Student Record using hash table:

The data structure is implemented using bucket array with separate chaining mechanism to store the elements. The key to the hash table is the student Id which will identify the bucket array index and a linked list in at each index of the bucket array to store the elements falling in that bucket. From the student id, we use the year and department part to calculate the hash code. All the students of the same year and same department will fall in the same bucket. But because of the compression map each bucket can hold students of different year and department. We discard the roll number in the hash function calculation.

Choice of Hash Function:

HashCode function:

The key used in the hash table is the student id which is of YYYYAAADDDD format. We make use of the YYYYAAA part of the student id to convert the student id to integer. We use polynomial hash function to convert AAA to an integer. Polynomial hash function takes into account the position of the character in the string thereby generating unique integer for different combinations of the department code. For polynomial hash function we use the prime number 33 to reduce the collisions. We add this integer with YYYY to get a unique number for year - department combination.

Compression Map:

We use the h(k) mod N formula to convert the hash code h(k) to one of the index in the bucket array of size N. We initialize the hash table with bucket array of size 31. This prime number ensures that the modulo returns unique numbers and avoids collision.

Input Validations:

1. The input file should have student id and the cgpa separated by delimiter “,”
2. Student Id should be of YYYYAAADDDD format where
   1. YYYY – year from 2008 to 2018
   2. AAA – Department code should be one of the CSC / ARC / ECE / MEC
   3. DDDD – 4 digit roll number in integer padded with 0 i.e if the roll number is 1 then this should be 0001.
3. The cgpa should be float number
4. Any record / line not conforming to 1, 2 and 3 above will not be considered / added to the hash table

Functions:

initializeHash:

populateHashTable:

hallOfFame:

newCourseList:

depAvg:

destroyHash: