Alexandria University
Faculty of engineering
Computer and Systems Engineering Department
CSE-224: Data Structures & Algorithms

Lab #4

Perfect Hashing

Team members:

ID	NAME
19015267	احمد عاشور السيد محمد سليمان
19017253	احمد حمدي احمد عثمان

Problem Statement

Universal Hashing

A probability distribution H over hash functions from U to {1, ..., M} is universal if for all
x = y in U, we have

P r[h(x) = h(y)] ≤ 1/M

Theorem 1

If H is universal, then for any set S ⊂ U, for any x ∈ U (that we might want to insert or lookup), for a random h taken from H, the expected number of collisions between x and other elements in S is at most N/M.

Constructing a Universal Hash Family: the Matrix Method

Let's say keys are u-bits long. Say the table size M is the power of 2, so an index is b-bits long with M = 2^b. What we'll do is pick h to be a random b-by-u 0/1 matrix, and define h(x) = hx, where we do addition mod 2. For instance:

$$\overset{h}{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}} \overset{x}{\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}} = \overset{h(x)}{\begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}}$$

We can show that for x = y, P r[h(x) = h(y)] = 1/M = 1/2^b

O(N^2)-Space Solution

Say we are willing to have a table whose size is quadratic in the size N of our dictionary S. Then, here is an easy method. Let H be universal and M = N^2. Pick a random h from H and try it out, hashing everything in S. So, we just try it, and if we got any collisions, we just try a new h. On average, we will only need to do this twice.

O(N)-Space Solution

The main idea for this method is to use universal hash functions in a 2-level scheme.
The method is as follows. We will first hash into a table of size N using universal hashing. This will produce some collisions. However, we will then rehash each bin using Method 1, squaring the size of the bin to get zero collisions. So, the way to think of this scheme is that we have a first-level hash function h and first-level table A, and then N second-level hash functions h1, ..., hN and N second-level tables A1, ..., AN. To look-up an element x, we first compute i = h(x) and then find the element in Ai[hi(x)].

You're required to:

1. Implement an O(N^2) as well as an O(N)-Space perfect hash table implemented as described.
2. Verify that the hash table you constructed consumes O(N^2)-space in the quadratic space method and O(N)-space in the linear space method.
3. Report the number of times required to re-build the hash table in the case of collision.
4. Deliver a report describing your implementation details.

Implementation

O(N^2)-Space Hash Table

- The size of the hash table is read from the user.
- The size gets squared and the smallest number power of 2 that covers the desired dictionary size is found to represent the table size.
- The hash table is filled initially with Integer.MIN_VALUE, and it is assumed that it won't be inserted. The hashing function is represented by a matrix of 0s and 1s, and it is generated randomly using the setMatrix() method.
- On every collision, the rehash() method gets called and the hash function is changed, and the function loops through the table and hashes every entry using the new hash function.

O(N)-Space Hash Table

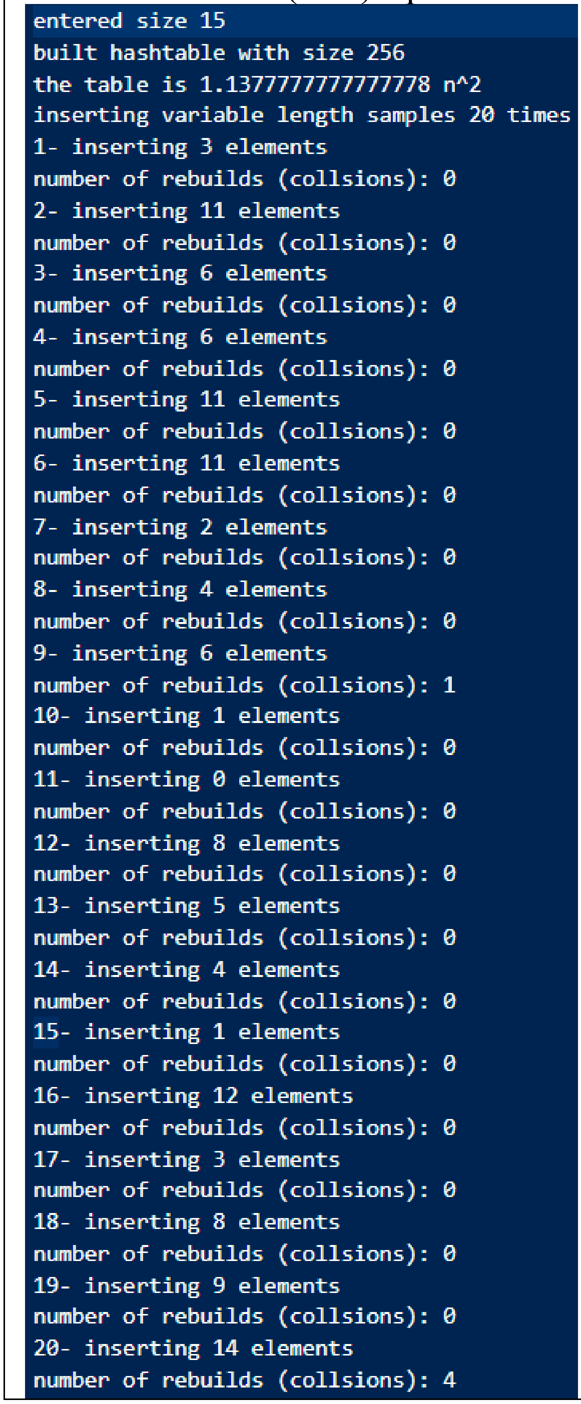
- The size of the hash table is read from the user.
- The smallest number power of 2 that covers the desired dictionary size is found to represent the table size (The Dictionary is represented as **Buckets**).
- The Buckets are initially constructed with zero size.
- Each Bucket has a list to maintain the elements, that are added to it, to be hashed at the end of insertions using O(N^2) – space hash table.
- The 1st–level hash function (h) is generated randomly using the setMatrix() method.
- On every collision at level 1, the elements are kept in that collided Bucket.
- At the end of the insertions, we check if (h) gives $\sum_i n_i^2 < 4n$;
  where: n is The memory needed for the 1st level table;
  n_i are the elements that hashed to the Bucket i.
  - When the condition holds, we invoke the function hashBuckets() which loops through each non–empty Bucket and hash its elements using Method 1.
  - When it doesn't hold, we invoke the function rehash() which will choose a new hash function (h') & rebuilds the hash table from the beginning.

Test cases

O(N^2)-Space Test cases

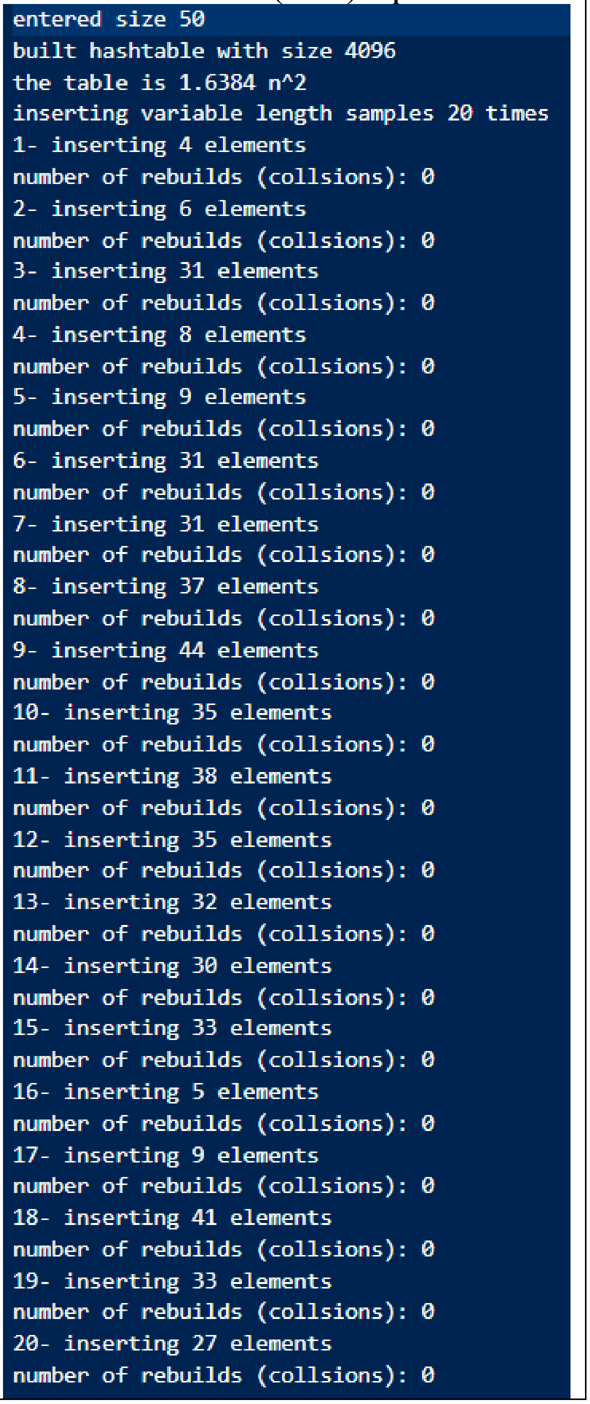
These tests Verify that the space required is O(N^2) & show the collisions number.

Test 1 – O(N^2)-Space



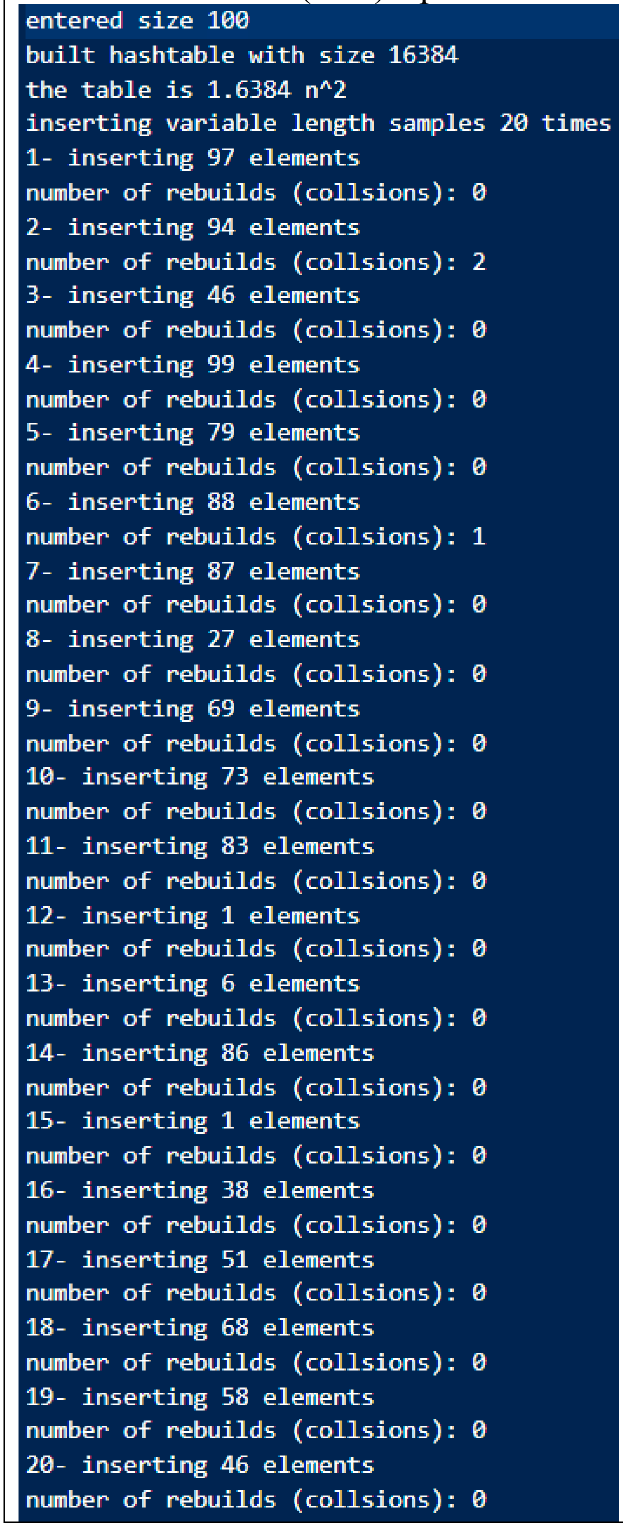
```
entered size 15
built hashtable with size 256
the table is 1.1377777777777778 n^2
inserting variable length samples 20 times
1- inserting 3 elements
number of rebuilds (collsions): 0
2- inserting 11 elements
number of rebuilds (collsions): 0
3- inserting 6 elements
number of rebuilds (collsions): 0
4- inserting 6 elements
number of rebuilds (collsions): 0
5- inserting 11 elements
number of rebuilds (collsions): 0
6- inserting 11 elements
number of rebuilds (collsions): 0
7- inserting 2 elements
number of rebuilds (collsions): 0
8- inserting 4 elements
number of rebuilds (collsions): 0
9- inserting 6 elements
number of rebuilds (collsions): 1
10- inserting 1 elements
number of rebuilds (collsions): 0
11- inserting 0 elements
number of rebuilds (collsions): 0
12- inserting 8 elements
number of rebuilds (collsions): 0
13- inserting 5 elements
number of rebuilds (collsions): 0
14- inserting 4 elements
number of rebuilds (collsions): 0
15- inserting 1 elements
number of rebuilds (collsions): 0
16- inserting 12 elements
number of rebuilds (collsions): 0
17- inserting 3 elements
number of rebuilds (collsions): 0
18- inserting 8 elements
number of rebuilds (collsions): 0
19- inserting 9 elements
number of rebuilds (collsions): 0
20- inserting 14 elements
number of rebuilds (collsions): 4
```

