ComS 311 Programing Assignment 1 Report

Yangxiao Wang, Cheng Song

1. **Algorithm for the method buildDataStructure():**

   In this method, we create a HashTable adding all the float data as Tuple type element into this HashTable. Tuple <floor(float data), float data> has two parameters, floor(float data) is the key and float data is the value. The hash value of Tuple element equals to (key*a + b) % (range of HashTable), a &b are 2 integers. Tuple's location depends on its own hash value.

2. **Algorithm for the method npHashNearestPoints(float p)**

   Since 1.2 and 2.1 are close, floor(1.2) = 1, floor(2.1) = 2. 1.2 and 0.9 are close, floor(1.2) = 1, floor(0.9) = 0. Therefore, we search key$\pm$1 using the search method from HashTable class. In search method, first, calculate the hash value of the searching float number, then get the result based on this hash value. Some specific non-close numbers may share the same hash value, we compare float p with all the elements from ArrayList from search result.

3.
```
buildDataStructure: 24.405
allNearestPointsNaive: 28.827
allNearestPointsHash: 0.062
```
all the units are seconds

```
public static void main(String[] args) throws FileNotFoundException
{
    NearestPoints np = new NearestPoints("points.txt");
    long time1 =  System.currentTimeMillis();
    np.buildDataStructure();
    time1 = System.currentTimeMillis() - time1;
    System.out.println("buildDataStructure: " + (double)time1/1000);

    time1 =  System.currentTimeMillis();
    np.allNearestPointsNaive();
    time1 = System.currentTimeMillis() - time1;
    System.out.println("allNearestPointsNaive: " + (double)time1/1000);

    time1 =  System.currentTimeMillis();
    np.allNearestPointsHash();
    time1 = System.currentTimeMillis() - time1;
    System.out.println("allNearestPointsHash: " + (double)time1/1000);
}
```