Team 3 Presentation:

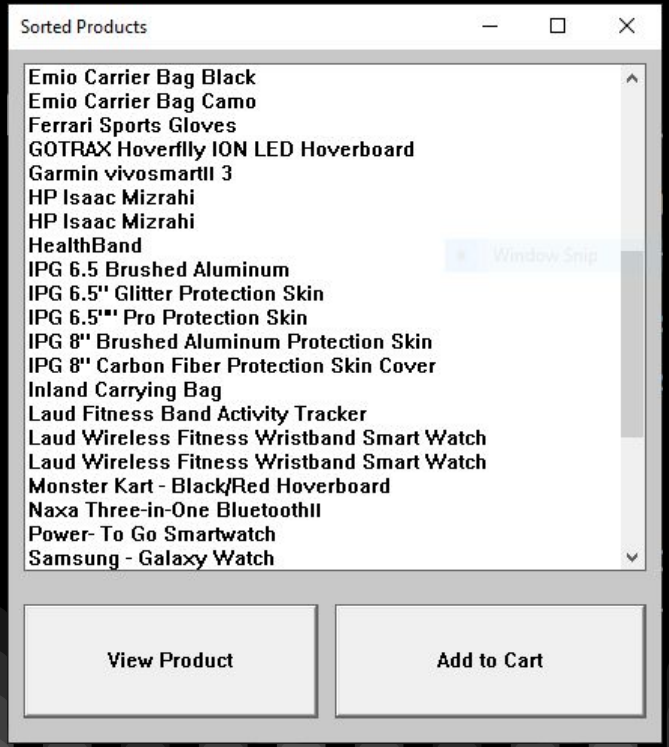
Brandon Nguyen_Connor Pietrasik_Donghun Han_Ryan Hobbs

Project Objective:

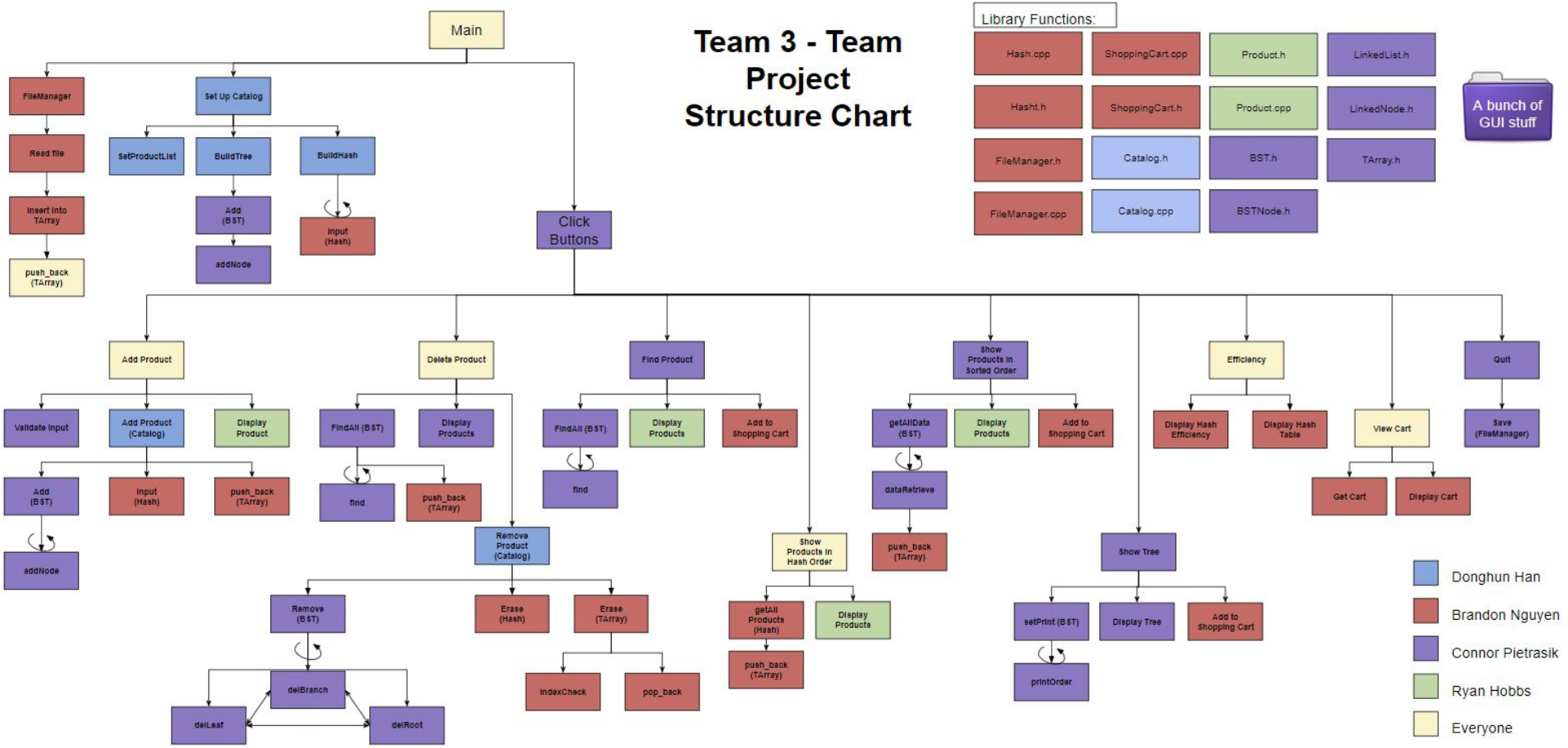
Emulate transactions via inventory manipulation

