HABIB UNIVERSITY

2022 May 18

Fenwick Trees

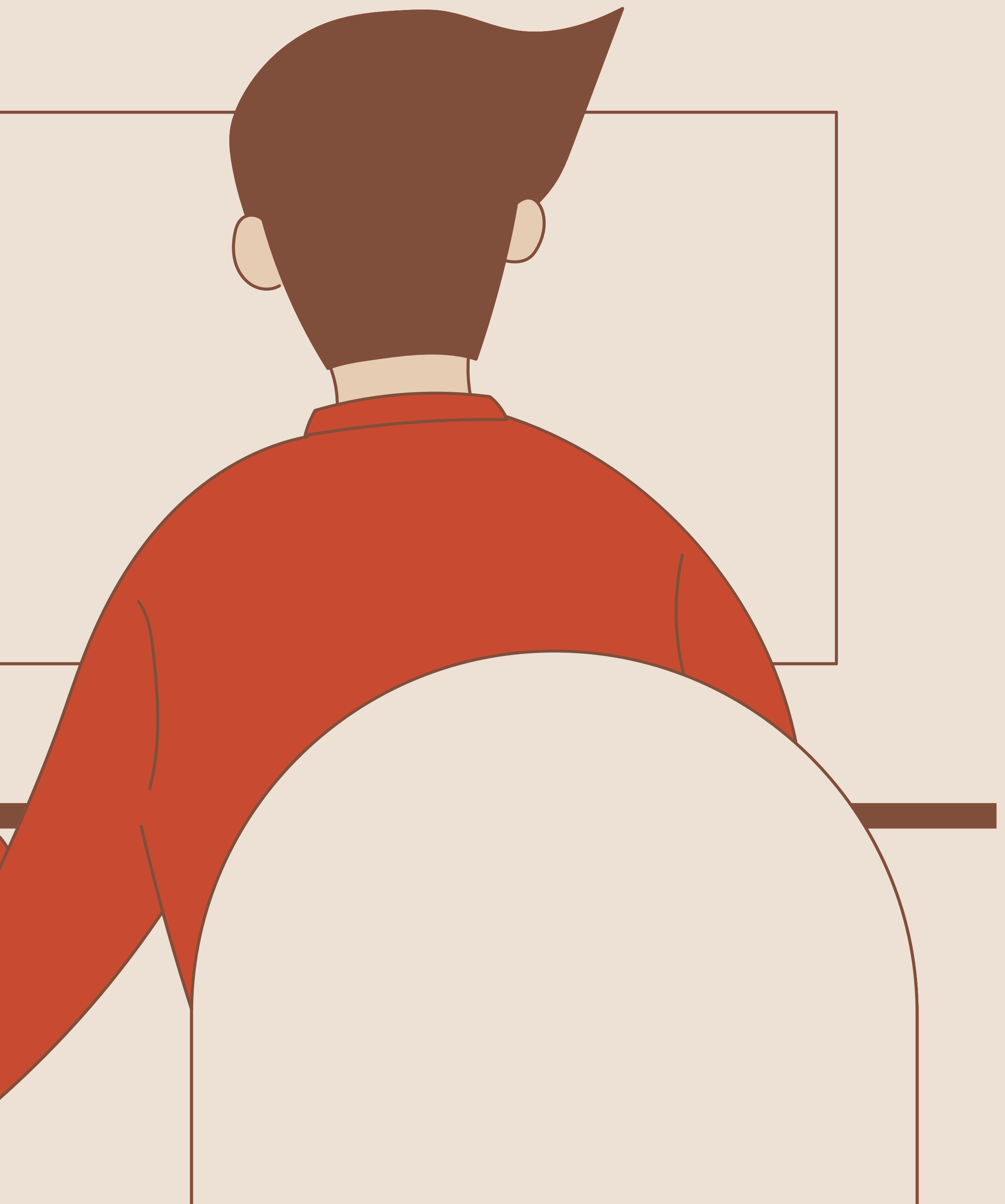
Students:

Hafsa Khurram
Durre Sakina Somji
Binish Fatima Basathia
Faiz Haseeb

Proffessor

Dr. Saleha Raza

Table of Contents

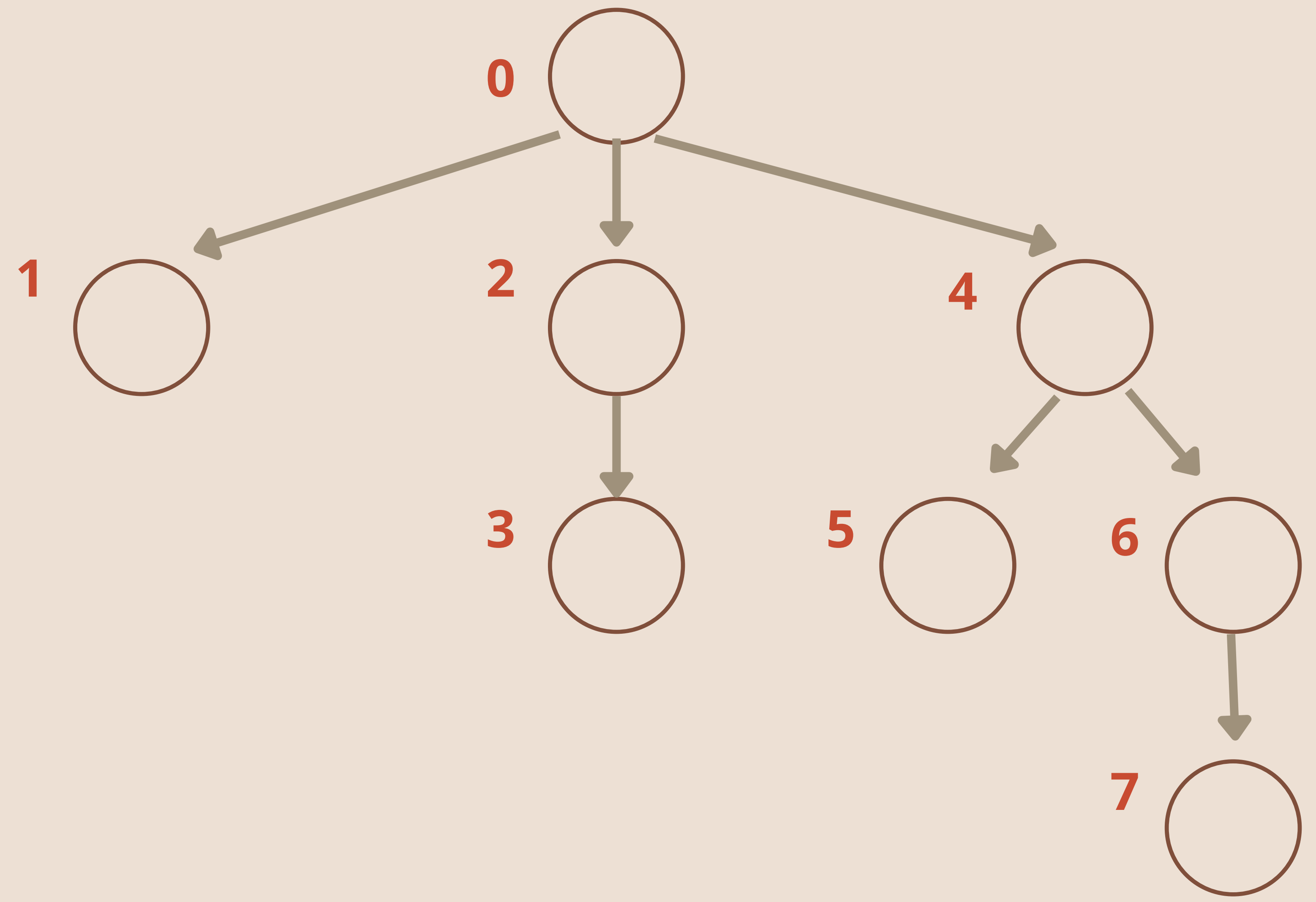


A Fenwick tree or Binary indexed tree is a data structure that can efficiently update elements and calculate prefix sums in a table of numbers.

A Fenwick tree is efficient when the sum of a data structure is being assessed.

Representation

A Fenwick tree can be represented in the following way:



The time complexity of building this from scratch is O(nlogn)

How do you decide what indexes are represented on a level?

So for example with 4 we will take the binary representation of the number

$(4)_{10} = (100)_2$

and then flip right most zeros one by one to make child nodes

What is represented on each index?

Each node represents the commutative sum of the array from the parent index to the child index.