Test 2 – O(N^2)-Space



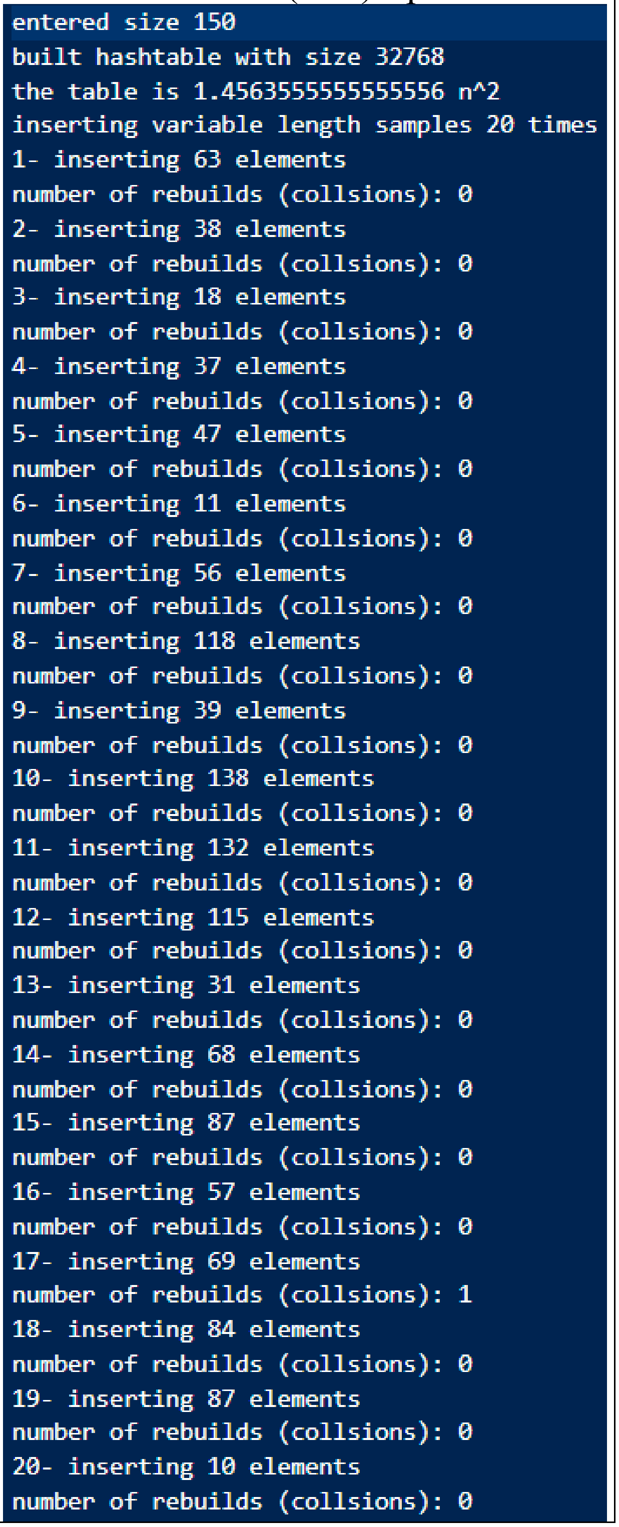
```
entered size 50
built hashtable with size 4096
the table is 1.6384 n^2
inserting variable length samples 20 times
1- inserting 4 elements
number of rebuilds (collsions): 0
2- inserting 6 elements
number of rebuilds (collsions): 0
3- inserting 31 elements
number of rebuilds (collsions): 0
4- inserting 8 elements
number of rebuilds (collsions): 0
5- inserting 9 elements
number of rebuilds (collsions): 0
6- inserting 31 elements
number of rebuilds (collsions): 0
7- inserting 31 elements
number of rebuilds (collsions): 0
8- inserting 37 elements
number of rebuilds (collsions): 0
9- inserting 44 elements
number of rebuilds (collsions): 0
10- inserting 35 elements
number of rebuilds (collsions): 0
11- inserting 38 elements
number of rebuilds (collsions): 0
12- inserting 35 elements
number of rebuilds (collsions): 0
13- inserting 32 elements
number of rebuilds (collsions): 0
14- inserting 30 elements
number of rebuilds (collsions): 0
15- inserting 33 elements
number of rebuilds (collsions): 0
16- inserting 5 elements
number of rebuilds (collsions): 0
17- inserting 9 elements
number of rebuilds (collsions): 0
18- inserting 41 elements
number of rebuilds (collsions): 0
19- inserting 33 elements
number of rebuilds (collsions): 0
20- inserting 27 elements
number of rebuilds (collsions): 0
```

Test 3 – O(N^2)-Space



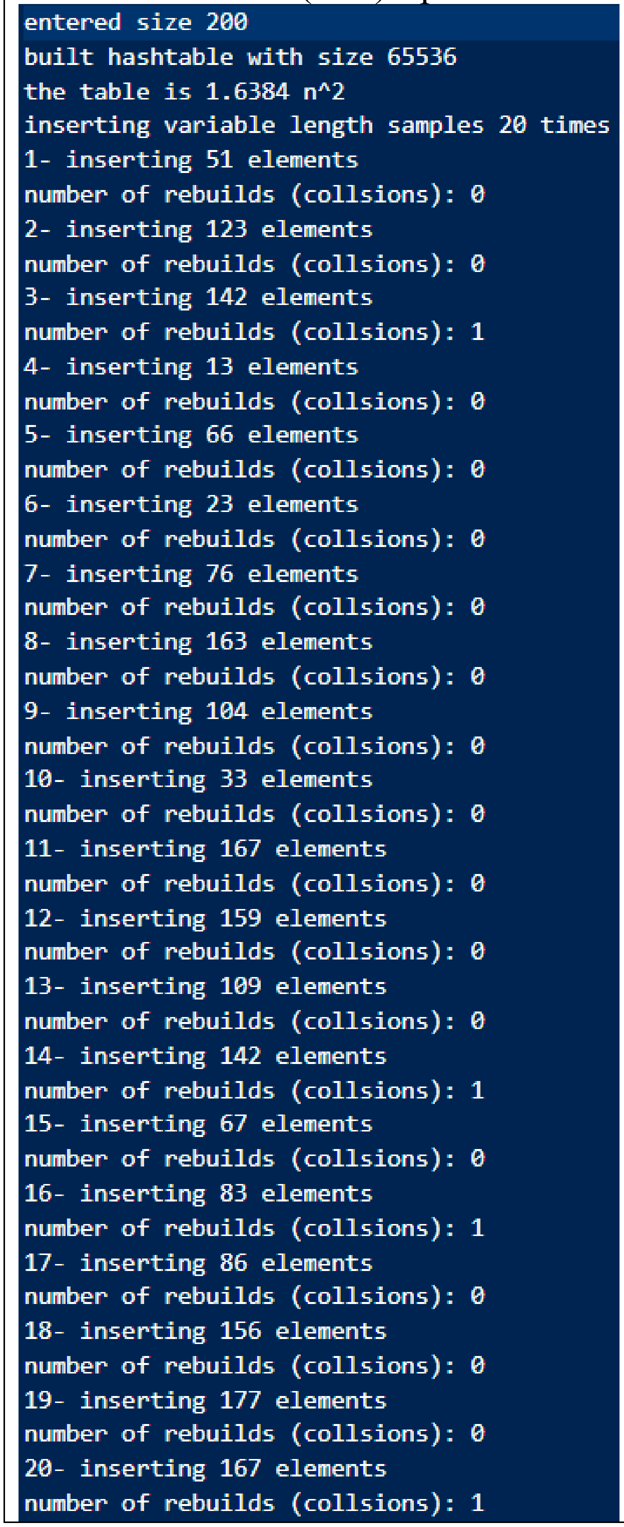
```
entered size 100
built hashtable with size 16384
the table is 1.6384 n^2
inserting variable length samples 20 times
1- inserting 97 elements
number of rebuilds (collsions): 0
2- inserting 94 elements
number of rebuilds (collsions): 2
3- inserting 46 elements
number of rebuilds (collsions): 0
4- inserting 99 elements
number of rebuilds (collsions): 0
5- inserting 79 elements
number of rebuilds (collsions): 0
6- inserting 88 elements
number of rebuilds (collsions): 1
7- inserting 87 elements
number of rebuilds (collsions): 0
8- inserting 27 elements
number of rebuilds (collsions): 0
9- inserting 69 elements
number of rebuilds (collsions): 0
10- inserting 73 elements
number of rebuilds (collsions): 0
11- inserting 83 elements
number of rebuilds (collsions): 0
12- inserting 1 elements
number of rebuilds (collsions): 0
13- inserting 6 elements
number of rebuilds (collsions): 0
14- inserting 86 elements
number of rebuilds (collsions): 0
15- inserting 1 elements
number of rebuilds (collsions): 0
16- inserting 38 elements
number of rebuilds (collsions): 0
17- inserting 51 elements
number of rebuilds (collsions): 0
18- inserting 68 elements
number of rebuilds (collsions): 0
19- inserting 58 elements
number of rebuilds (collsions): 0
20- inserting 46 elements
number of rebuilds (collsions): 0
```

Test 4 – O(N^2)-Space



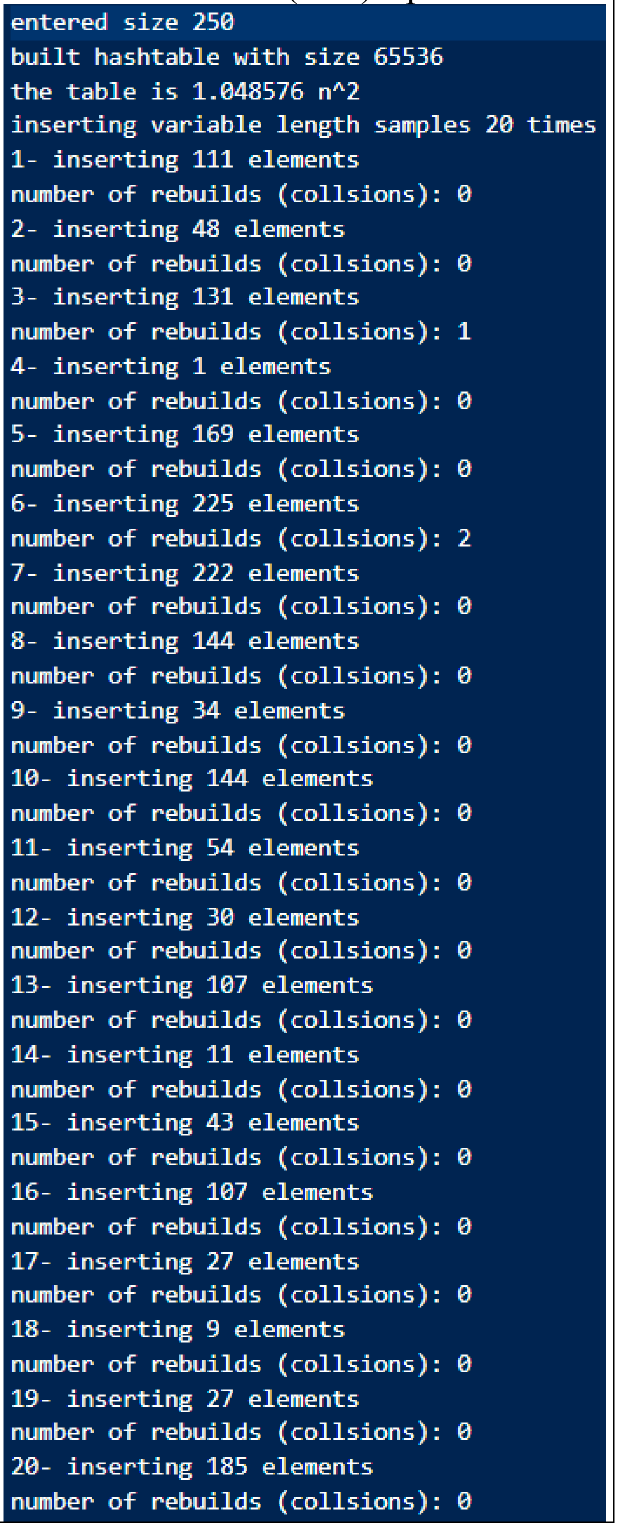
```
entered size 150
built hashtable with size 32768
the table is 1.4563555555555556 n^2
inserting variable length samples 20 times
1- inserting 63 elements
number of rebuilds (collsions): 0
2- inserting 38 elements
number of rebuilds (collsions): 0
3- inserting 18 elements
number of rebuilds (collsions): 0
4- inserting 37 elements
number of rebuilds (collsions): 0
5- inserting 47 elements
number of rebuilds (collsions): 0
6- inserting 11 elements
number of rebuilds (collsions): 0
7- inserting 56 elements
number of rebuilds (collsions): 0
8- inserting 118 elements
number of rebuilds (collsions): 0
9- inserting 39 elements
number of rebuilds (collsions): 0
10- inserting 138 elements
number of rebuilds (collsions): 0
11- inserting 132 elements
number of rebuilds (collsions): 0
12- inserting 115 elements
number of rebuilds (collsions): 0
13- inserting 31 elements
number of rebuilds (collsions): 0
14- inserting 68 elements
number of rebuilds (collsions): 0
15- inserting 87 elements
number of rebuilds (collsions): 0
16- inserting 57 elements
number of rebuilds (collsions): 0
17- inserting 69 elements
number of rebuilds (collsions): 1
18- inserting 84 elements
number of rebuilds (collsions): 0
19- inserting 87 elements
number of rebuilds (collsions): 0
20- inserting 10 elements
number of rebuilds (collsions): 0
```