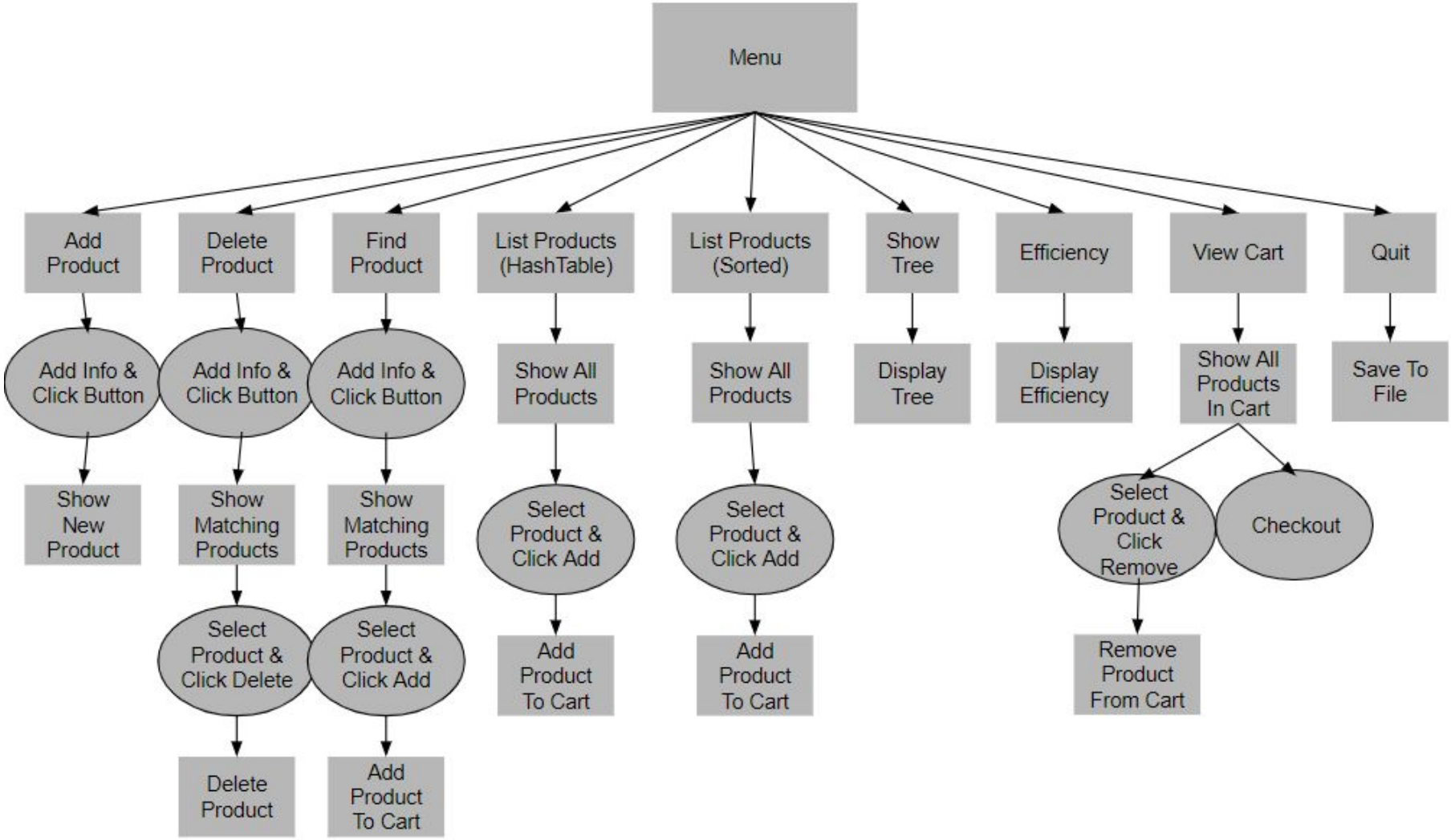
- Cloned sections of Fry's Electronics' sales database

