Austin Bullard
Erin Elsbernd
23 February 2017
Computer Science 311

buildDataStructures() Algorithm:

Create a new HashTable, finding the next largest prime number after pointList.size(), and using it as the size of the table.
For all the points in pointList, create a new tuple with the key set to floor(point) (casted as an integer), and the value as the floating point number itself.
Then add the new Tuple to the HashTable.

npHashNearestPoints(float p) Algorithm:

Create two empty ArrayLists one of type Float (nearPoints) and the other of type Tuple (tuplesP/tuplesPminus1/tuplesPplus1).

Get the floor of p and search for it in the HashTable using the HashTable search function where the floor of p is passed in as the key and is hashed to find the index in the ArrayList. This will return an ArrayList of points that are at the hashed index if p exists in the Table. Save these points in tuplesP.

Iterate through all tuples in tuplesP and check if the values are close to p using the abs(p-q) <= 1 inequality. Add the values that fulfill the inequality to nearPoints.

Repeat the previous two steps for p-1, and p+1 as the input to HashTable.search. Store their return values in tuplesPminus1 and Pminus2 respectively and add the close points to the nearPoints ArrayList.

Running Times:

Time to run allNearestPointsNaive(): 18797ms
Time to run allNearestPointsHash(): 236ms