Test 5 – O(N^2)-Space



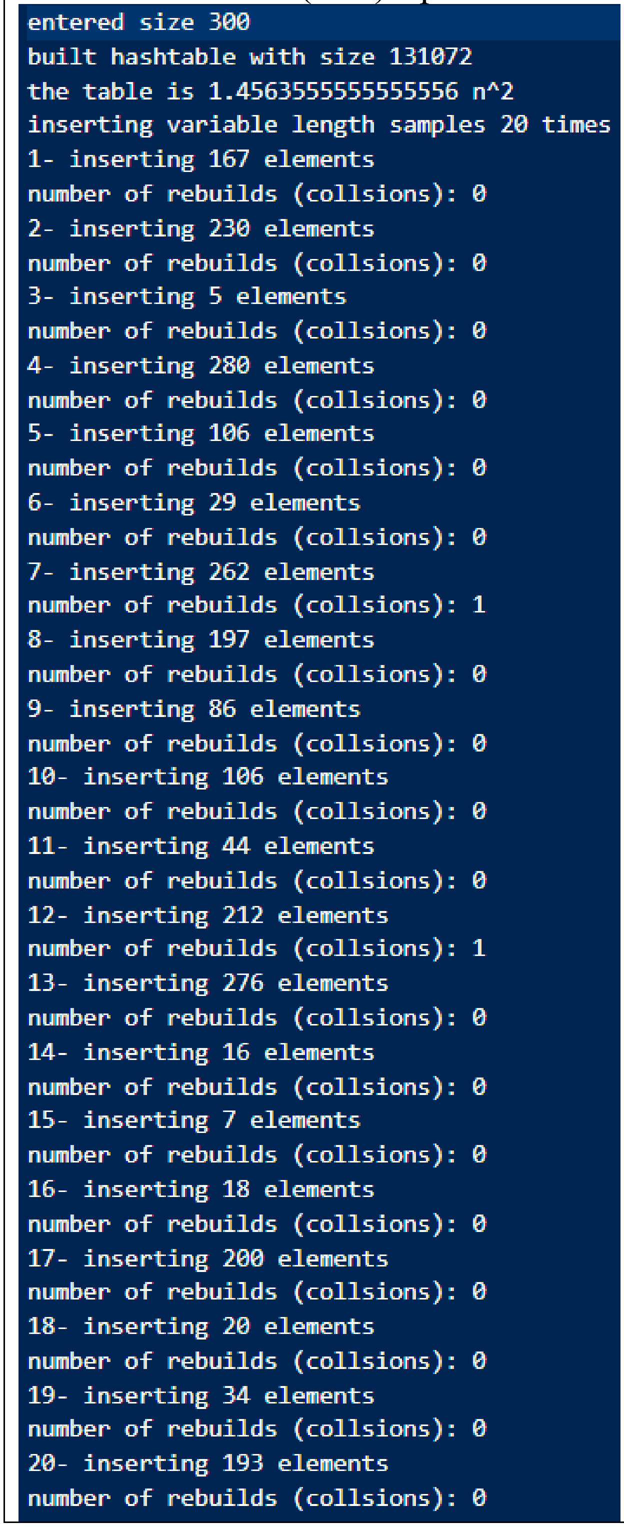
```
entered size 200
built hashtable with size 65536
the table is 1.6384 n^2
inserting variable length samples 20 times
1- inserting 51 elements
number of rebuilds (collsions): 0
2- inserting 123 elements
number of rebuilds (collsions): 0
3- inserting 142 elements
number of rebuilds (collsions): 1
4- inserting 13 elements
number of rebuilds (collsions): 0
5- inserting 66 elements
number of rebuilds (collsions): 0
6- inserting 23 elements
number of rebuilds (collsions): 0
7- inserting 76 elements
number of rebuilds (collsions): 0
8- inserting 163 elements
number of rebuilds (collsions): 0
9- inserting 104 elements
number of rebuilds (collsions): 0
10- inserting 33 elements
number of rebuilds (collsions): 0
11- inserting 167 elements
number of rebuilds (collsions): 0
12- inserting 159 elements
number of rebuilds (collsions): 0
13- inserting 109 elements
number of rebuilds (collsions): 0
14- inserting 142 elements
number of rebuilds (collsions): 1
15- inserting 67 elements
number of rebuilds (collsions): 0
16- inserting 83 elements
number of rebuilds (collsions): 1
17- inserting 86 elements
number of rebuilds (collsions): 0
18- inserting 156 elements
number of rebuilds (collsions): 0
19- inserting 177 elements
number of rebuilds (collsions): 0
20- inserting 167 elements
number of rebuilds (collsions): 1
```

Test 6 – O(N^2)-Space



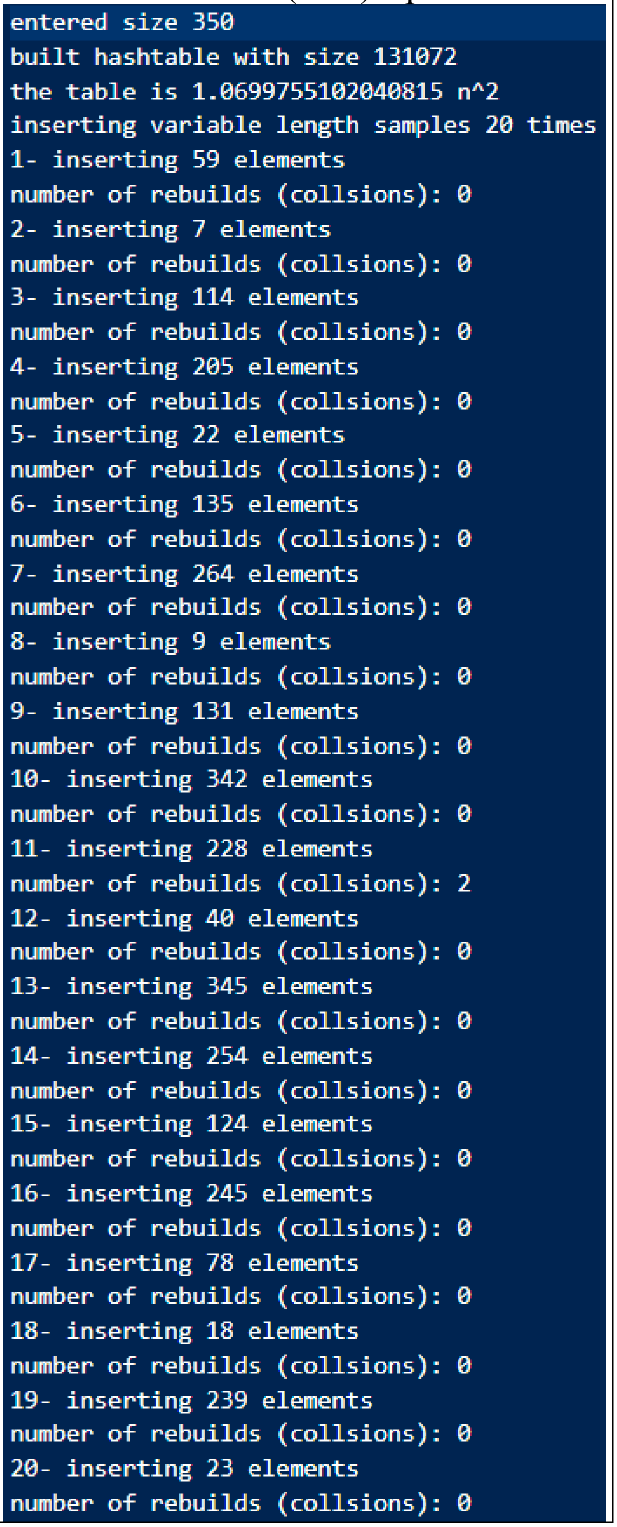
```
entered size 250
built hashtable with size 65536
the table is 1.048576 n^2
inserting variable length samples 20 times
1- inserting 111 elements
number of rebuilds (collsions): 0
2- inserting 48 elements
number of rebuilds (collsions): 0
3- inserting 131 elements
number of rebuilds (collsions): 1
4- inserting 1 elements
number of rebuilds (collsions): 0
5- inserting 169 elements
number of rebuilds (collsions): 0
6- inserting 225 elements
number of rebuilds (collsions): 2
7- inserting 222 elements
number of rebuilds (collsions): 0
8- inserting 144 elements
number of rebuilds (collsions): 0
9- inserting 34 elements
number of rebuilds (collsions): 0
10- inserting 144 elements
number of rebuilds (collsions): 0
11- inserting 54 elements
number of rebuilds (collsions): 0
12- inserting 30 elements
number of rebuilds (collsions): 0
13- inserting 107 elements
number of rebuilds (collsions): 0
14- inserting 11 elements
number of rebuilds (collsions): 0
15- inserting 43 elements
number of rebuilds (collsions): 0
16- inserting 107 elements
number of rebuilds (collsions): 0
17- inserting 27 elements
number of rebuilds (collsions): 0
18- inserting 9 elements
number of rebuilds (collsions): 0
19- inserting 27 elements
number of rebuilds (collsions): 0
20- inserting 185 elements
number of rebuilds (collsions): 0
```

Test 7 – O(N^2)-Space



```
entered size 300
built hashtable with size 131072
the table is 1.4563555555555556 n^2
inserting variable length samples 20 times
1- inserting 167 elements
number of rebuilds (collsions): 0
2- inserting 230 elements
number of rebuilds (collsions): 0
3- inserting 5 elements
number of rebuilds (collsions): 0
4- inserting 280 elements
number of rebuilds (collsions): 0
5- inserting 106 elements
number of rebuilds (collsions): 0
6- inserting 29 elements
number of rebuilds (collsions): 0
7- inserting 262 elements
number of rebuilds (collsions): 1
8- inserting 197 elements
number of rebuilds (collsions): 0
9- inserting 86 elements
number of rebuilds (collsions): 0
10- inserting 106 elements
number of rebuilds (collsions): 0
11- inserting 44 elements
number of rebuilds (collsions): 0
12- inserting 212 elements
number of rebuilds (collsions): 1
13- inserting 276 elements
number of rebuilds (collsions): 0
14- inserting 16 elements
number of rebuilds (collsions): 0
15- inserting 7 elements
number of rebuilds (collsions): 0
16- inserting 18 elements
number of rebuilds (collsions): 0
17- inserting 200 elements
number of rebuilds (collsions): 0
18- inserting 20 elements
number of rebuilds (collsions): 0
19- inserting 34 elements
number of rebuilds (collsions): 0
20- inserting 193 elements
number of rebuilds (collsions): 0
```