Features:



1. Add New Product
2. Delete Product
3. Find Product
4. List Products(Hash Table)
5. List Products(Sorted)
6. Show Tree
7. Efficiency
8. Add to Shopping Cart
9. Checkout Shopping Cart
10. Quit

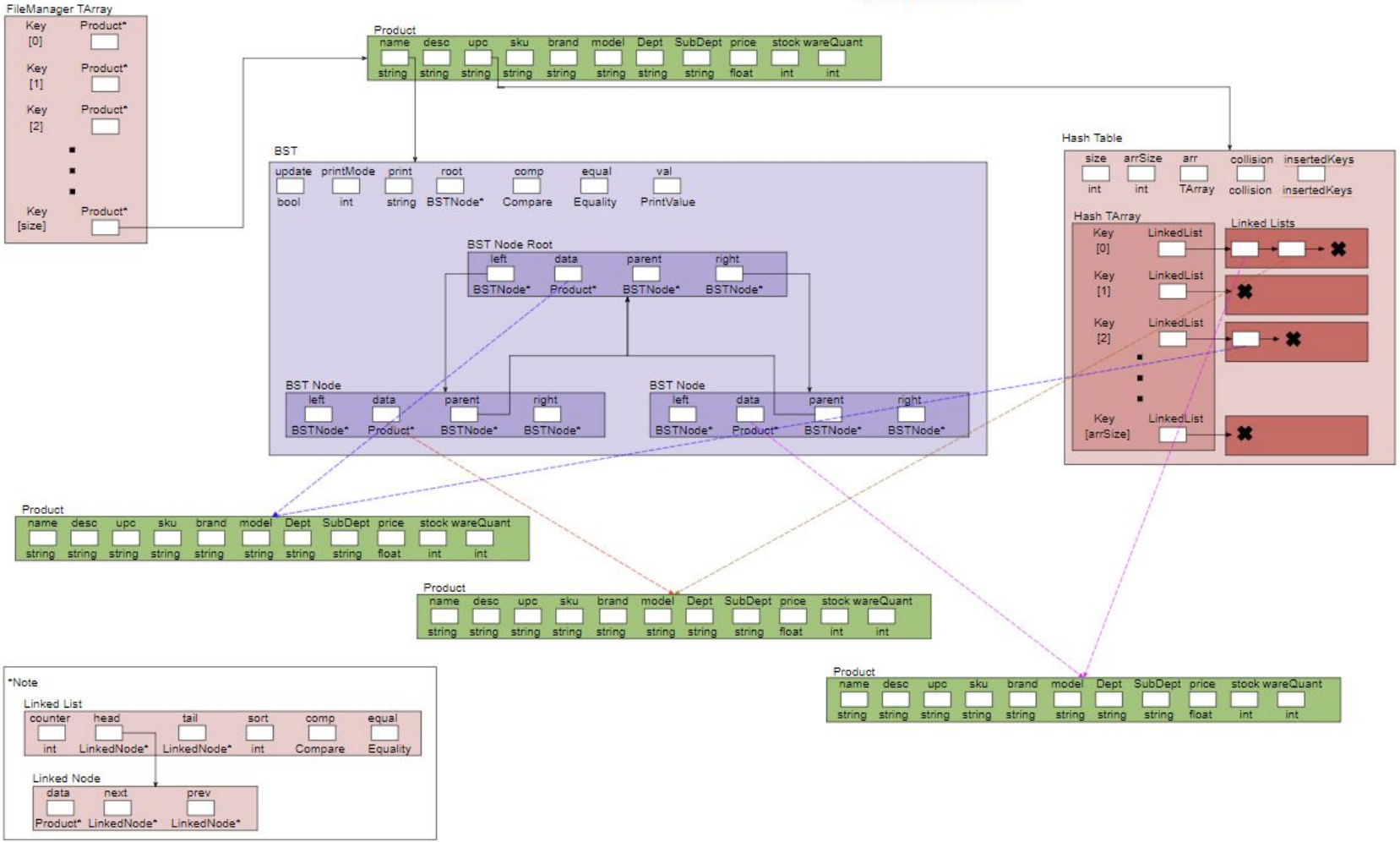

Team 3 - Team Project Structure Chart
Library Functions:
Hash.cpp
ShoppingCart.cpp
Product.h
LinkedList.h
Hasht.h
ShoppingCart.h
Product.cpp
LinkedNode.h
FileManager.h
Catalog.h
BST.h
TArray.h
FileManager.cpp
Catalog.cpp
BSTNode.h
A bunch of GUI stuff
Main
FileManager
Read file
Insert into TArray
push_back (TArray)
Set Up Catalog
setProductList
BuildTree
Add (BST)
addNode
BuildHash
Input (Hash)
Click Buttons
Add Product
Validate Input
Add Product (Catalog)
Add (BST)
addNode
Input (Hash)
Display Product
push_back (TArray)
Delete Product
FindAll (BST)
find
Display Products
push_back (TArray)
Remove Product (Catalog)
Remove (BST)
delBranch
delLeaf
delRoot
Erase (Hash)
Erase (TArray)
IndexCheck
pop_back
Find Product
FindAll (BST)
find
Display Products
Add to Shopping Cart
Show Products in Hash Order
getAll Products (Hash)
push_back (TArray)
Display Products
Show Products in Sorted Order
getAllData (BST)
dataRetrieve
push_back (TArray)
Display Products
Add to Shopping Cart
Show Tree
setPrint (BST)
