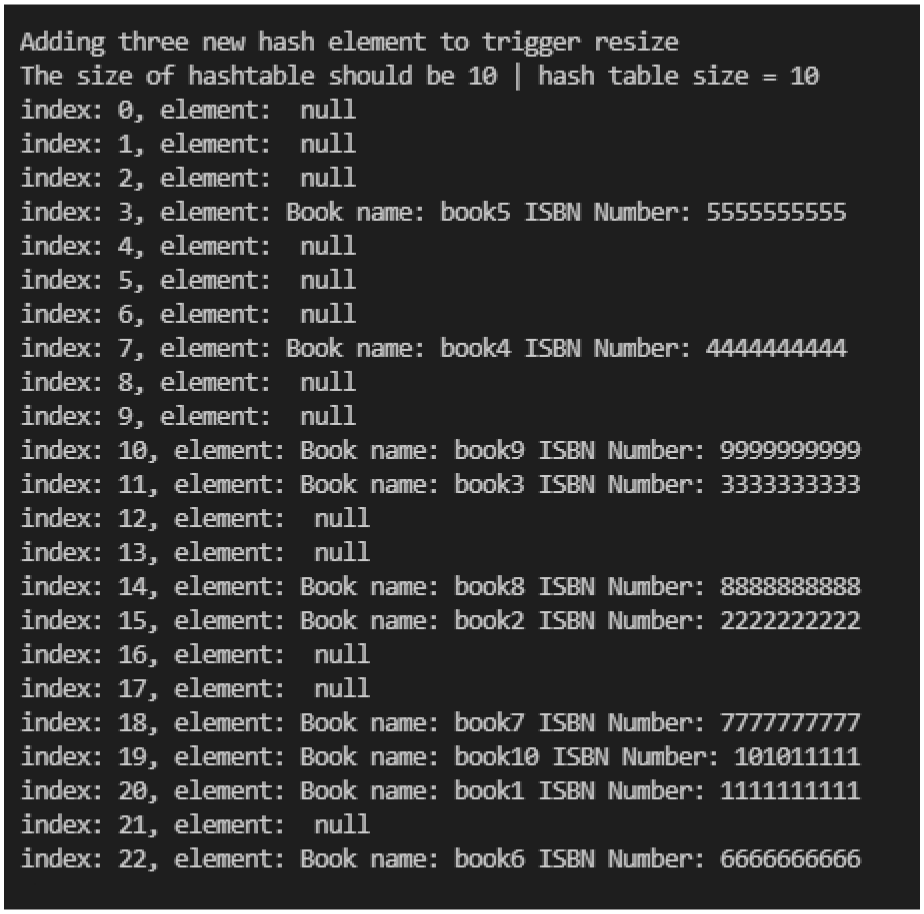
```
Adding element with hash value 1111111111 to cause collision
The new element should sit index 4
index: 0, element:  null
index: 1, element: Book name: book10 ISBN Number: 101011111
index: 2, element: Book name: book2 ISBN Number: 2222222222
index: 3, element: Book name: book3 ISBN Number: 3333333333
index: 4, element: Book name: book1 ISBN Number: 1111111111
index: 5, element: Book name: book5 ISBN Number: 5555555555
index: 6, element: Book name: book6 ISBN Number: 6666666666
index: 7, element:  null
index: 8, element:  null
index: 9, element: Book name: book9 ISBN Number: 9999999999
index: 10, element:  null
```

c. Create a new hash table while adding the very same element but with slightly different order. If you notice, compared to the previous Hash Table the hash element in index 1 and 4 changed position. This shows that the order of adding elements does affect the order of the hash table.



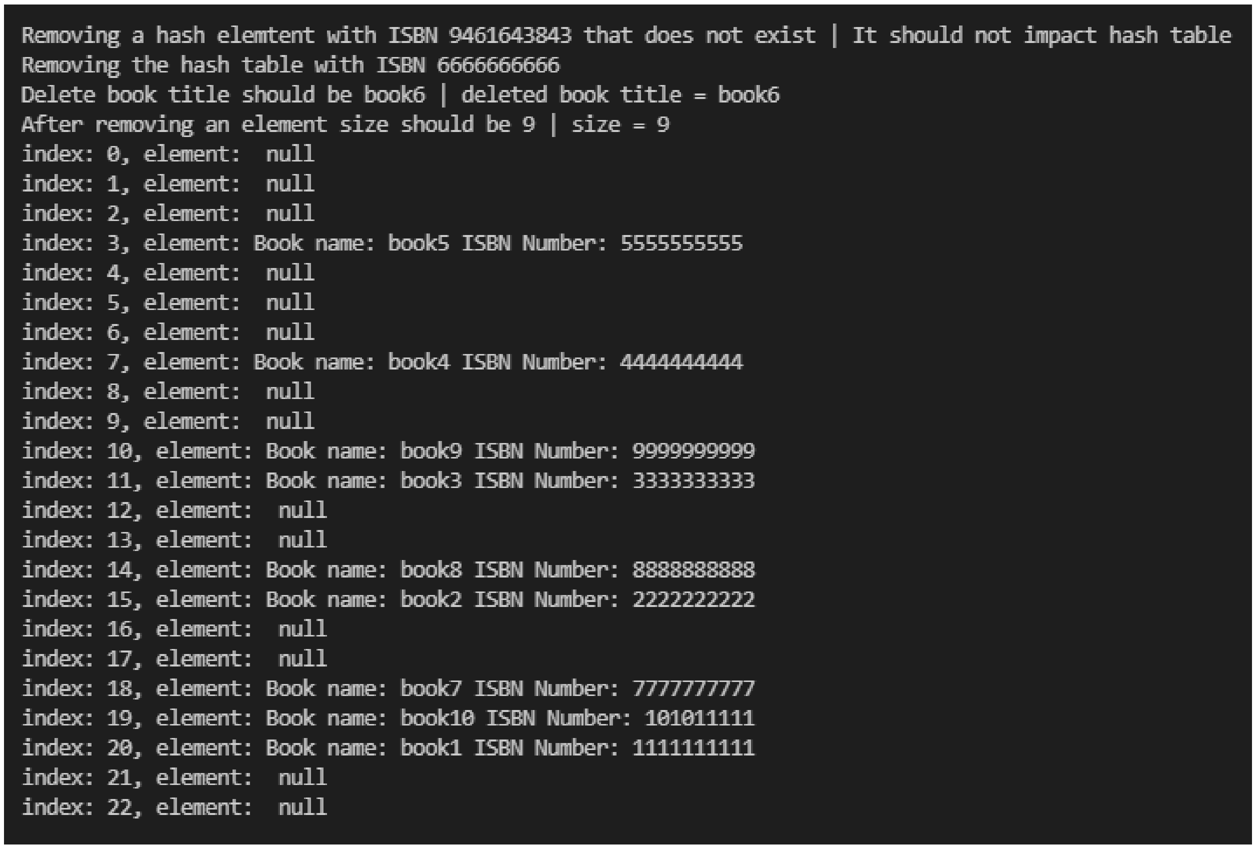
```
Creating a second hash table with the same value but chaning order of hash input
index: 0, element:  null
index: 1, element: Book name: book1 ISBN Number: 1111111111
index: 2, element: Book name: book2 ISBN Number: 2222222222
index: 3, element: Book name: book3 ISBN Number: 3333333333
index: 4, element: Book name: book10 ISBN Number: 101011111
index: 5, element: Book name: book5 ISBN Number: 5555555555
index: 6, element: Book name: book6 ISBN Number: 6666666666
index: 7, element:  null
index: 8, element:  null
index: 9, element: Book name: book9 ISBN Number: 9999999999
index: 10, element:  null
```