Test 8 – O(N^2)-Space

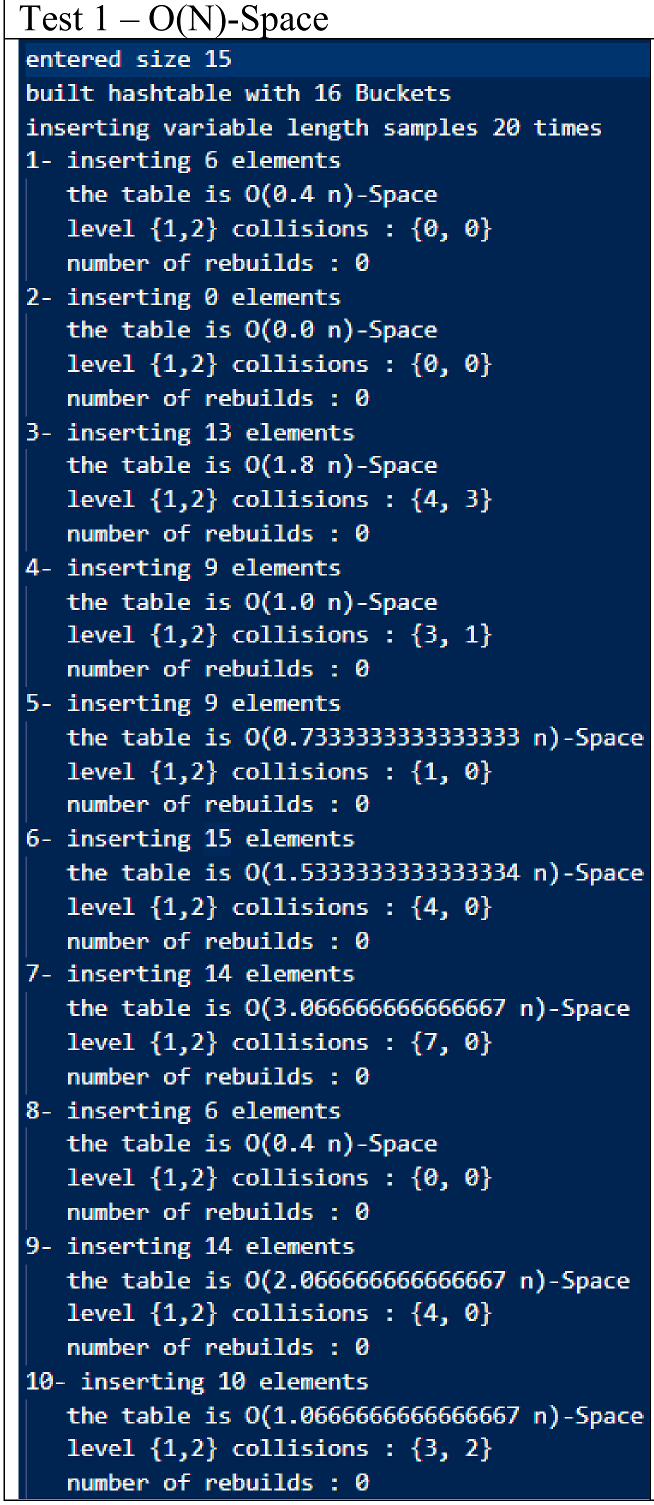


```
entered size 350
built hashtable with size 131072
the table is 1.0699755102040815 n^2
inserting variable length samples 20 times
1- inserting 59 elements
number of rebuilds (collsions): 0
2- inserting 7 elements
number of rebuilds (collsions): 0
3- inserting 114 elements
number of rebuilds (collsions): 0
4- inserting 205 elements
number of rebuilds (collsions): 0
5- inserting 22 elements
number of rebuilds (collsions): 0
6- inserting 135 elements
number of rebuilds (collsions): 0
7- inserting 264 elements
number of rebuilds (collsions): 0
8- inserting 9 elements
number of rebuilds (collsions): 0
9- inserting 131 elements
number of rebuilds (collsions): 0
10- inserting 342 elements
number of rebuilds (collsions): 0
11- inserting 228 elements
number of rebuilds (collsions): 2
12- inserting 40 elements
number of rebuilds (collsions): 0
13- inserting 345 elements
number of rebuilds (collsions): 0
14- inserting 254 elements
number of rebuilds (collsions): 0
15- inserting 124 elements
number of rebuilds (collsions): 0
16- inserting 245 elements
number of rebuilds (collsions): 0
17- inserting 78 elements
number of rebuilds (collsions): 0
18- inserting 18 elements
number of rebuilds (collsions): 0
19- inserting 239 elements
number of rebuilds (collsions): 0
20- inserting 23 elements
number of rebuilds (collsions): 0
```

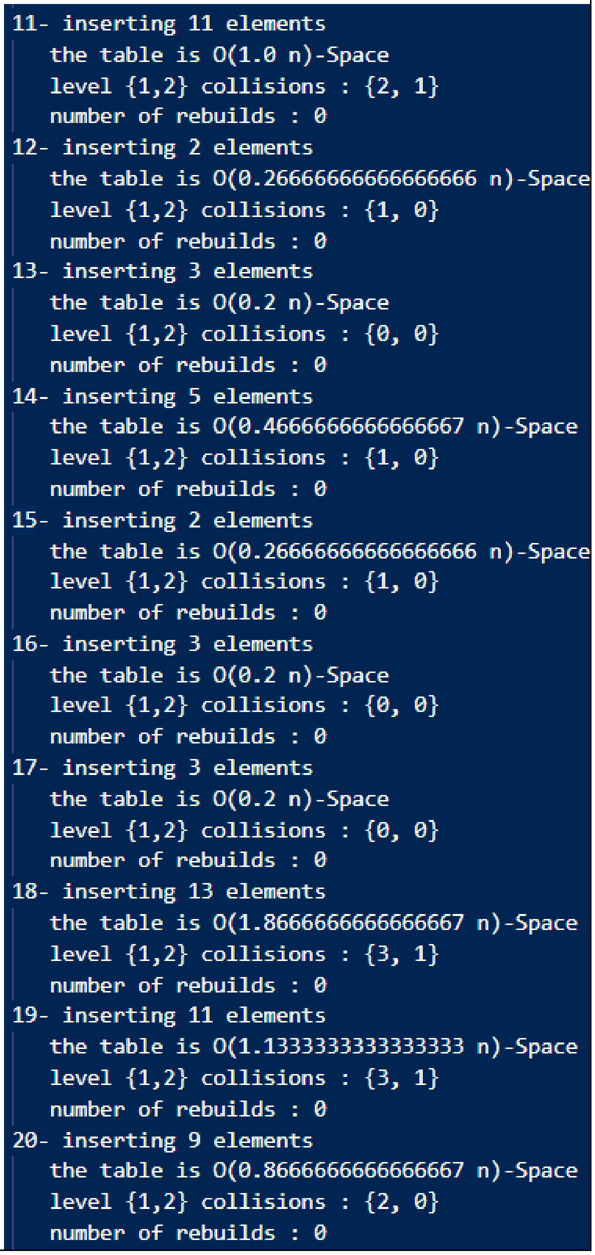
O(N)-Space Test cases

These tests verify that, for each sample, the space required is O(N) & show the collisions that happened in the 2 levels and the number of rebuilds.

Test 1 – O(N)-Space

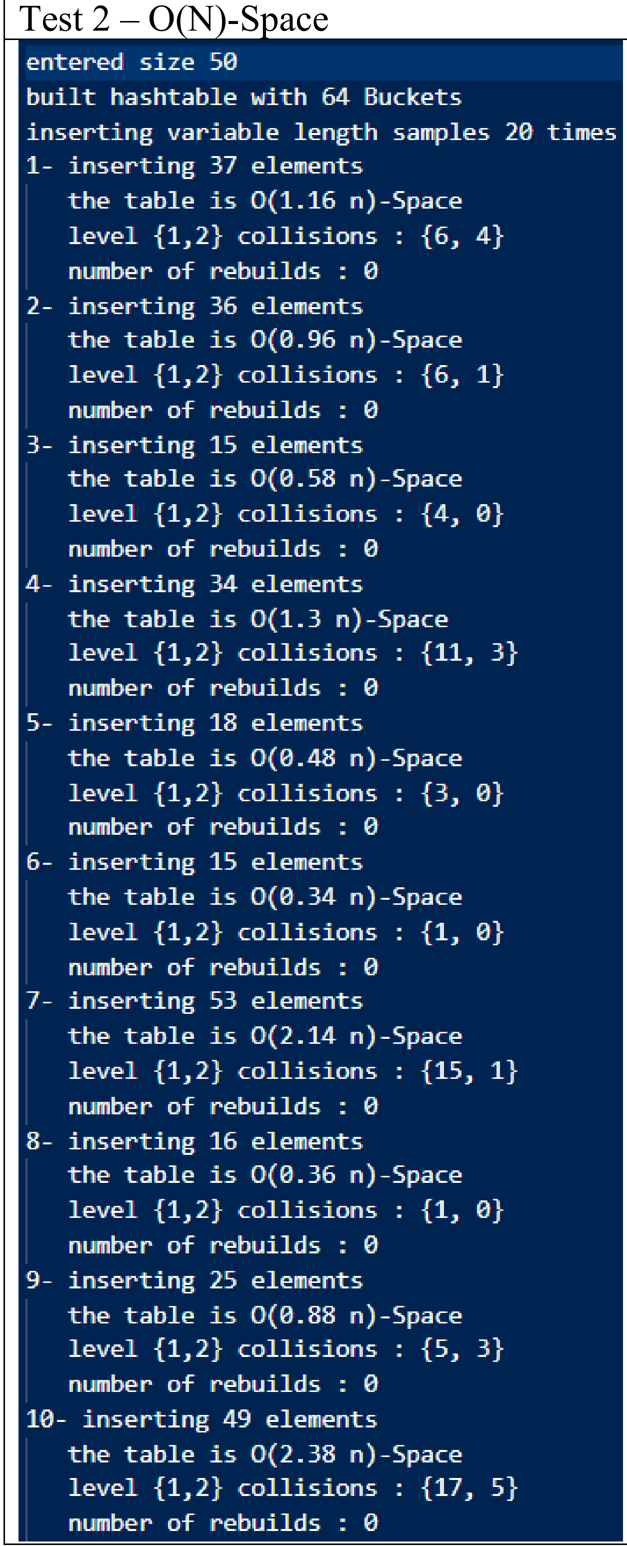


```
entered size 15
built hashtable with 16 Buckets
inserting variable length samples 20 times
1- inserting 6 elements
   the table is O(0.4 n)-Space
   level {1,2} collisions : {0, 0}
   number of rebuilds : 0
2- inserting 0 elements
   the table is O(0.0 n)-Space
   level {1,2} collisions : {0, 0}
   number of rebuilds : 0
3- inserting 13 elements
   the table is O(1.8 n)-Space
   level {1,2} collisions : {4, 3}
   number of rebuilds : 0
4- inserting 9 elements
   the table is O(1.0 n)-Space
   level {1,2} collisions : {3, 1}
   number of rebuilds : 0
5- inserting 9 elements
   the table is O(0.7333333333333333 n)-Space
   level {1,2} collisions : {1, 0}
   number of rebuilds : 0
6- inserting 15 elements
   the table is O(1.5333333333333334 n)-Space
   level {1,2} collisions : {4, 0}
   number of rebuilds : 0
7- inserting 14 elements
   the table is O(3.066666666666667 n)-Space
   level {1,2} collisions : {7, 0}
   number of rebuilds : 0
8- inserting 6 elements
   the table is O(0.4 n)-Space
   level {1,2} collisions : {0, 0}
   number of rebuilds : 0
9- inserting 14 elements
   the table is O(2.066666666666667 n)-Space
   level {1,2} collisions : {4, 0}
   number of rebuilds : 0
10- inserting 10 elements
   the table is O(1.0666666666666667 n)-Space
   level {1,2} collisions : {3, 2}
   number of rebuilds : 0
```

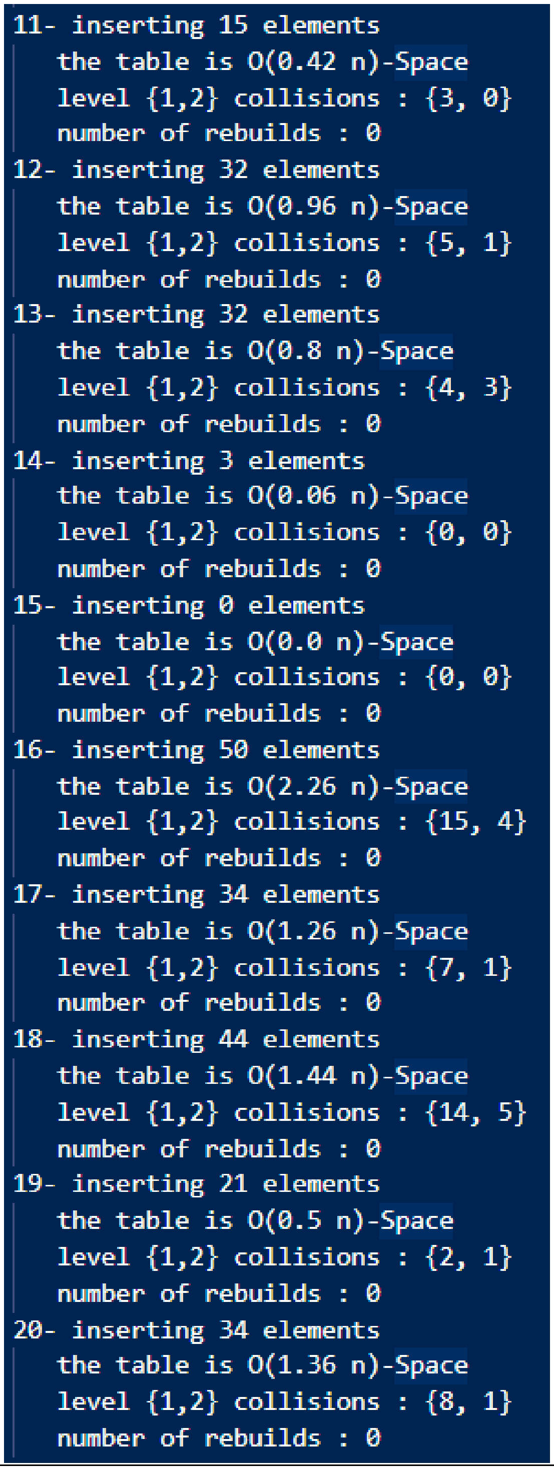


```
11- inserting 11 elements
   the table is O(1.0 n)-Space
   level {1,2} collisions : {2, 1}
   number of rebuilds : 0
12- inserting 2 elements
   the table is O(0.26666666666666666 n)-Space
   level {1,2} collisions : {1, 0}
   number of rebuilds : 0
13- inserting 3 elements
   the table is O(0.2 n)-Space
   level {1,2} collisions : {0, 0}
   number of rebuilds : 0
14- inserting 5 elements
   the table is O(0.4666666666666667 n)-Space
   level {1,2} collisions : {1, 0}
   number of rebuilds : 0
15- inserting 2 elements
   the table is O(0.26666666666666666 n)-Space
   level {1,2} collisions : {1, 0}
   number of rebuilds : 0
16- inserting 3 elements
   the table is O(0.2 n)-Space
   level {1,2} collisions : {0, 0}
   number of rebuilds : 0
17- inserting 3 elements
   the table is O(0.2 n)-Space
   level {1,2} collisions : {0, 0}
   number of rebuilds : 0
18- inserting 13 elements
   the table is O(1.8666666666666667 n)-Space
   level {1,2} collisions : {3, 1}
   number of rebuilds : 0
19- inserting 11 elements
   the table is O(1.1333333333333333 n)-Space
   level {1,2} collisions : {3, 1}
   number of rebuilds : 0
20- inserting 9 elements
   the table is O(0.8666666666666667 n)-Space
   level {1,2} collisions : {2, 0}
   number of rebuilds : 0
```