printOrder
Display Tree
Add to Shopping Cart
Efficiency
Display Hash Efficiency
Display Hash Table
View Cart
Get Cart
Display Cart
Quit
Save (FileManager)
Donghun Han
Brandon Nguyen
Connor Pietrasik
Ryan Hobbs
Everyone


Menu
Add Product
Delete Product
Find Product
List Products (HashTable)
List Products (Sorted)
Show Tree
Efficiency
View Cart
Quit
Add Info & Click Button
Add Info & Click Button
Add Info & Click Button
Show All Products
Show All Products
Display Tree
Display Efficiency
Show All Products In Cart
Save To File
Show New Product
Show Matching Products
Show Matching Products
Select Product & Click Add
Select Product & Click Add
Select Product & Click Remove
Checkout
Select Product & Click Delete
Select Product & Click Add
Add Product To Cart
Add Product To Cart
Remove Product From Cart
Delete Product
Add Product To Cart

CIS 22C

DATA STRUCTURE DIAGRAM

TEAM PROJECT
TEAM NUMBER 3



Hash Table

Hash Function:

Hashing key: string UPC

Add every odd character of a string

The total sum is modulated by the Hash Array Size*

*Hash Array Size is the next prime number of the amount of Products in the TSV.

```
int total = 0;
for (int i = 0; i < str.size(); i++) {
    if (i % 2) total += str[i];
}
```

Hash Table (cont.)

