How the program works

The program constitution

Class name	function
SalesInfoMiner.java	Main class
Product.java	Represent the information of product
Receipt.java	Represent the information of receipt
Item.java	Represent the related product of a particular product in several receipt
ProductReader.java	Read product file and validate data
ReceiptReader.java	Read transaction file and validate data correct
Analyst.java	Data mining
AVLTree.java	The main data structure to store data.

The program flow

1. Read the file of product and validate the data then store each Product to AVL Tree pList.
2. Read the file of receipt and validate the data then store each Receipt to AVL Tree rList. When reading the product ID in Receipt, do the first data mining task, search the Product in pList and then add one sale to the attribute: int salesCount.
3. After finished read file, do the second data mining task. Determines the five other products most commonly bought with each product.
4. Resort the Product according to the sales count.
5. Output the sorted Products with each top five products.

The data structure selection

In CSE2ALG labs, we studied how to implement the BS Tree, AVL Tree and Hash Table so far. We also know some other data structures in lectures, however the data structures like RB Tree are too hard to implement for an undergraduate student.

At first, I did not read assignment very carefully. I thought I could use the data structure in Java collection. Because the product should be sorted by key (product ID) and also need be sorted by value(sales count and description), the Map is the best choice. Very quickly, I finished the assignment using Java Map. The time consuming in P6.txt and S6.txt is from 1:35 min (using 5G RAM) to 3:13 min (using 4G RAM), which I did not optimize the sort algorism. The Memory occupancy is unacceptable.

So I try to use the data structure I learnt from lab. Due to the feature of BS Tree, the BS Tree can become a degenerate tree when the data are inserting to the tree in

order. The search performance is become to complexity O (N). The receipts are stored orderly by the key (receipt ID). So that BS Tree is the worst choice.

In order to make a comparison, I tried using BS tree as main data structure. The time consuming in P4.txt and S4.txt is around 44 mins which proved that BS tree is not a good choice. However, BS tree could easily extend to the AVL Tree.

After using AVL Tree, the time consuming in P6.txt and S6.txt is around 2 mins and only used around 1.5G RAM. The time consuming in P5.txt and S5.txt is around 1.5 mins and only used around 1.5G RAM. It indicated that search in AVL Tree have complexity O (log N).

For the reasons of other subjects assignments have nearly some deadline and the performance of AVL Tree is as same as that of the Collection Map, I did not have tried use Hash Table.

Overall, I think I reached the goal of doing this assignment. Done is better than perfect.

Correctness Testing

Test case description	Test data used Expected outcome Actual outcome
For product file	
1.Product Id is invalid – not consisting of 13 digits	5 0001676923e90 6 Honey - Wild Flower 400G ERROR: Invalid Product ID Line 5 in p1.txt Product 0001676923e90 does not consists of 13 digits Check product id ERROR: Invalid Product ID Line 5 in p1.txt Product 0001676923e90 does not consists of 13 digits Check product id
2.Product description is empty/missing	7 0000493930840 8 ERROR: Product description is empty Line 8 in p1.txt Check product description ERROR: Product description is empty Line 8 in p1.txt Check product description