Test 2 – O(N)-Space

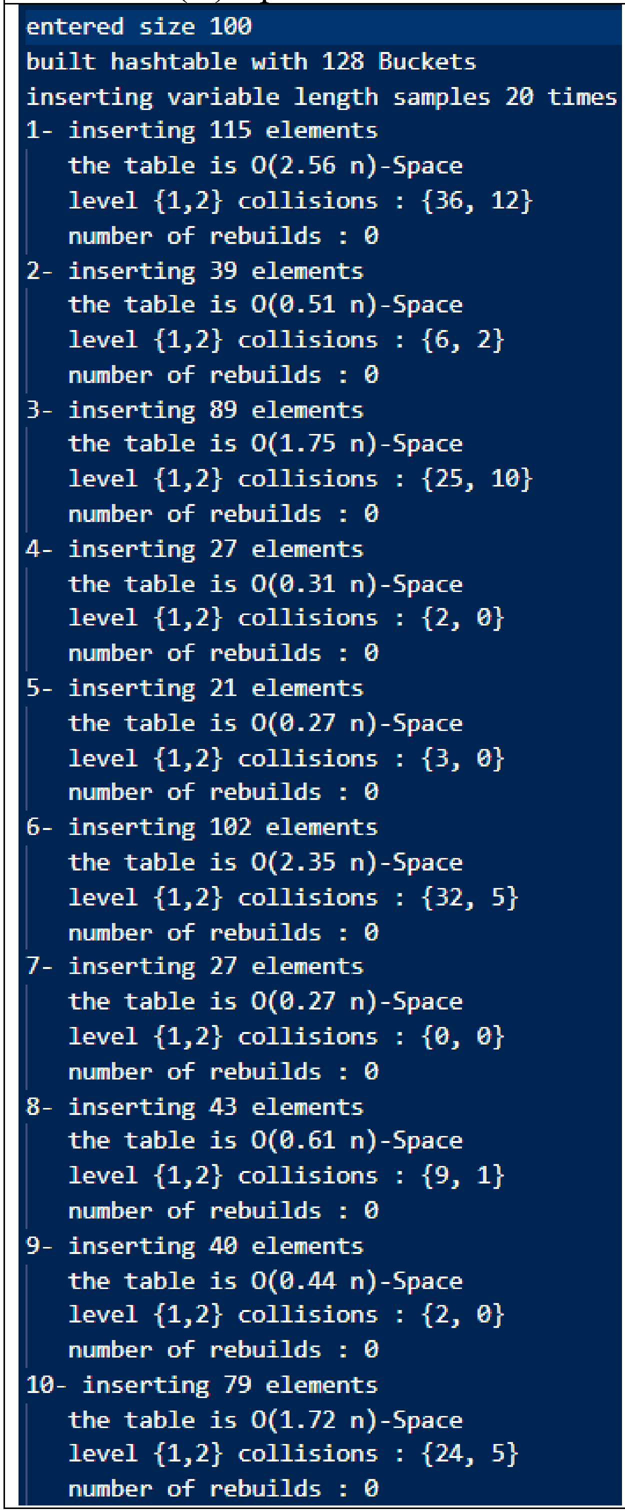


```
entered size 50
built hashtable with 64 Buckets
inserting variable length samples 20 times
1- inserting 37 elements
   the table is O(1.16 n)-Space
   level {1,2} collisions : {6, 4}
   number of rebuilds : 0
2- inserting 36 elements
   the table is O(0.96 n)-Space
   level {1,2} collisions : {6, 1}
   number of rebuilds : 0
3- inserting 15 elements
   the table is O(0.58 n)-Space
   level {1,2} collisions : {4, 0}
   number of rebuilds : 0
4- inserting 34 elements
   the table is O(1.3 n)-Space
   level {1,2} collisions : {11, 3}
   number of rebuilds : 0
5- inserting 18 elements
   the table is O(0.48 n)-Space
   level {1,2} collisions : {3, 0}
   number of rebuilds : 0
6- inserting 15 elements
   the table is O(0.34 n)-Space
   level {1,2} collisions : {1, 0}
   number of rebuilds : 0
7- inserting 53 elements
   the table is O(2.14 n)-Space
   level {1,2} collisions : {15, 1}
   number of rebuilds : 0
8- inserting 16 elements
   the table is O(0.36 n)-Space
   level {1,2} collisions : {1, 0}
   number of rebuilds : 0
9- inserting 25 elements
   the table is O(0.88 n)-Space
   level {1,2} collisions : {5, 3}
   number of rebuilds : 0
10- inserting 49 elements
   the table is O(2.38 n)-Space
   level {1,2} collisions : {17, 5}
   number of rebuilds : 0
```

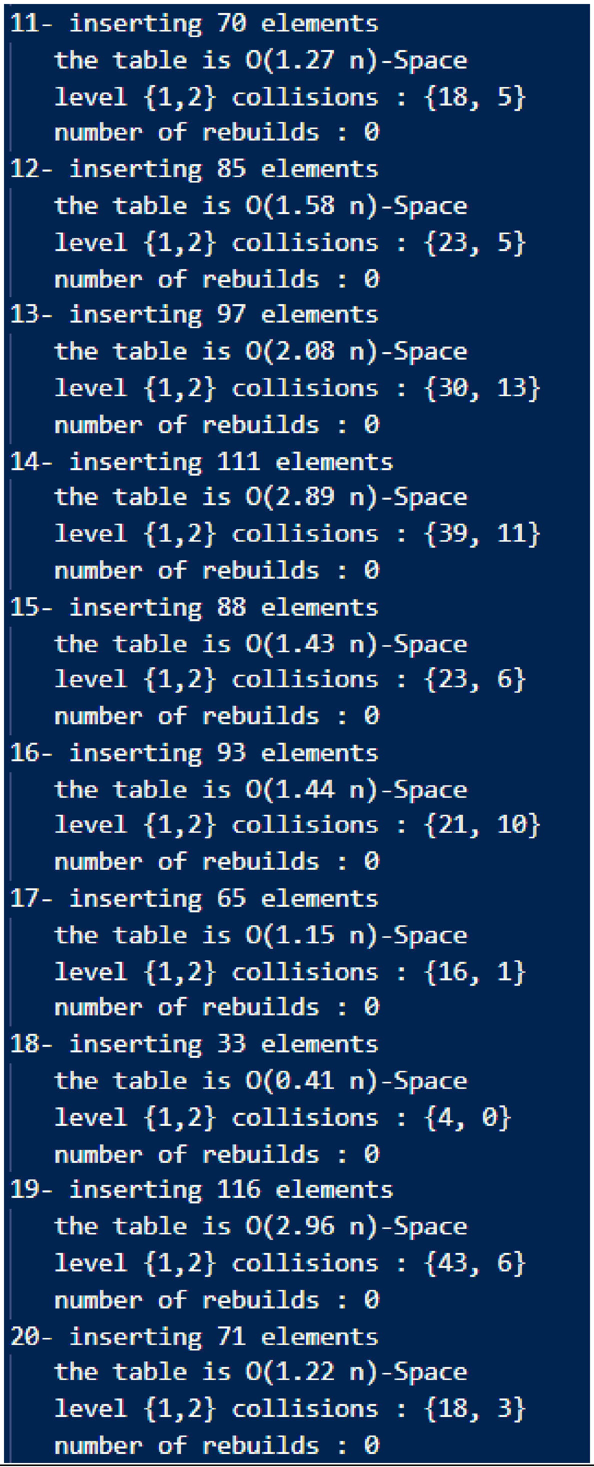


```
11- inserting 15 elements
   the table is O(0.42 n)-Space
   level {1,2} collisions : {3, 0}
   number of rebuilds : 0
12- inserting 32 elements
   the table is O(0.96 n)-Space
   level {1,2} collisions : {5, 1}
   number of rebuilds : 0
13- inserting 32 elements
   the table is O(0.8 n)-Space
   level {1,2} collisions : {4, 3}
   number of rebuilds : 0
14- inserting 3 elements
   the table is O(0.06 n)-Space
   level {1,2} collisions : {0, 0}
   number of rebuilds : 0
15- inserting 0 elements
   the table is O(0.0 n)-Space
   level {1,2} collisions : {0, 0}
   number of rebuilds : 0
16- inserting 50 elements
   the table is O(2.26 n)-Space
   level {1,2} collisions : {15, 4}
   number of rebuilds : 0
17- inserting 34 elements
   the table is O(1.26 n)-Space
   level {1,2} collisions : {7, 1}
   number of rebuilds : 0
18- inserting 44 elements
   the table is O(1.44 n)-Space
   level {1,2} collisions : {14, 5}
   number of rebuilds : 0
19- inserting 21 elements
   the table is O(0.5 n)-Space
   level {1,2} collisions : {2, 1}
   number of rebuilds : 0
20- inserting 34 elements
   the table is O(1.36 n)-Space
   level {1,2} collisions : {8, 1}
   number of rebuilds : 0
```

Test 3 – O(N)-Space

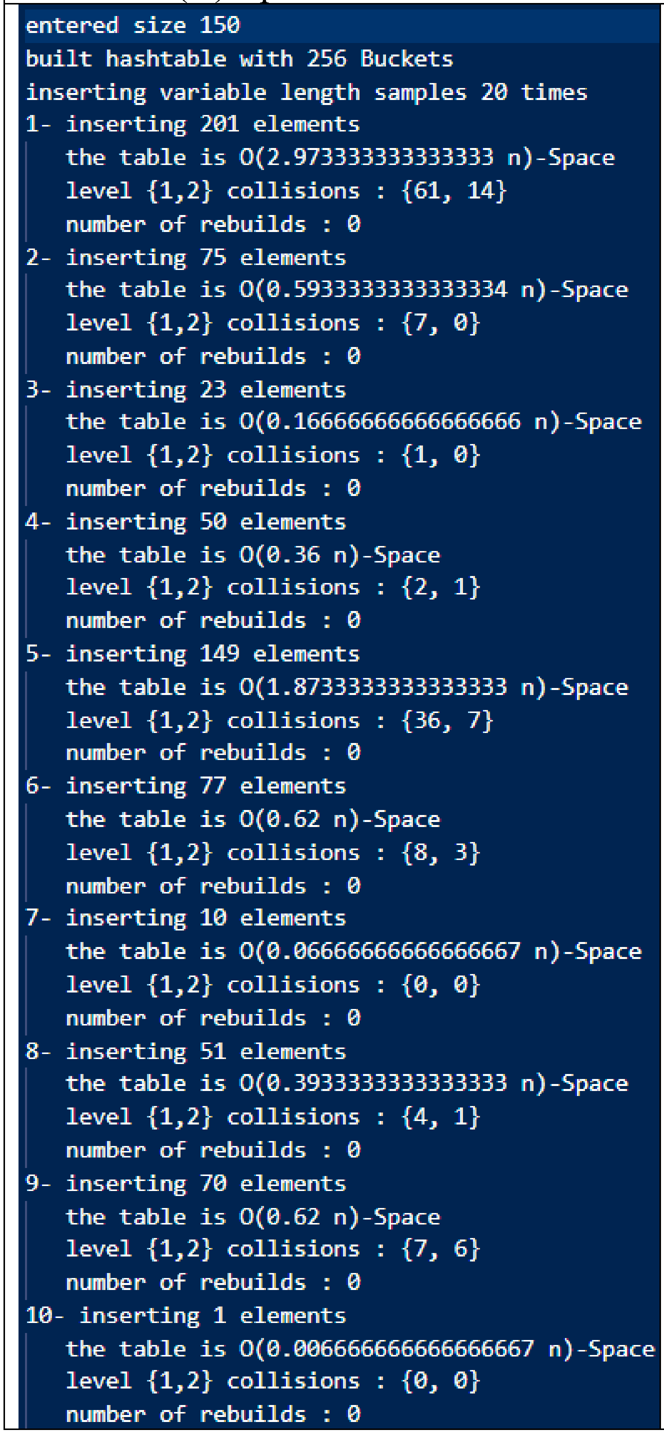


```
entered size 100
built hashtable with 128 Buckets
inserting variable length samples 20 times
1- inserting 115 elements
   the table is O(2.56 n)-Space
   level {1,2} collisions : {36, 12}
   number of rebuilds : 0
2- inserting 39 elements
   the table is O(0.51 n)-Space
   level {1,2} collisions : {6, 2}
   number of rebuilds : 0
3- inserting 89 elements
   the table is O(1.75 n)-Space
   level {1,2} collisions : {25, 10}
   number of rebuilds : 0
4- inserting 27 elements
   the table is O(0.31 n)-Space
   level {1,2} collisions : {2, 0}
   number of rebuilds : 0
5- inserting 21 elements
   the table is O(0.27 n)-Space
   level {1,2} collisions : {3, 0}
   number of rebuilds : 0
6- inserting 102 elements
   the table is O(2.35 n)-Space
   level {1,2} collisions : {32, 5}
   number of rebuilds : 0
7- inserting 27 elements
   the table is O(0.27 n)-Space
   level {1,2} collisions : {0, 0}
   number of rebuilds : 0
8- inserting 43 elements
   the table is O(0.61 n)-Space
   level {1,2} collisions : {9, 1}
   number of rebuilds : 0
9- inserting 40 elements
   the table is O(0.44 n)-Space
   level {1,2} collisions : {2, 0}
   number of rebuilds : 0
10- inserting 79 elements
   the table is O(1.72 n)-Space
   level {1,2} collisions : {24, 5}
   number of rebuilds : 0
```