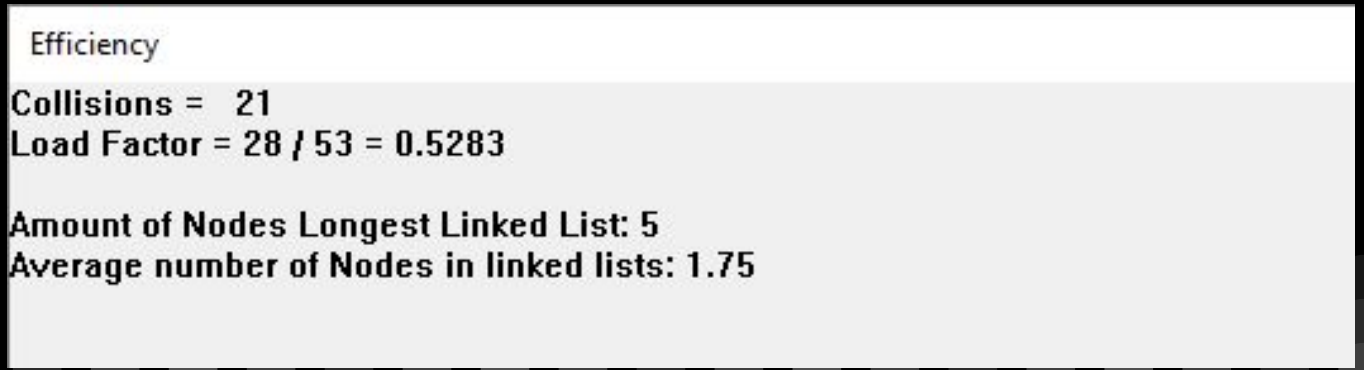
Ex:
Size in tsv: 50
Hash Array Size: 53

UPC: 888255177223
56 + 56 + 53 + 49 + 55 + 50 = 319

Key = 319 % 53 = 1

Hash Efficiency

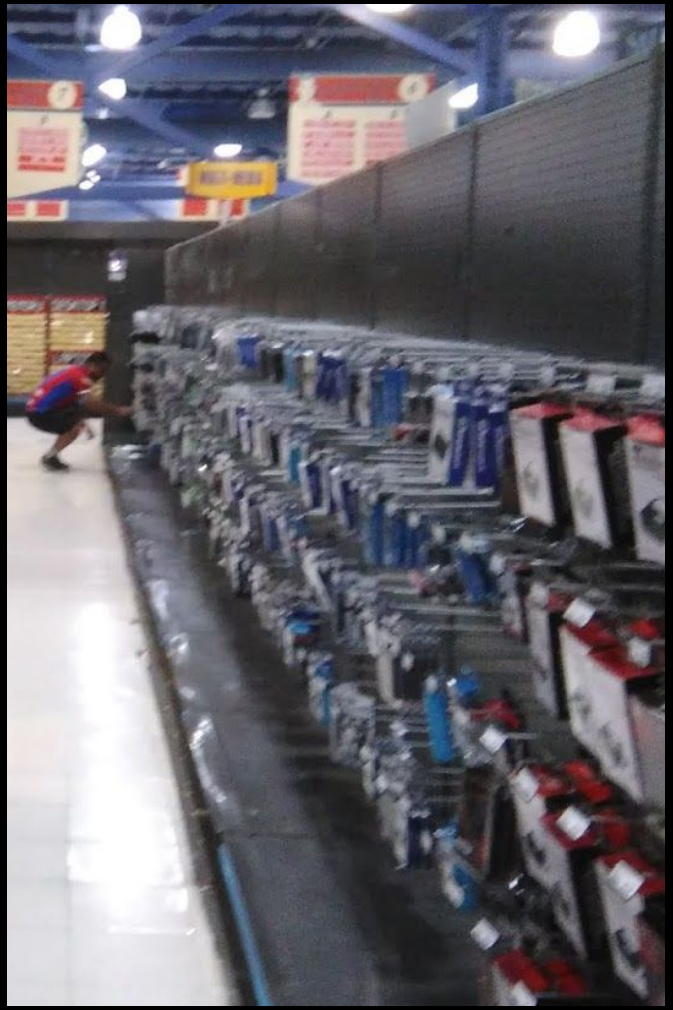
Because collisions are solved through Linked List, there is a high amount of collisions and a small load factor



Product Class

Attributes

- Name
- Description
- UPC Number
- SKU Number
- Brand
- Model
- Department
- Sub Department
- Price
- Stock
- Warehouse Quantity



BST

- Sorts based on name
- Works for whatever
- Can search it
- 9/10 (didn’t add balance)

Catalog

- Catalog class Creates
    - hash table
    - tree database

  from TSV file.

- Main function can fetch data and process it to display customer’s request.