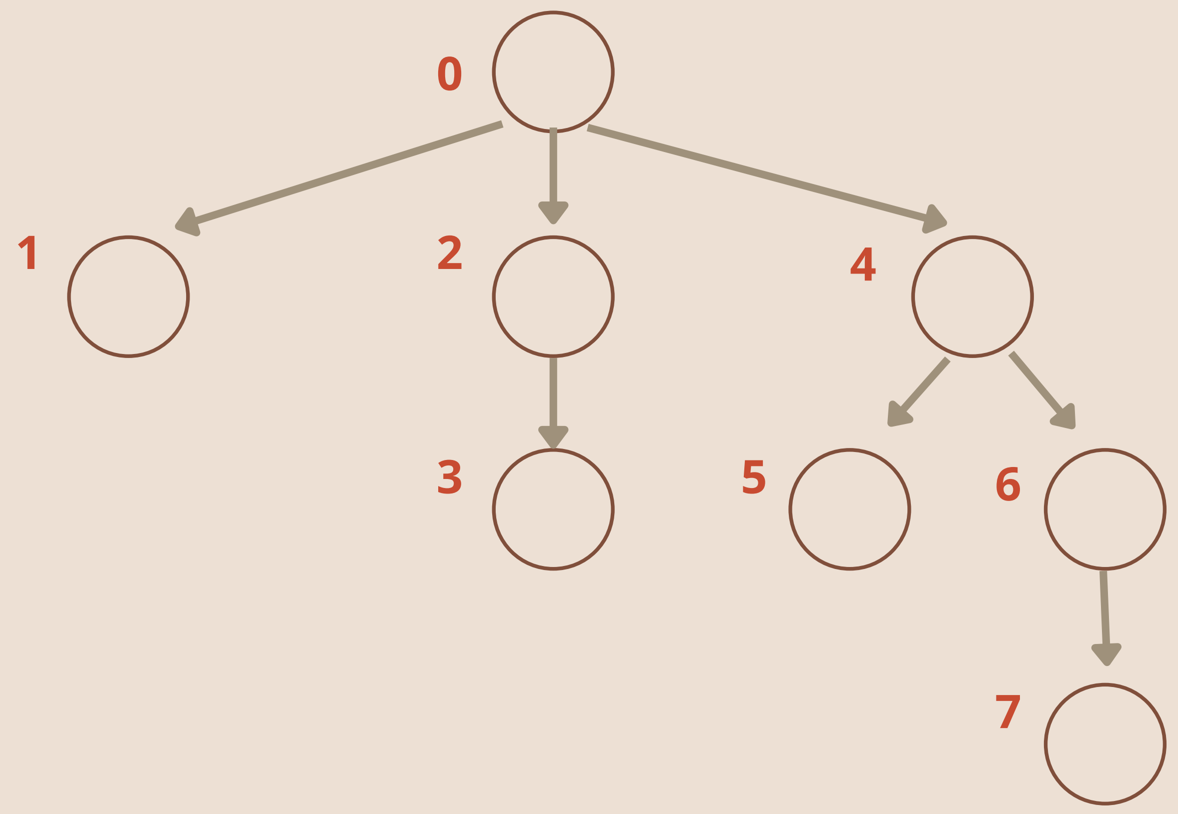
4 = 100

flip_1 = 101 -> 5

flip_2 = 110 -> 6

6 = 110

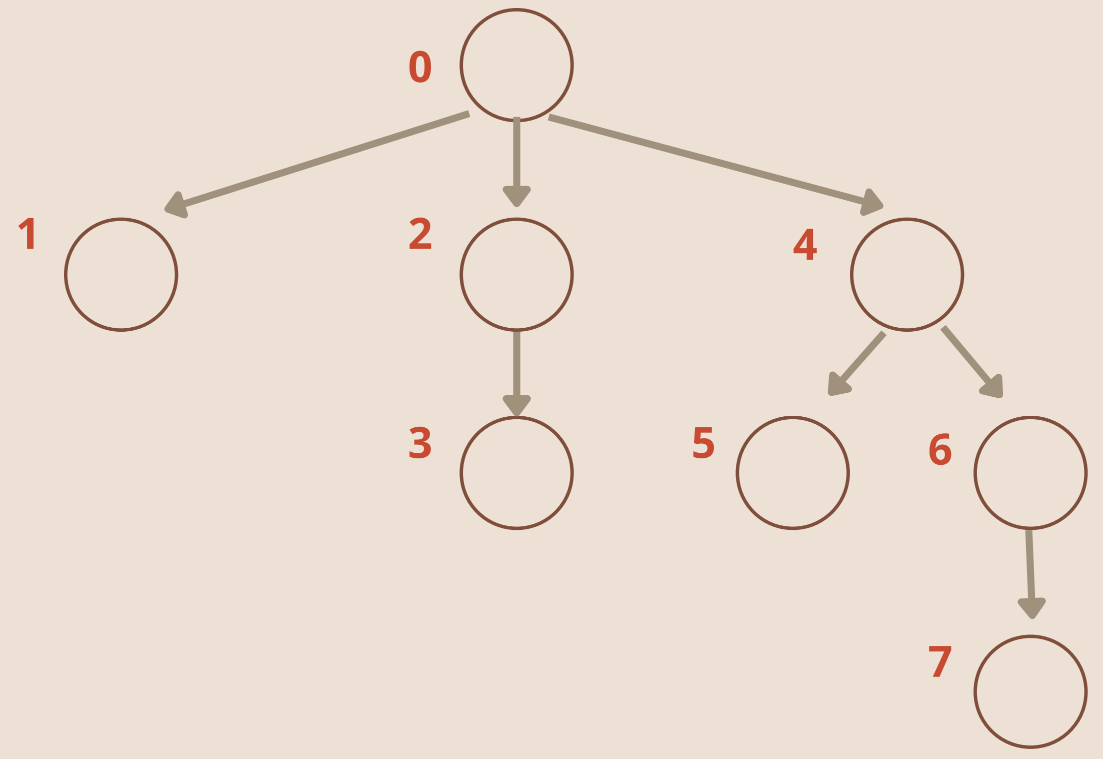
flip_1 = 111 -> 7



Updating an Index

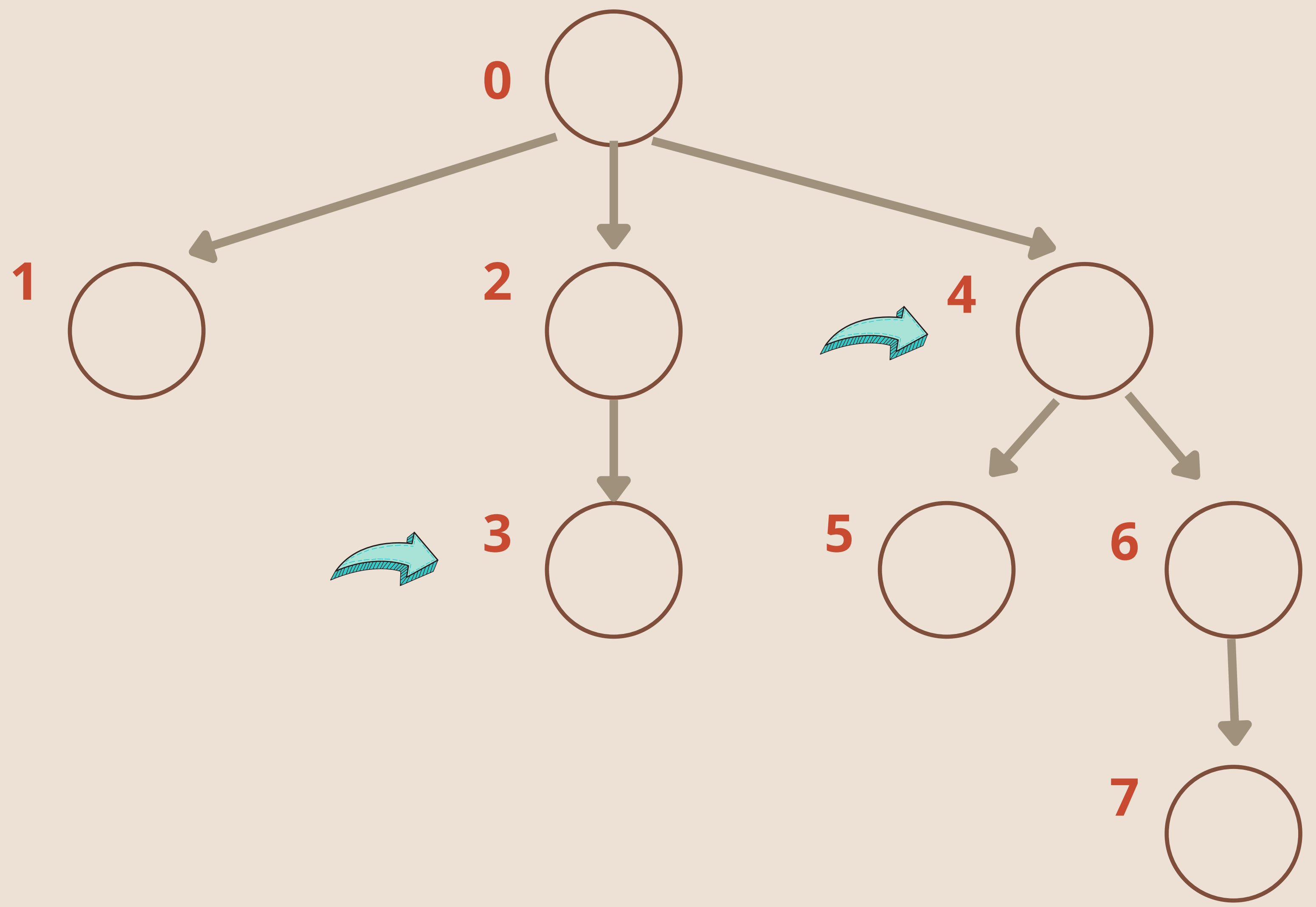
To update an index we update the index in question and then update the following indexes impacted so for instance while updating 4 we will update on the tree (4+1) = 5, 6, (sibling nodes), and then the next nodes of the zeroth level. The time complexity for this is O(log n).

```
update(x, val)
1) Initialize the current index as x+1.
2) Do the following while the current index is smaller than or equal to n.
...a) Add the val to BITree[index]
...b) Go to next element of BITree[index]. The next element can be obtained by incrementing the last set bit of the current index, i.e., index = index + (index & (-index))
```