3.Product Id is not unique	5 0001676927890 6 Honey - Wild Flower 400G 7 0001676927890 8 Strawberry Jam - Cottees 500g
	ERROR: Product Id is not Unique Line 7 in p1.txt Product 0001676927890 is not unique Check product id
	ERROR: Product Id is not Unique Line 7 in p1.txt Product 0001676927890 is not unique Check product id
For sales transaction file	
4.Line 1 of a sale transaction does not start with “Receipt number:”	10 Receipt: 3 11 Number of items: 3
	ERROR: Format mistake Line 10 in s1.txt Check file format 'Receipt number:'
	ERROR: Format mistake Line 10 in s1.txt Check file format 'Receipt number:'
5.Receipt number (on line 1) is not a positive integer	15 Receipt number: -4 16 Number of items: 4
	ERROR: Receipt number is not a positive integer Line 15 in s1.txt Check receipt number
	ERROR: Receipt number is not a positive integer Line 15 in s1.txt Check receipt number
6.Line 2 of a sale transaction does not start with “Number of items:”	34 Receipt number: 8 35 Number : 2
	ERROR: Format mistake Line 35 in s1.txt Check file format 'Number of items:'
	ERROR: Format mistake Line 35 in s1.txt Check file format 'Number of items:'
7.Number of items (on line 2) is not a positive integer	34 Receipt number: 8 35 Number of items:
	ERROR: The number of items is not a positive integer Line 35 in s1.txt Check the number of items
	ERROR: The number of items is not a positive integer Line 35 in s1.txt Check the number of items
8.Product ID does not consists of 13 digits	32 Number of items: 1 33 0001577141s64
	ERROR: Invalid Product ID Line 33 in s1.txt Product0001577141s64 does not consists of 13 digits Check product id

	ERROR: Invalid Product ID Line 33 in s1.txt Product0001577141s64 does not consists of 13 digits Check product id
9.Product ID in a sale transaction has duplicates	29 0002010580409 30 0002010580409 31 Receipt number: 7
	ERROR: Product Id has duplicates in a sale transaction Line 30 in s1.txt Product0002010580409 has duplicates Check product id
	ERROR: Product Id has duplicates in a sale transaction Line 30 in s1.txt Product0002010580409 has duplicates Check product id
10.Product ID does not exist in the product file	11 Number of items: 3 12 0002010580409 13 9999999999999 14 0000609616686
	ERROR: The Product ID does not exist in the product file Line 13 in s1.txt Product id:9999999999999 does not exist in the product file Check the Product ID
	ERROR: The Product ID does not exist in the product file Line 13 in s1.txt Product id:9999999999999 does not exist in the product file Check the Product ID
11.There are not enough product ID's listed for a particular transaction (i.e. less than the number listed on line 2)	27 Number of items: 3 28 0000493930840 29 0000609616686 30 Receipt number: 7
	ERROR: Nor enough Product ID's listed for a transaction Line 30 in s1.txt Change the number of items
	ERROR: Nor enough Product ID's listed for a transaction Line 30 in s1.txt Change the number of items

Efficiency Testing

Testing platform: Intel i7-2630QM 2.00GHz, RAM 8GB

Data Structure	P1.txt/ S1.txt	P2.txt/ S2.txt	P3.txt/ S3.txt	P4.txt / S4.txt	P5.txt / S5.txt	P6.txt / S6.txt
Hash Map	127ms	78ms	951ms	37,721ms	67,943ms	138,938ms
BS Tree	62ms	63ms	5,554ms	>10mins	>15mins	
AVL Tree	50ms	55ms	1,112ms	41,769ms	82,258ms	120,900ms

From the table, we can see BS Tree is the worst data structure. After we seen the data set, we found that the products in product file are not sorted by product id or product description and receipts in transcription file are sorted by receipts id.

> **In a *Binary Search Tree*, all the nodes on the left of a given node have key values less than (or equal to) the key of that node (from ALG lecture 09)**

When I use BS Tree to store receipts compared by receipts id (int), the tree will become to Degenerate tree which can be treat as list, for the reason that BS Tree become badly unbalanced.

The binary search in this tree is become to linear search which is much slower than binary search. It is the main reason for BS Tree has worst performance.

AVL Tree is much better than BS Tree. Because AVL Tree automatically rebalance itself if one branch becomes deeper than a sibling. And the insert, search and delete are guaranteed to have complexity O (log N).

For the Hash Map, it is built on hash table. The expected-time O (1) complexity of hash table is better than AVL Tree, but in practice factors these two data structures generally competitive. The performance of Hash Table is depend on having a good hashCode() method and is statistical. One of the shortages of Hash Table is that it does not keep data in order. Another one is that Hash Table takes much more memory than AVL Tree.

In conclusion, AVL Tree is a good and efficiency choice for a programmer who is not good at write hash code.