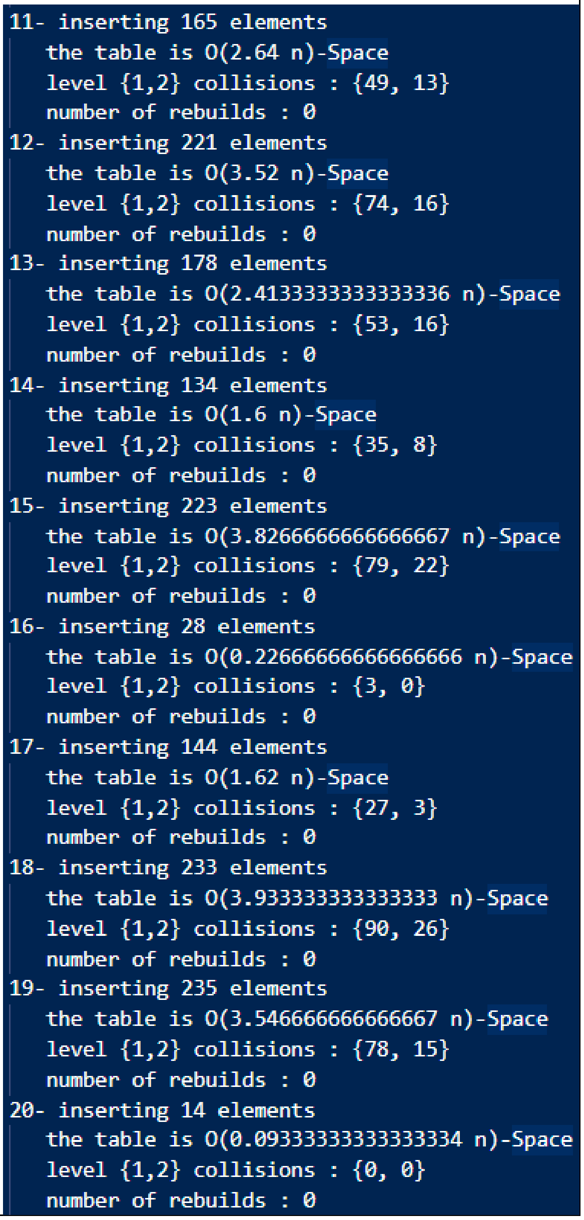


```
11- inserting 70 elements
   the table is O(1.27 n)-Space
   level {1,2} collisions : {18, 5}
   number of rebuilds : 0
12- inserting 85 elements
   the table is O(1.58 n)-Space
   level {1,2} collisions : {23, 5}
   number of rebuilds : 0
13- inserting 97 elements
   the table is O(2.08 n)-Space
   level {1,2} collisions : {30, 13}
   number of rebuilds : 0
14- inserting 111 elements
   the table is O(2.89 n)-Space
   level {1,2} collisions : {39, 11}
   number of rebuilds : 0
15- inserting 88 elements
   the table is O(1.43 n)-Space
   level {1,2} collisions : {23, 6}
   number of rebuilds : 0
16- inserting 93 elements
   the table is O(1.44 n)-Space
   level {1,2} collisions : {21, 10}
   number of rebuilds : 0
17- inserting 65 elements
   the table is O(1.15 n)-Space
   level {1,2} collisions : {16, 1}
   number of rebuilds : 0
18- inserting 33 elements
   the table is O(0.41 n)-Space
   level {1,2} collisions : {4, 0}
   number of rebuilds : 0
19- inserting 116 elements
   the table is O(2.96 n)-Space
   level {1,2} collisions : {43, 6}
   number of rebuilds : 0
20- inserting 71 elements
   the table is O(1.22 n)-Space
   level {1,2} collisions : {18, 3}
   number of rebuilds : 0
```

Test 4 – O(N)-Space

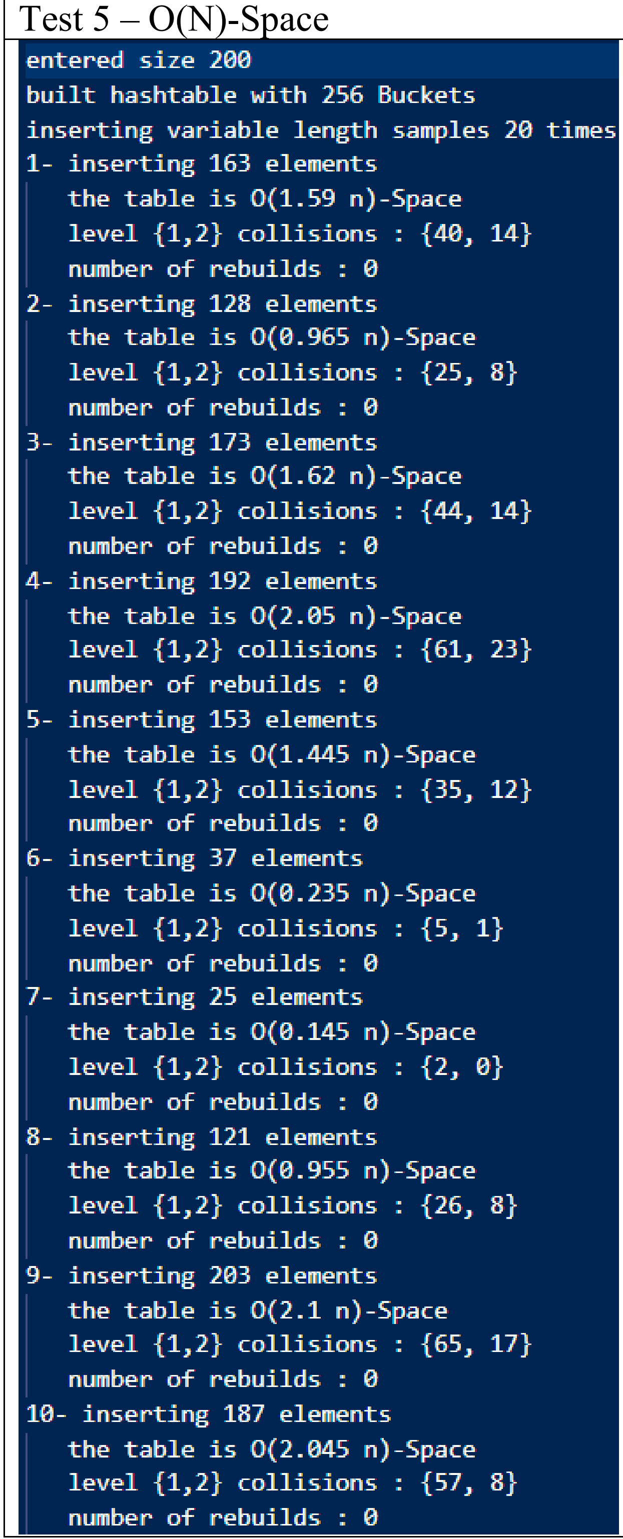


```
entered size 150
built hashtable with 256 Buckets
inserting variable length samples 20 times
1- inserting 201 elements
   the table is O(2.973333333333333 n)-Space
   level {1,2} collisions : {61, 14}
   number of rebuilds : 0
2- inserting 75 elements
   the table is O(0.5933333333333334 n)-Space
   level {1,2} collisions : {7, 0}
   number of rebuilds : 0
3- inserting 23 elements
   the table is O(0.16666666666666666 n)-Space
   level {1,2} collisions : {1, 0}
   number of rebuilds : 0
4- inserting 50 elements
   the table is O(0.36 n)-Space
   level {1,2} collisions : {2, 1}
   number of rebuilds : 0
5- inserting 149 elements
   the table is O(1.8733333333333333 n)-Space
   level {1,2} collisions : {36, 7}
   number of rebuilds : 0
6- inserting 77 elements
   the table is O(0.62 n)-Space
   level {1,2} collisions : {8, 3}
   number of rebuilds : 0
7- inserting 10 elements
   the table is O(0.06666666666666667 n)-Space
   level {1,2} collisions : {0, 0}
   number of rebuilds : 0
8- inserting 51 elements
   the table is O(0.3933333333333333 n)-Space
   level {1,2} collisions : {4, 1}
   number of rebuilds : 0
9- inserting 70 elements
   the table is O(0.62 n)-Space
   level {1,2} collisions : {7, 6}
   number of rebuilds : 0
10- inserting 1 elements
   the table is O(0.006666666666666667 n)-Space
   level {1,2} collisions : {0, 0}
   number of rebuilds : 0
```

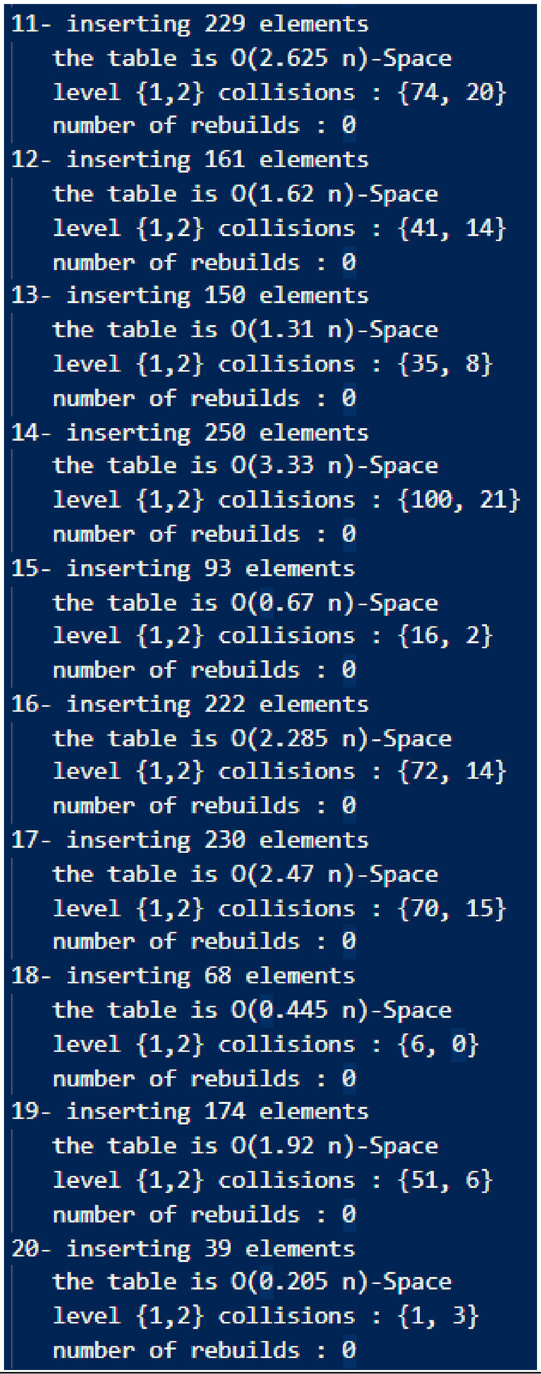


```
11- inserting 165 elements
   the table is O(2.64 n)-Space
   level {1,2} collisions : {49, 13}
   number of rebuilds : 0
12- inserting 221 elements
   the table is O(3.52 n)-Space
   level {1,2} collisions : {74, 16}
   number of rebuilds : 0
13- inserting 178 elements
   the table is O(2.4133333333333336 n)-Space
   level {1,2} collisions : {53, 16}
   number of rebuilds : 0
14- inserting 134 elements
   the table is O(1.6 n)-Space
   level {1,2} collisions : {35, 8}
   number of rebuilds : 0
15- inserting 223 elements
   the table is O(3.8266666666666667 n)-Space
   level {1,2} collisions : {79, 22}
   number of rebuilds : 0
16- inserting 28 elements
   the table is O(0.22666666666666666 n)-Space
   level {1,2} collisions : {3, 0}
   number of rebuilds : 0
17- inserting 144 elements
   the table is O(1.62 n)-Space
   level {1,2} collisions : {27, 3}
   number of rebuilds : 0
18- inserting 233 elements
   the table is O(3.933333333333333 n)-Space
   level {1,2} collisions : {90, 26}
   number of rebuilds : 0
19- inserting 235 elements
   the table is O(3.546666666666667 n)-Space
   level {1,2} collisions : {78, 15}
   number of rebuilds : 0
20- inserting 14 elements
   the table is O(0.09333333333333334 n)-Space
   level {1,2} collisions : {0, 0}
   number of rebuilds : 0
```