Updating an Index

For example in the following example, 2 on the array and 3 on the tree are updated.



3 = 011

2's = 101

3 & 2's[3] = 001

011 + 001 = 4

4 = 100

2's = 100

3 & 2's[3] = 100

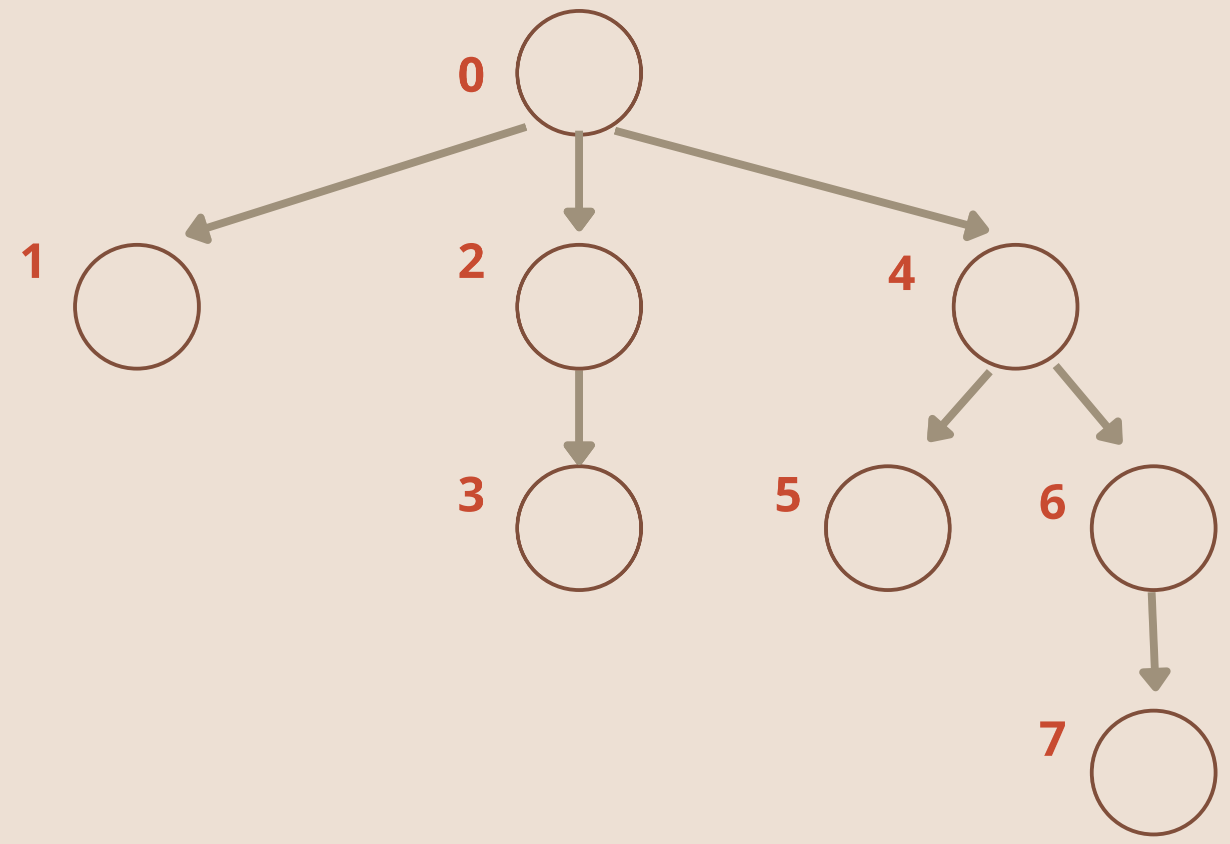
100 + 100 = 8

The positions three and four are updated

Get Commutative Sum

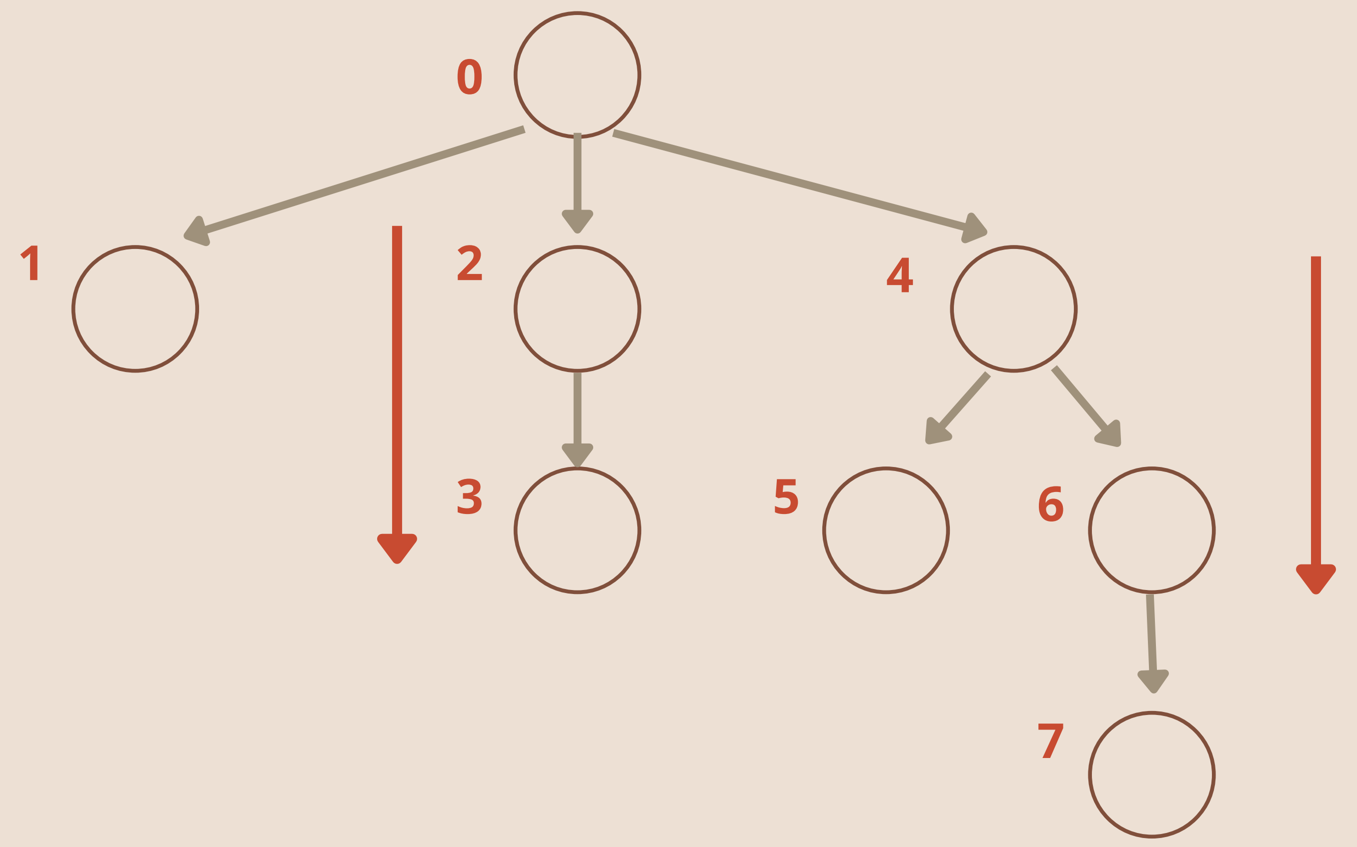
The Fenwick tree has another main operation which is basically to get the commutative sum between two indexes.

```
etSum(x):
1) Initialize the output sum as 0, the current index as x+1.
2) Do following while the current index is greater than 0.
...a) Add tree[index] to sum
...b) Go to the parent of tree[index]. The parent can be obtained by removing
the last set bit from the current index, i.e., index = index – (index & (-index))
3) Return sum.
```



Get Commutative Sum

For example, if we are getting the commutative sum from (2,5)



x = Tree[2] + Tree [3]

y = Tree [4] + Tree[6]

Sum = y - x

Time Complexity = O(logn)

Commutative Frequency Index

Find the cumulative sum of an array up to an index.

Array = [1 , 2 , 4 , 6 , 9 , 3 , 2 , 5]

Cumulative sum till index 2 = Array[0] + Array[1] + Array[2]

1 + 2 + 4 = 7

Cumulative sum till index 5 = Array[0] + Array[1] + Array[2] + Array[3] + Array[4] + Array[5]

1 + 2 + 4 + 6 + 9 + 3 = 25

Retrieval time complexity = O(1)

Update time complexity = O(n)

