Advanced Data Structures (COP5536)
Fall 2018

Harshal Magan Patil
UFID – 55528581
harshal.patil@ufl.edu

Problem Statement:

Implement a feature for the “DuckDuckGo” search engine where we are required to count the n most popular keywords that are used in their search engine at any time. We use a max priority structure i.e the Max Fibonacci heap to keep a track of the frequencies of keyword. We also will make use of a hash table where the keys are the searched words and the value is the frequency so that we can retrieve and put keywords with a amortized cost of O(1). The input will be provided in a file with the keyword and its corresponding frequency.

Instructions for executing the program:

1. Unzip the folder first
2. Cd into the folder with the name Patil_Harshal
3. Open the terminal here and type : *make*
4. This will compile the code and generate a file named “keywordcounter”
5. Now type in the terminal : *java keywordcounter input.txt*
6. The output will be generated in a file called output_file.txt

Structure of the program and its method prototypes:

The project has been divided into two main classes:

1. keywordcounter
2. fibonacciheap
3. Node

Node Class:

Lets start with the most easiest class i.e the Node class it stores the structure of the node of the Fibonacci heap nodes.

It stores all the attributes of the nodes:

1. val - maintains the frequency of the keyword
2. degree - number of children of the node
3. childCut - childcut value either true or false
4. word - stores the string
5. parent - pointer to the parent node
6. child - pointer to the child node
7. prev - pointer to the previous node in the doubly linked list
8. next - pointer to the the next node in the doubly linked list

Node(String word, int val)		
Description	Constructor of the Node class	
Parameters	Word	The keyword
	Val	The frequency of the keyword
Return value	None	

Keywordcounter class:

The keywordcounter class is where the main method resides and this class mainly defines the ways of parsing the input file and writing the output to the output file.

public static void main(String[] args)		
Description	Main method is the entry point of the program	
Parameters	String[] args	Takes the input file as the parameter
Return value	None	

public void read_file(BufferedReader input, BufferedWriter writer)		
Description	Reads from the input file the keywords	
Parameters	input	Pointer to the associated input file
	writer	Pointer to the associated input file
Return value	None	

public void write_file(BufferedWriter writer,Node node, String checks)		
Description	Method to write write out the top results of the search in the output file	
Parameters	writer	Pointer to output file
	node	Pointer to the node
	checks	Checks if a new line is to be inserted
Return value	None	

public keywordcounter()		
Description	Constructor of the keyword counter class creates the Fibonacci heap object and also initializes the HashMap	
Parameters	None	-
Return value	None	

MaxFibonacciHeap class:

This class is responsible for defining all the operations that need to be performed on the heap.

public Node insert(String word, int val)		
Description	Method to insert the (word,val) pair into the fibonacci heap and meld it with the heap	
Parameters	word	The keyword
	val	Frequency of the keyword
Return value	Returns the inserted object	

public Node removeMax()		
Description	Method to remove the max Node and meld the corresponding children with the already present roots and do pairwise combine	
Parameters	None	-
Return value	Returns the removed node	

public void increaseKey(Node pointer, int newFreq)		
Description	method to increase the value of a preexisting key also checks for childcut value and performs cascadingcut	
Parameters	pointer	Pointer to the node
	newFreq	Value of the new frequency
Return value	None	

public Node meld(Node temp1, Node temp2)		
Description	Method to meld two nodes	
Parameters	Temp1	Pointer to the first node
	Temp2	Pointer to the second node
Return value	Max of the two nodes passed to the function	

public void cascadingCut(Node pointer)		
Description	Method to perform the cascading cut operation on the Fibonacci heap	
Parameters	Pointer	Pointer to the node we removed
Return value	None	

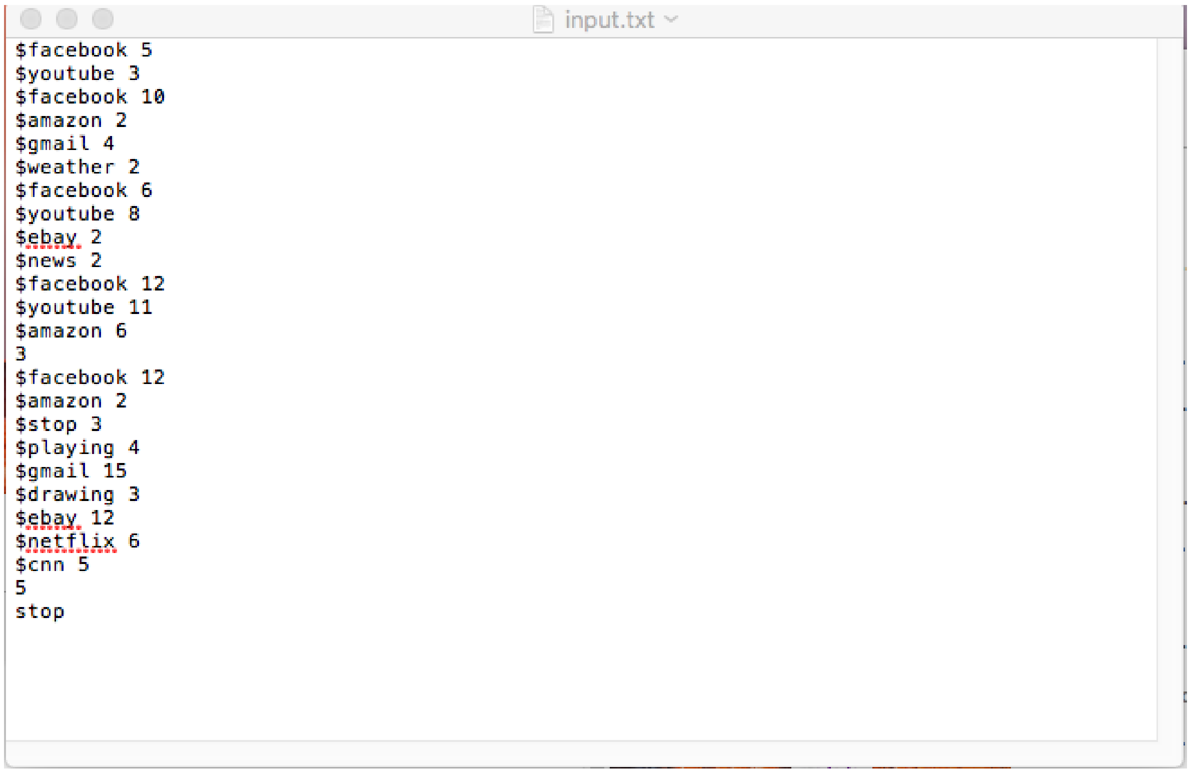
public void remove(Node pointer, Node upper)		
Description	Method to remove a node from its parent and then perform a meld operation with the already present list of roots	
Parameters	pointer	Pointer to the node we want to removed
	upper	Pointer to the parent node pointed by pointer
Return value	None	