d. Adding 3 new hash elements, so it triggers expanded capacity to resize hash table.



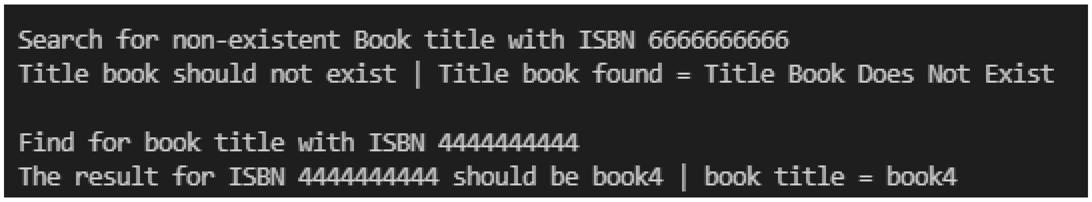
```
Adding three new hash element to trigger resize
The size of hashtable should be 10 | hash table size = 10
index: 0, element:  null
index: 1, element:  null
index: 2, element:  null
index: 3, element: Book name: book5 ISBN Number: 5555555555
index: 4, element:  null
index: 5, element:  null
index: 6, element:  null
index: 7, element: Book name: book4 ISBN Number: 4444444444
index: 8, element:  null
index: 9, element:  null
index: 10, element: Book name: book9 ISBN Number: 9999999999
index: 11, element: Book name: book3 ISBN Number: 3333333333
index: 12, element:  null
index: 13, element:  null
index: 14, element: Book name: book8 ISBN Number: 8888888888
index: 15, element: Book name: book2 ISBN Number: 2222222222
index: 16, element:  null
index: 17, element:  null
index: 18, element: Book name: book7 ISBN Number: 7777777777
index: 19, element: Book name: book10 ISBN Number: 101011111
index: 20, element: Book name: book1 ISBN Number: 1111111111
index: 21, element:  null
index: 22, element: Book name: book6 ISBN Number: 6666666666
```

e. Removing a non-existent ISBN number element and an existent element with ISBN 6666666666.



```
Removing a hash elemtent with ISBN 9461643843 that does not exist | It should not impact hash table
Removing the hash table with ISBN 6666666666
Delete book title should be book6 | deleted book title = book6
After removing an element size should be 9 | size = 9
index: 0, element:  null
index: 1, element:  null
index: 2, element:  null
index: 3, element: Book name: book5 ISBN Number: 5555555555
index: 4, element:  null
index: 5, element:  null
index: 6, element:  null
index: 7, element: Book name: book4 ISBN Number: 4444444444
index: 8, element:  null
index: 9, element:  null
index: 10, element: Book name: book9 ISBN Number: 9999999999
index: 11, element: Book name: book3 ISBN Number: 3333333333
index: 12, element:  null
index: 13, element:  null
index: 14, element: Book name: book8 ISBN Number: 8888888888
index: 15, element: Book name: book2 ISBN Number: 2222222222
index: 16, element:  null
index: 17, element:  null
index: 18, element: Book name: book7 ISBN Number: 7777777777
index: 19, element: Book name: book10 ISBN Number: 101011111
index: 20, element: Book name: book1 ISBN Number: 1111111111
index: 21, element:  null
index: 22, element:  null
```

f. Finding the book title for previously deleted ISBN number 6666666666 and the ISBN number 4444444444 that is present in the hash table database.



```
Search for non-existent Book title with ISBN 6666666666
Title book should not exist | Title book found = Title Book Does Not Exist

Find for book title with ISBN 4444444444
The result for ISBN 4444444444 should be book4 | book title = book4
```