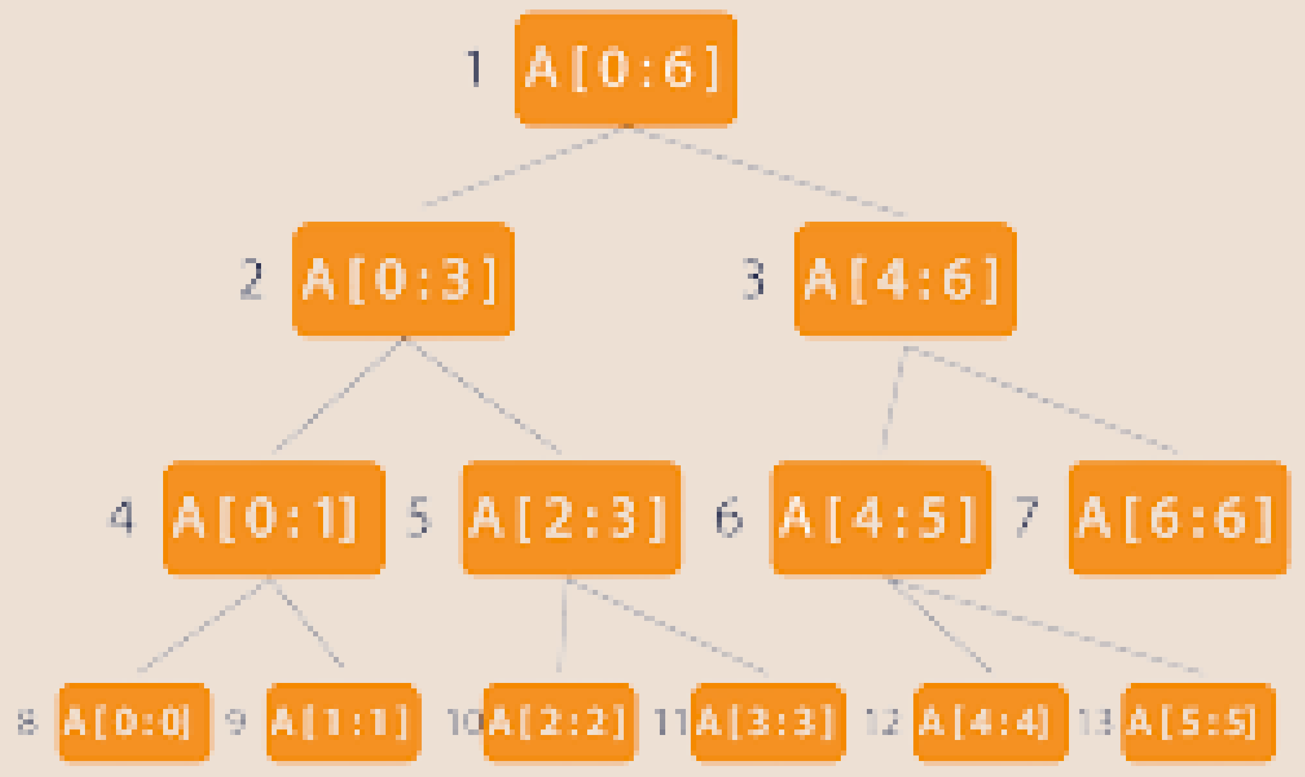
Similar Data Structures

Obviously, this is not the only option for prefix sums in fact we have the following other mechanisms used popularly:

- Segment Trees
- Sum Trees
- Square Root Decomposition
- Sparse Table

Segment Tree

The segment tree is a type of data structure from computational geometry. It is essentially a binary tree in whose nodes we store the information about the segments of a linear data structure such as an array. Furthermore, it helps us solve questions on range queries along with updates.



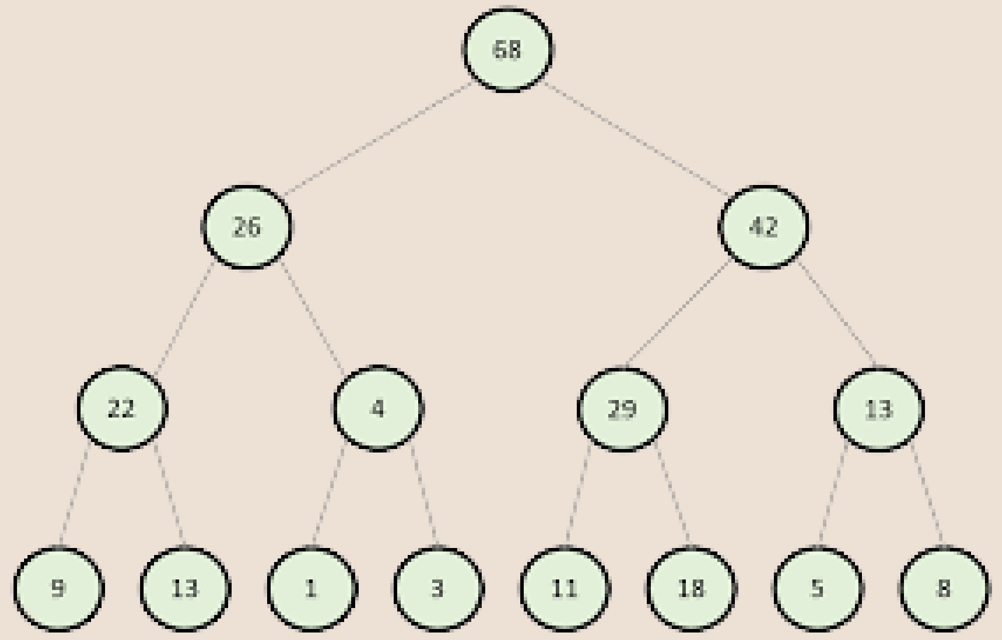
Segment Tree

Segment Tree vs Fenwick Tree

	Segment	Fenwick
Query	answers each query in O(logN)	answers each query in O(logN)
Preprocessing	preprocessing done in O(N)	preprocessing done in O(NlogN)
Pros	Pros: good time complexity.	Pros: the shortest code, good time complexity, requires less space
Cons	Cons: larger amount of code compared to the other data structures.	Cons: Fenwick tree can only be used for queries with L=1, so it is not applicable to many problems.

