Especificación

En este problema, se implementará una estructura de datos llamada 'Hash table', la cual es la implementación de una tabla en donde la llave es una posición en la lista calculada por una función de hash, y en esa posición se encuentran los valores relacionados a esa llave.

Entrada

Una localidad en un estadio (llave), un comprador con su nombre, cédula y cantidad de boletas compradas.

Salida

Una hash table con los datos de los compradores por localidad.

Estrategia

La clase Heap se basa en una lista, la cual tiene un orden específico para ser interpretada como un árbol de búsqueda, además de una política en donde se decide si es un Max Heap o un Min Heap, para construir el Heap se intercambian elementos en la lista, verificando que cada uno cumpla las propiedades del Heap con respecto a su padre y a sus hijos.

Componentes

La clase contiene herramientas para: construir una hash table a partir de una lista, agregar elementos a la hash table, eliminar elementos de la hash table, actualizar elementos de la hash table, obtener la hash table.

Pruebas

La función pruebas contiene una prueba, en donde a partir de una lista se genera una hast table, allí se puede verificar que la lista es convertida correctamente en la estructura de datos mencionada anteriormente.

Complejidad temporal

La complejidad temporal de los algoritmos buildHash y search es O(n), son los algoritmos más costosos, por tanto, la complejidad temporal del programa es O(n).

Complejidad espacial

La hash table requiere una lista de tamaño n, en donde n es el tamaño especificado para la tabla, por tanto tenemos una complejidad espacial lineal.