Q1 d) Implement Bucket Sort Algorithm. Analyze its time complexity.

TIME COMPLEXITY:
1. BEST CASE: O(n) if the input numbers are in uniform distribution
2. WORST CASE: O(n ^ 2)

SPACE COMPLEXITY: O(n)

```
#include <iostream>
#include <vector>
#include <random>


using namespace std;



void bucketSort(vector<double> & A)
{
    int n = int(A.size());

    vector<double> bucket[n];

    for(int i = 0; i < n; i++)
        bucket[int(n * A[i])].push_back(A[i]);

    for(int i = 0; i < n; i++)
         sort(bucket[i].begin(), bucket[i].end());

    int index = 0;

    for(int i = 0; i < n; i++)
        for(int j = 0; j < bucket[i].size(); j++)
            A[index++] = bucket[i][j];
}


int main()
{
    vector<double> A(100);
    for(int i = 0; i < 100; ++i)
        A[i] = ((double) rand() / (RAND_MAX));
    cout << "INITIAL STATE OF ARRAY:\n";
    for(int i = 0; i < 100; ++i)
        cout << A[i] << " ";
    cout << endl << endl;
    bucketSort(A);
    cout << "SORTED ARRAY:\n";
    for(int i = 0; i < 100; ++i)
        cout << A[i] << " ";
    cout << endl;
}
```

OUTPUT:

```
INITIAL STATE OF ARRAY:
7.82637e-06 0.131538 0.755605 0.45865 0.532767 0.218959 0.0470446 0.678865
 0.679296 0.934693 0.383502 0.519416 0.830965 0.0345721 0.0534616 0.5297
 0.671149 0.00769819 0.383416 0.0668422 0.417486 0.686773 0.588977 0.930436
 0.846167 0.526929 0.0919649 0.653919 0.415999 0.701191 0.910321 0.762198
 0.262453 0.0474645 0.736082 0.328234 0.632639 0.75641 0.991037 0.365339
 0.247039 0.98255 0.72266 0.753356 0.651519 0.0726859 0.631635 0.884707
 0.27271 0.436411 0.766495 0.477732 0.237774 0.274907 0.359265 0.166507
 0.486517 0.897656 0.909208 0.0605643 0.904653 0.504523 0.516292 0.319033
 0.986642 0.493977 0.266145 0.0907329 0.947764 0.0737491 0.500707 0.384142
 0.277082 0.913817 0.529747 0.464446 0.94098 0.050084 0.761514 0.770205
 0.827817 0.125365 0.0158677 0.688455 0.868247 0.629543 0.736225 0.725412
 0.999458 0.888572 0.233195 0.306322 0.351015 0.513274 0.591114 0.845982
 0.412081 0.841511 0.269317 0.415395

SORTED ARRAY:
7.82637e-06 0.00769819 0.0158677 0.0345721 0.0470446 0.0474645 0.050084
 0.0534616 0.0605643 0.0668422 0.0726859 0.0737491 0.0907329 0.0919649
 0.125365 0.131538 0.166507 0.218959 0.233195 0.237774 0.247039 0.262453
 0.266145 0.269317 0.27271 0.274907 0.277082 0.306322 0.319033 0.328234
 0.351015 0.359265 0.365339 0.383416 0.383502 0.384142 0.412081 0.415395
 0.415999 0.417486 0.436411 0.45865 0.464446 0.477732 0.486517 0.493977
 0.500707 0.504523 0.513274 0.516292 0.519416 0.526929 0.5297 0.529747
 0.532767 0.588977 0.591114 0.629543 0.631635 0.632639 0.651519 0.653919
 0.671149 0.678865 0.679296 0.686773 0.688455 0.701191 0.72266 0.725412
 0.736082 0.736225 0.753356 0.755605 0.75641 0.761514 0.762198 0.766495
 0.770205 0.827817 0.830965 0.841511 0.845982 0.846167 0.868247 0.884707
 0.888572 0.897656 0.904653 0.909208 0.910321 0.913817 0.930436 0.934693
 0.94098 0.947764 0.98255 0.986642 0.991037 0.999458
Program ended with exit code: 0
```