Sum Tree

- Sum Tree is a special kind of binary tree where the value of a parent node is equal to the sum of the values of its children.
- We can store the data in a Sum Tree and retrieve it accordingly with only the complexity of O(logn).
- The drawback to this however is the large storage space required



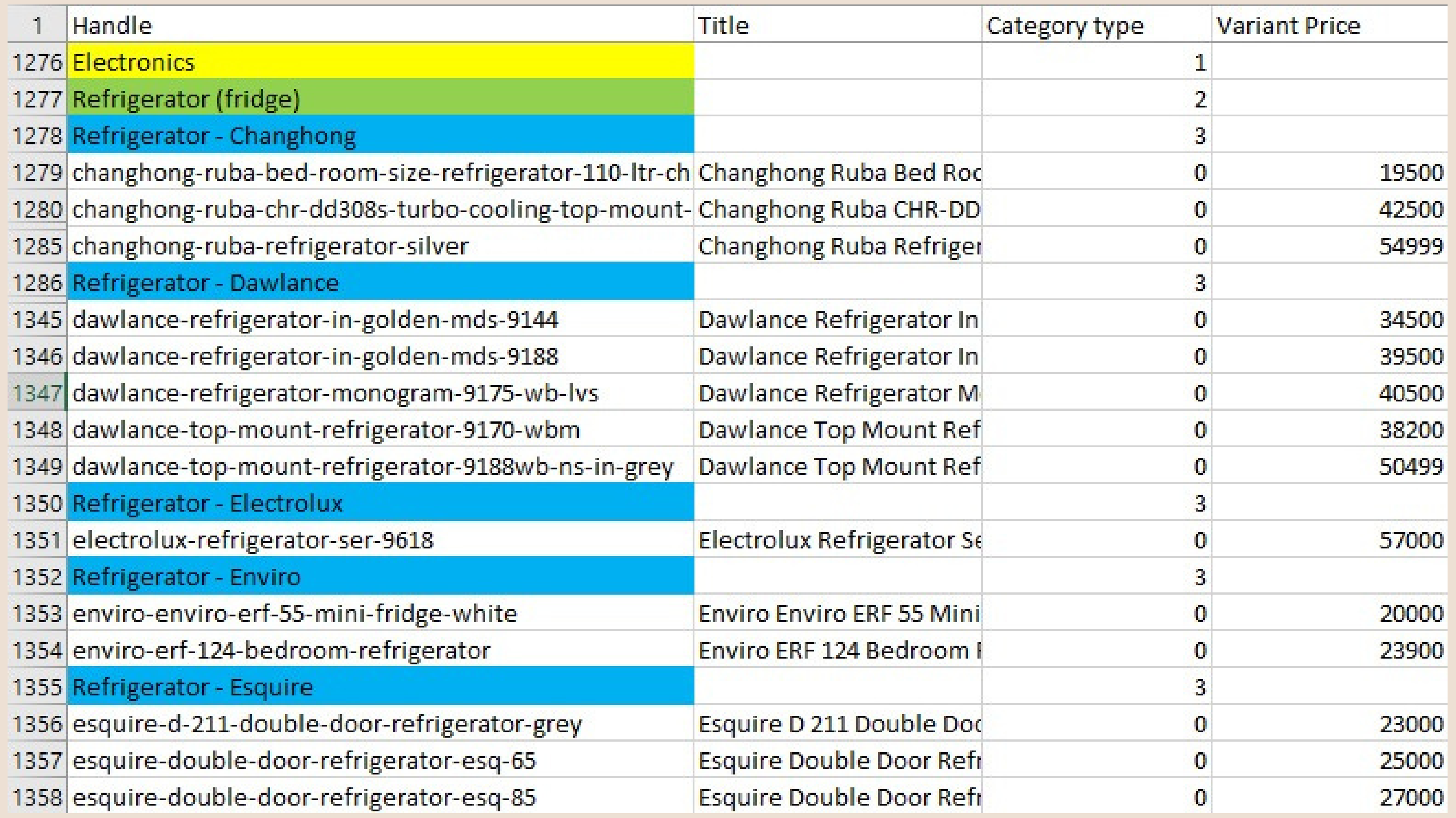
The Grocery Store Barometer

Just like a Barometer measures the pressure of a day this tool will measure the sales of a grocery store relative to different categories.
This tool will help shopkeepers assess how different items and categories are selling in their store, this is especially helpful when the items are too much for a shopkeeper to remember as the case study of Emaan Mart which has over 2000 items for sale.

Sorting

The data from the grocery store was sorted into the following categories:

$$\begin{tikzcd}
\text{Main Category} \arrow[d] \\
\text{Category} \arrow[d] \\
\text{Sub Category} \arrow[d] \\
\text{Products}
\end{tikzcd}$$



1	Handle	Title	Category type	Variant Price
1276	Electronics		1	
1277	Refrigerator (fridge)		2	
1278	Refrigerator - Changhong		3	
1279	changhong-ruba-bed-room-size-refrigerator-110-ltr-ch	Changhong Ruba Bed Roc	0	19500
1280	changhong-ruba-chr-dd308s-turbo-cooling-top-mount-	Changhong Ruba CHR-DD	0	42500
1285	changhong-ruba-refrigerator-silver	Changhong Ruba Refriger	0	54999
1286	Refrigerator - Dawlance		3	
1345	dawlance-refrigerator-in-golden-mds-9144	Dawlance Refrigerator In	0	34500
1346	dawlance-refrigerator-in-golden-mds-9188	Dawlance Refrigerator In	0	39500
1347	dawlance-refrigerator-monogram-9175-wb-lvs	Dawlance Refrigerator M	0	40500
1348	dawlance-top-mount-refrigerator-9170-wbm	Dawlance Top Mount Ref	0	38200
1349	dawlance-top-mount-refrigerator-9188wb-ns-in-grey	Dawlance Top Mount Ref	0	50499
1350	Refrigerator - Electrolux		3	
1351	electrolux-refrigerator-ser-9618	Electrolux Refrigerator Se	0	57000
1352	Refrigerator - Enviro		3	
1353	enviro-enviro-erf-55-mini-fridge-white	Enviro Enviro ERF 55 Mini	0	20000
1354	enviro-erf-124-bedroom-refrigerator	Enviro ERF 124 Bedroom	0	23900
1355	Refrigerator - Esquire		3	
1356	esquire-d-211-double-door-refrigerator-grey	Esquire D 211 Double Doc	0	23000
1357	esquire-double-door-refrigerator-esq-65	Esquire Double Door Refr	0	25000
1358	esquire-double-door-refrigerator-esq-85	Esquire Double Door Refr	0	27000