Test 5 – O(N)-Space

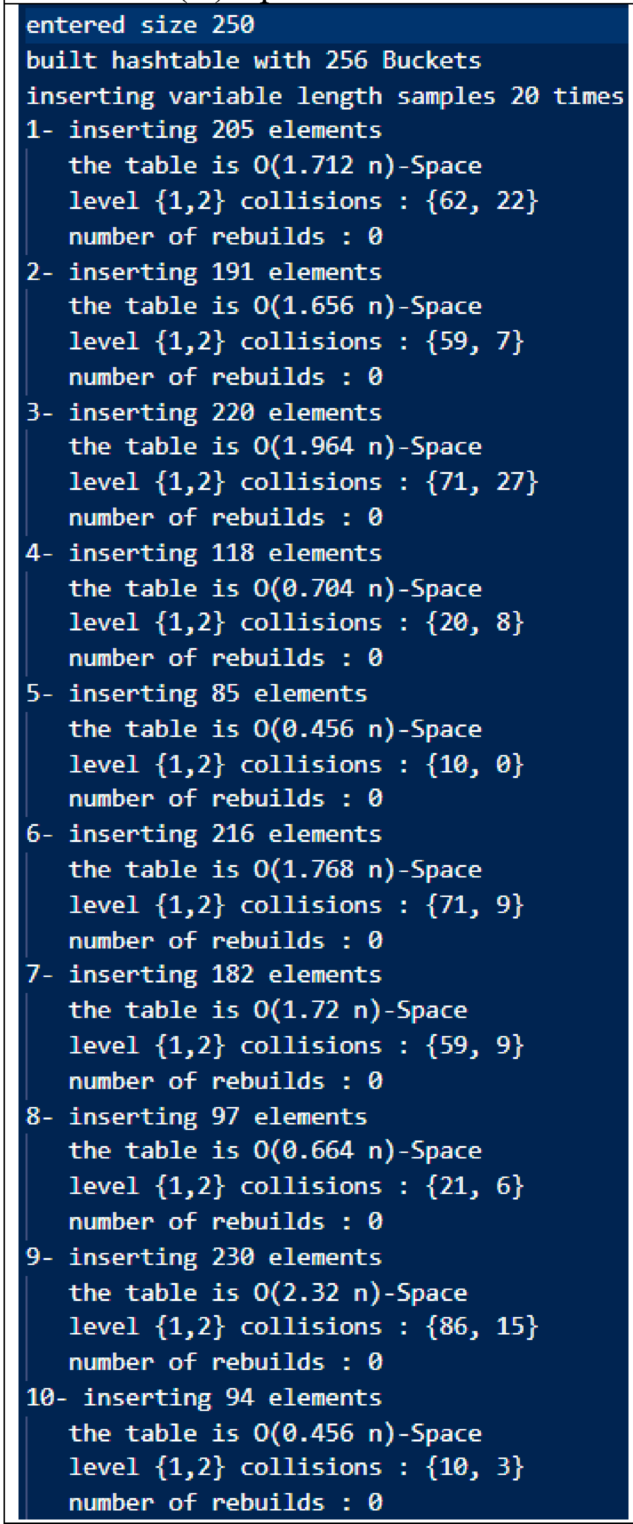


```
entered size 200
built hashtable with 256 Buckets
inserting variable length samples 20 times
1- inserting 163 elements
   the table is O(1.59 n)-Space
   level {1,2} collisions : {40, 14}
   number of rebuilds : 0
2- inserting 128 elements
   the table is O(0.965 n)-Space
   level {1,2} collisions : {25, 8}
   number of rebuilds : 0
3- inserting 173 elements
   the table is O(1.62 n)-Space
   level {1,2} collisions : {44, 14}
   number of rebuilds : 0
4- inserting 192 elements
   the table is O(2.05 n)-Space
   level {1,2} collisions : {61, 23}
   number of rebuilds : 0
5- inserting 153 elements
   the table is O(1.445 n)-Space
   level {1,2} collisions : {35, 12}
   number of rebuilds : 0
6- inserting 37 elements
   the table is O(0.235 n)-Space
   level {1,2} collisions : {5, 1}
   number of rebuilds : 0
7- inserting 25 elements
   the table is O(0.145 n)-Space
   level {1,2} collisions : {2, 0}
   number of rebuilds : 0
8- inserting 121 elements
   the table is O(0.955 n)-Space
   level {1,2} collisions : {26, 8}
   number of rebuilds : 0
9- inserting 203 elements
   the table is O(2.1 n)-Space
   level {1,2} collisions : {65, 17}
   number of rebuilds : 0
10- inserting 187 elements
   the table is O(2.045 n)-Space
   level {1,2} collisions : {57, 8}
   number of rebuilds : 0
```

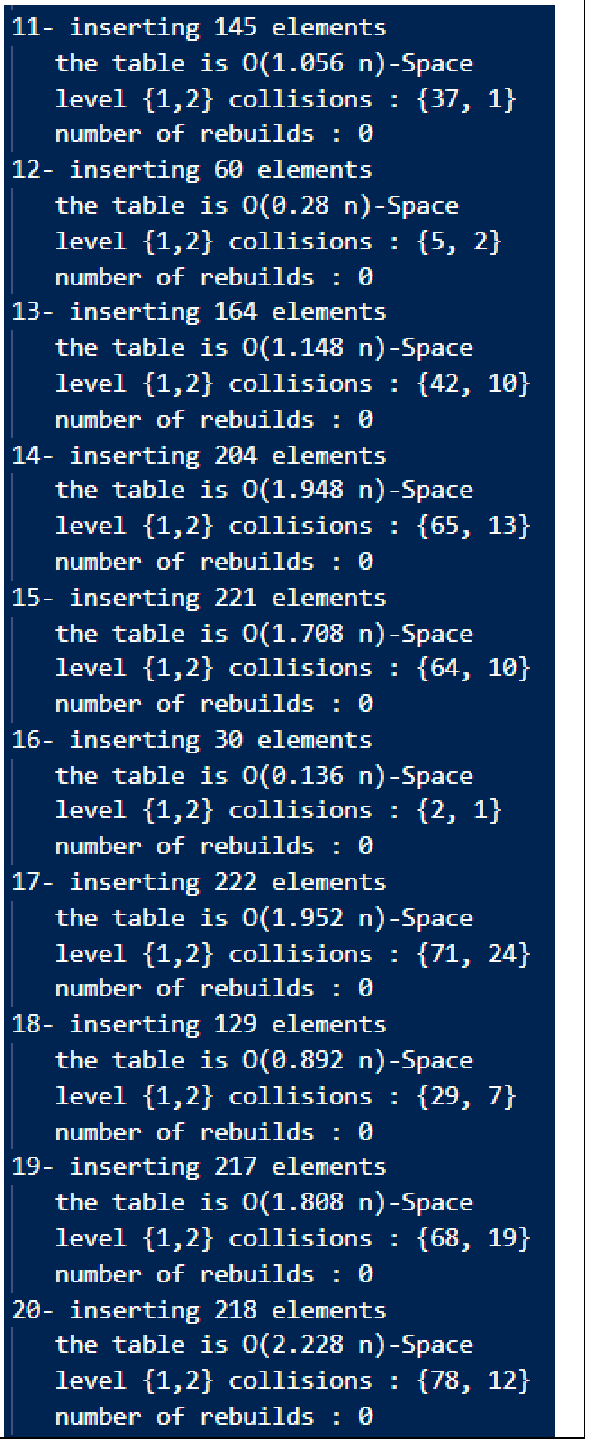


```
11- inserting 229 elements
   the table is O(2.625 n)-Space
   level {1,2} collisions : {74, 20}
   number of rebuilds : 0
12- inserting 161 elements
   the table is O(1.62 n)-Space
   level {1,2} collisions : {41, 14}
   number of rebuilds : 0
13- inserting 150 elements
   the table is O(1.31 n)-Space
   level {1,2} collisions : {35, 8}
   number of rebuilds : 0
14- inserting 250 elements
   the table is O(3.33 n)-Space
   level {1,2} collisions : {100, 21}
   number of rebuilds : 0
15- inserting 93 elements
   the table is O(0.67 n)-Space
   level {1,2} collisions : {16, 2}
   number of rebuilds : 0
16- inserting 222 elements
   the table is O(2.285 n)-Space
   level {1,2} collisions : {72, 14}
   number of rebuilds : 0
17- inserting 230 elements
   the table is O(2.47 n)-Space
   level {1,2} collisions : {70, 15}
   number of rebuilds : 0
18- inserting 68 elements
   the table is O(0.445 n)-Space
   level {1,2} collisions : {6, 0}
   number of rebuilds : 0
19- inserting 174 elements
   the table is O(1.92 n)-Space
   level {1,2} collisions : {51, 6}
   number of rebuilds : 0
20- inserting 39 elements
   the table is O(0.205 n)-Space
   level {1,2} collisions : {1, 3}
   number of rebuilds : 0
```

Test 6 – O(N)-Space

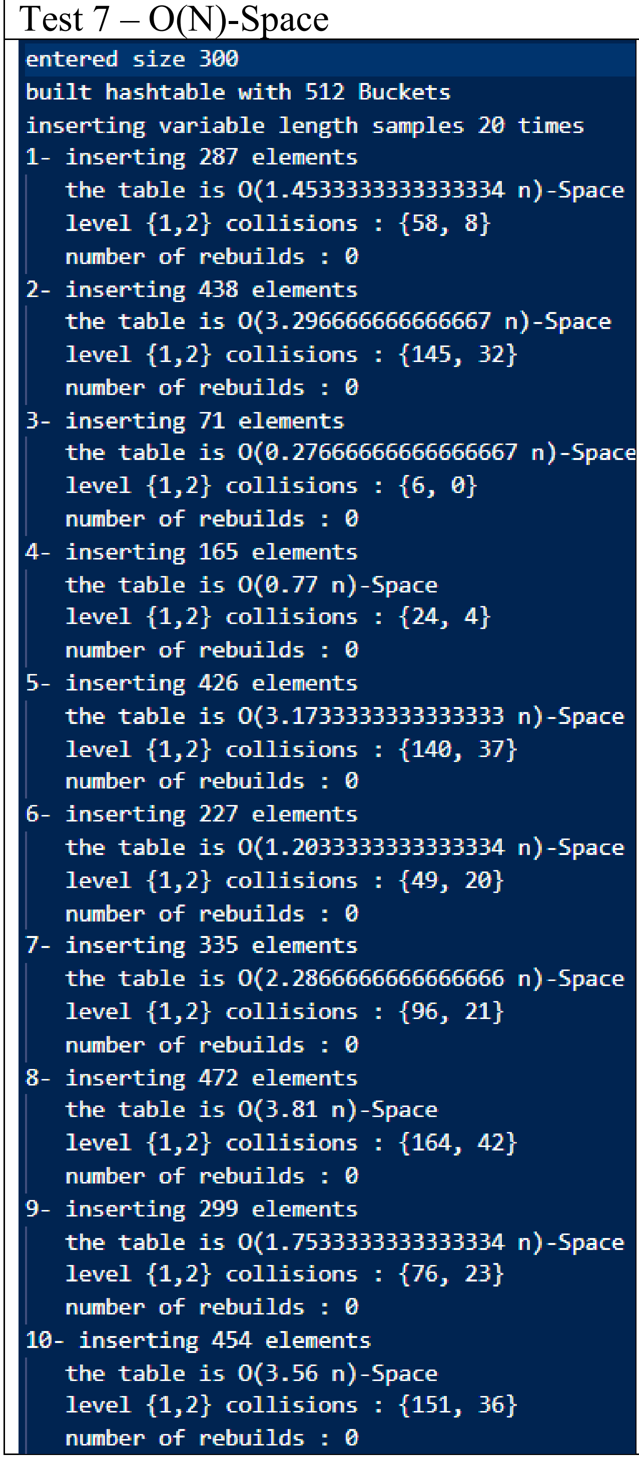


```
entered size 250
built hashtable with 256 Buckets
inserting variable length samples 20 times
1- inserting 205 elements
   the table is O(1.712 n)-Space
   level {1,2} collisions : {62, 22}
   number of rebuilds : 0
2- inserting 191 elements
   the table is O(1.656 n)-Space
   level {1,2} collisions : {59, 7}
   number of rebuilds : 0
3- inserting 220 elements
   the table is O(1.964 n)-Space
   level {1,2} collisions : {71, 27}
   number of rebuilds : 0
4- inserting 118 elements
   the table is O(0.704 n)-Space
   level {1,2} collisions : {20, 8}
   number of rebuilds : 0
5- inserting 85 elements
   the table is O(0.456 n)-Space
   level {1,2} collisions : {10, 0}
   number of rebuilds : 0
6- inserting 216 elements
   the table is O(1.768 n)-Space
   level {1,2} collisions : {71, 9}
   number of rebuilds : 0
7- inserting 182 elements
   the table is O(1.72 n)-Space
   level {1,2} collisions : {59, 9}
   number of rebuilds : 0
8- inserting 97 elements
   the table is O(0.664 n)-Space
   level {1,2} collisions : {21, 6}
   number of rebuilds : 0
9- inserting 230 elements
   the table is O(2.32 n)-Space
   level {1,2} collisions : {86, 15}
   number of rebuilds : 0
10- inserting 94 elements
   the table is O(0.456 n)-Space
   level {1,2} collisions : {10, 3}
   number of rebuilds : 0
```