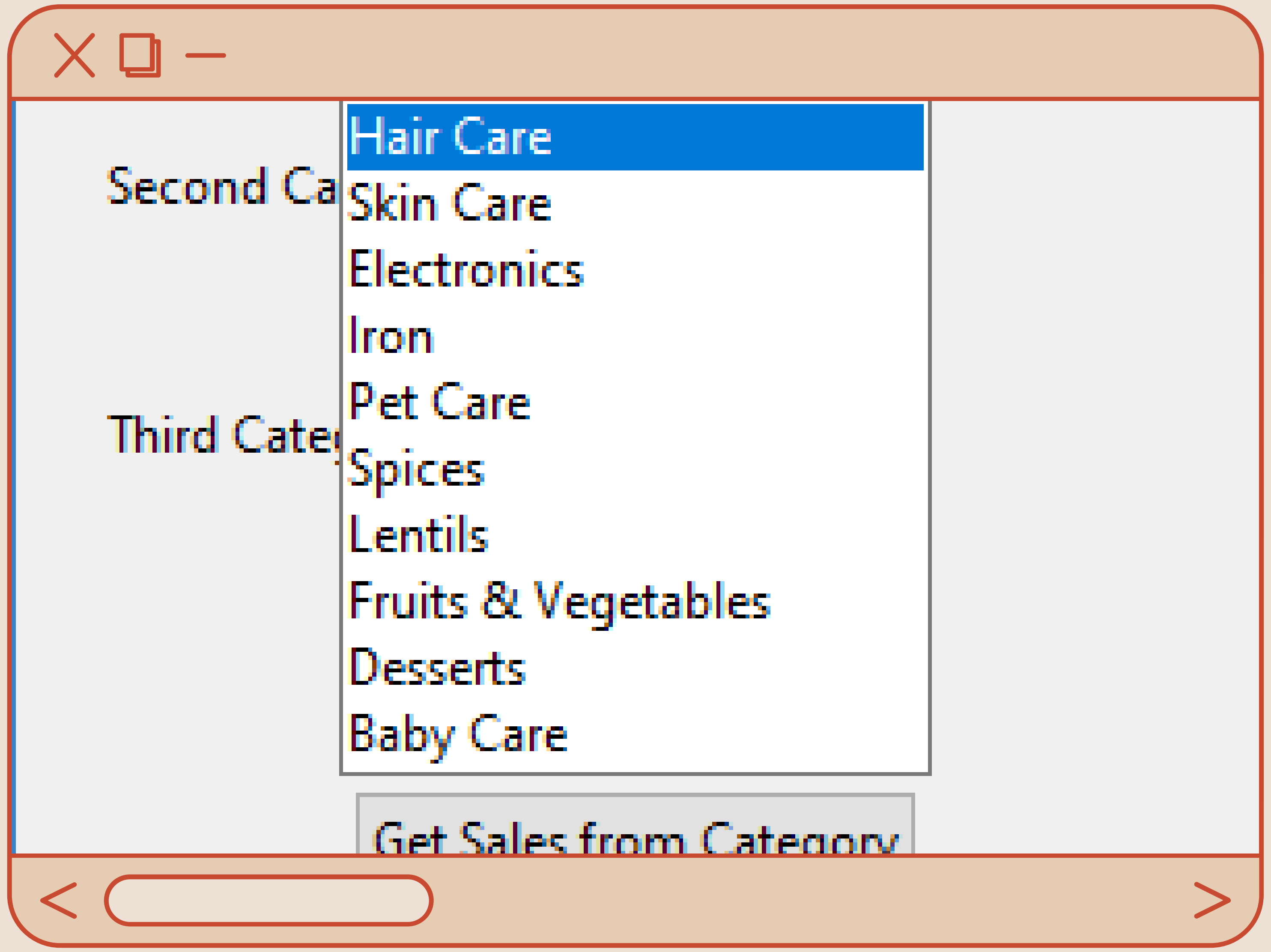
Backend Summary

This data was then cross-referenced to a particular index so that the commutative sum (ie. sum over one of the sorting levels could be found as well as updation each time a sale was made.

This acts as a literal barometer telling the shopkeeper where the sales pressure is more or less!

Front End

We used GUIs to make the Barometer user friendly



HABIB UNIVERSITY

2022 May 18

Thank you for listening!

Let's see this in action

Students:

Hafsa Khurram
Durre Sakina Somji
Binish Fatima Basathia
Faiz Haseeb

Proffessor

Dr. Saleha Raza

Fenwick Trees

- https://cp-algorithms.com/sequences/rmq.html#solution
- https://www.hackerearth.com/practice/data-structures/advanced-data-structures/segment-trees/tutorial/
- Binary Indexed Tree or Fenwick Tree - GeeksforGeeks
- Dependent Drop Downs and List Boxes - Python Tkinter GUI Tutorial https://www.youtube.com/watch?v=bH9r3wM9Idw&ab_channel=Codemy.com
- Creating a button in tkinter. https://www.tutorialspoint.com/python3/tk_button.htm
- Listboxes in Tkinter https://www.tutorialspoint.com/python/tk_listbox.htm

Proffessor

Dr. Saleha Raza