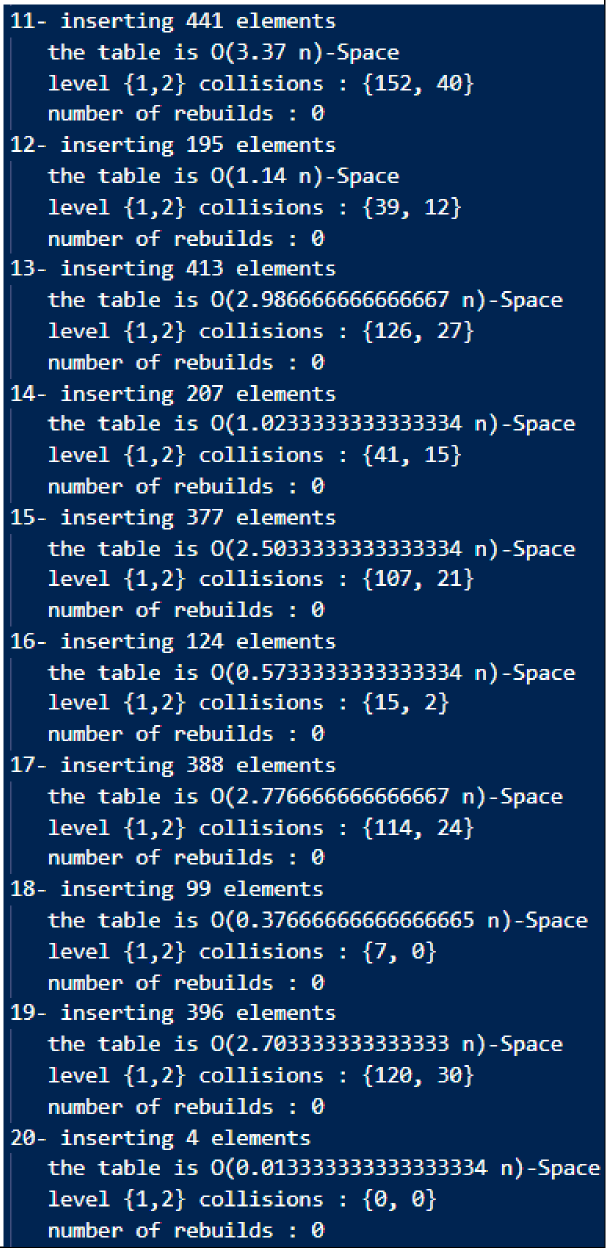


```
11- inserting 145 elements
   the table is O(1.056 n)-Space
   level {1,2} collisions : {37, 1}
   number of rebuilds : 0
12- inserting 60 elements
   the table is O(0.28 n)-Space
   level {1,2} collisions : {5, 2}
   number of rebuilds : 0
13- inserting 164 elements
   the table is O(1.148 n)-Space
   level {1,2} collisions : {42, 10}
   number of rebuilds : 0
14- inserting 204 elements
   the table is O(1.948 n)-Space
   level {1,2} collisions : {65, 13}
   number of rebuilds : 0
15- inserting 221 elements
   the table is O(1.708 n)-Space
   level {1,2} collisions : {64, 10}
   number of rebuilds : 0
16- inserting 30 elements
   the table is O(0.136 n)-Space
   level {1,2} collisions : {2, 1}
   number of rebuilds : 0
17- inserting 222 elements
   the table is O(1.952 n)-Space
   level {1,2} collisions : {71, 24}
   number of rebuilds : 0
18- inserting 129 elements
   the table is O(0.892 n)-Space
   level {1,2} collisions : {29, 7}
   number of rebuilds : 0
19- inserting 217 elements
   the table is O(1.808 n)-Space
   level {1,2} collisions : {68, 19}
   number of rebuilds : 0
20- inserting 218 elements
   the table is O(2.228 n)-Space
   level {1,2} collisions : {78, 12}
   number of rebuilds : 0
```

Test 7 – O(N)-Space

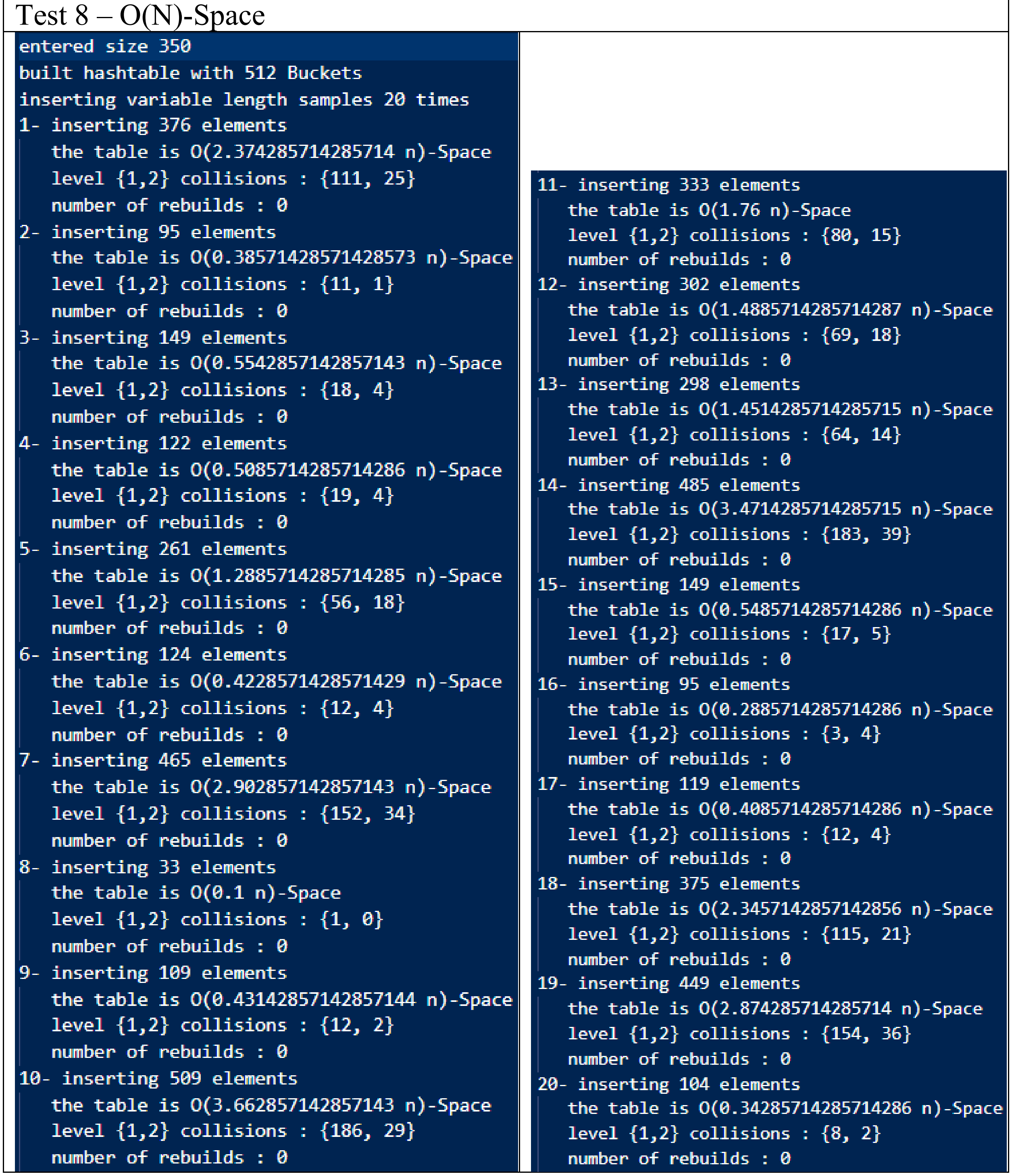


```
entered size 300
built hashtable with 512 Buckets
inserting variable length samples 20 times
1- inserting 287 elements
   the table is O(1.4533333333333334 n)-Space
   level {1,2} collisions : {58, 8}
   number of rebuilds : 0
2- inserting 438 elements
   the table is O(3.296666666666667 n)-Space
   level {1,2} collisions : {145, 32}
   number of rebuilds : 0
3- inserting 71 elements
   the table is O(0.27666666666666667 n)-Space
   level {1,2} collisions : {6, 0}
   number of rebuilds : 0
4- inserting 165 elements
   the table is O(0.77 n)-Space
   level {1,2} collisions : {24, 4}
   number of rebuilds : 0
5- inserting 426 elements
   the table is O(3.1733333333333333 n)-Space
   level {1,2} collisions : {140, 37}
   number of rebuilds : 0
6- inserting 227 elements
   the table is O(1.2033333333333334 n)-Space
   level {1,2} collisions : {49, 20}
   number of rebuilds : 0
7- inserting 335 elements
   the table is O(2.2866666666666666 n)-Space
   level {1,2} collisions : {96, 21}
   number of rebuilds : 0
8- inserting 472 elements
   the table is O(3.81 n)-Space
   level {1,2} collisions : {164, 42}
   number of rebuilds : 0
9- inserting 299 elements
   the table is O(1.7533333333333334 n)-Space
   level {1,2} collisions : {76, 23}
   number of rebuilds : 0
10- inserting 454 elements
   the table is O(3.56 n)-Space
   level {1,2} collisions : {151, 36}
   number of rebuilds : 0
```



```
11- inserting 441 elements
   the table is O(3.37 n)-Space
   level {1,2} collisions : {152, 40}
   number of rebuilds : 0
12- inserting 195 elements
   the table is O(1.14 n)-Space
   level {1,2} collisions : {39, 12}
   number of rebuilds : 0
13- inserting 413 elements
   the table is O(2.986666666666667 n)-Space
   level {1,2} collisions : {126, 27}
   number of rebuilds : 0
14- inserting 207 elements
   the table is O(1.0233333333333334 n)-Space
   level {1,2} collisions : {41, 15}
   number of rebuilds : 0
15- inserting 377 elements
   the table is O(2.5033333333333334 n)-Space
   level {1,2} collisions : {107, 21}
   number of rebuilds : 0
16- inserting 124 elements
   the table is O(0.5733333333333334 n)-Space
   level {1,2} collisions : {15, 2}
   number of rebuilds : 0
17- inserting 388 elements
   the table is O(2.776666666666667 n)-Space
   level {1,2} collisions : {114, 24}
   number of rebuilds : 0
18- inserting 99 elements
   the table is O(0.37666666666666665 n)-Space
   level {1,2} collisions : {7, 0}
   number of rebuilds : 0
19- inserting 396 elements
   the table is O(2.703333333333333 n)-Space
   level {1,2} collisions : {120, 30}
   number of rebuilds : 0
20- inserting 4 elements
   the table is O(0.013333333333333334 n)-Space
   level {1,2} collisions : {0, 0}
   number of rebuilds : 0
```

Test 8 – O(N)-Space



```
entered size 350
built hashtable with 512 Buckets
inserting variable length samples 20 times
1- inserting 376 elements
   the table is O(2.374285714285714 n)-Space
   level {1,2} collisions : {111, 25}
   number of rebuilds : 0
2- inserting 95 elements
   the table is O(0.38571428571428573 n)-Space
   level {1,2} collisions : {11, 1}
   number of rebuilds : 0
3- inserting 149 elements
   the table is O(0.5542857142857143 n)-Space
   level {1,2} collisions : {18, 4}
   number of rebuilds : 0
4- inserting 122 elements
   the table is O(0.5085714285714286 n)-Space
   level {1,2} collisions : {19, 4}
   number of rebuilds : 0
5- inserting 261 elements
   the table is O(1.2885714285714285 n)-Space
   level {1,2} collisions : {56, 18}
   number of rebuilds : 0
6- inserting 124 elements
   the table is O(0.4228571428571429 n)-Space
   level {1,2} collisions : {12, 4}
   number of rebuilds : 0
7- inserting 465 elements
   the table is O(2.902857142857143 n)-Space
   level {1,2} collisions : {152, 34}
   number of rebuilds : 0
8- inserting 33 elements
   the table is O(0.1 n)-Space
   level {1,2} collisions : {1, 0}
   number of rebuilds : 0
9- inserting 109 elements
   the table is O(0.43142857142857144 n)-Space
   level {1,2} collisions : {12, 2}
   number of rebuilds : 0
10- inserting 509 elements
   the table is O(3.662857142857143 n)-Space
   level {1,2} collisions : {186, 29}
   number of rebuilds : 0
```

```
11- inserting 333 elements
   the table is O(1.76 n)-Space
   level {1,2} collisions : {80, 15}
   number of rebuilds : 0
12- inserting 302 elements
   the table is O(1.4885714285714287 n)-Space
   level {1,2} collisions : {69, 18}
   number of rebuilds : 0
13- inserting 298 elements
   the table is O(1.4514285714285715 n)-Space
   level {1,2} collisions : {64, 14}
   number of rebuilds : 0
14- inserting 485 elements
   the table is O(3.4714285714285715 n)-Space
   level {1,2} collisions : {183, 39}
   number of rebuilds : 0
15- inserting 149 elements
   the table is O(0.5485714285714286 n)-Space
   level {1,2} collisions : {17, 5}
   number of rebuilds : 0
16- inserting 95 elements
   the table is O(0.2885714285714286 n)-Space
   level {1,2} collisions : {3, 4}
   number of rebuilds : 0
17- inserting 119 elements
   the table is O(0.4085714285714286 n)-Space
   level {1,2} collisions : {12, 4}
   number of rebuilds : 0
18- inserting 375 elements
   the table is O(2.3457142857142856 n)-Space
   level {1,2} collisions : {115, 21}
   number of rebuilds : 0
19- inserting 449 elements
   the table is O(2.874285714285714 n)-Space
   level {1,2} collisions : {154, 36}
   number of rebuilds : 0
20- inserting 104 elements
   the table is O(0.34285714285714286 n)-Space
   level {1,2} collisions : {8, 2}
   number of rebuilds : 0
```