public void pairwiseCombine()		
Description	Method used to merge two equal degree roots taking two roots at a time and also making sure the maxNode points to the maximum	
Parameters	None	-
Return value	None	

public void reinsert(List<Node> nodeList)		
Description	Method to reinsert all the the max nodes back into the fibonacci heap	
Parameters	nodeList	List of all the nodes that are to be inserted back into the Fibonacci heap
Return value	None	

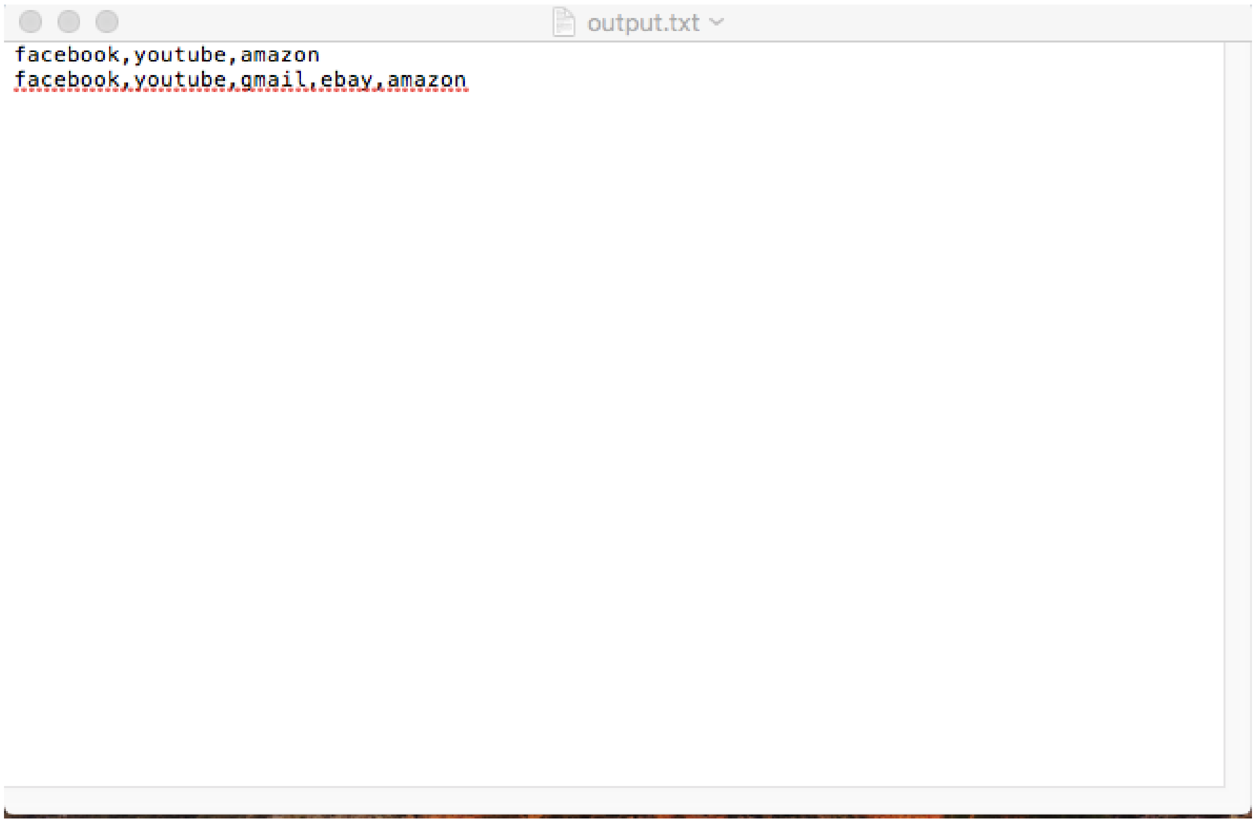
Sample Results:

Input file:


input.txt

```
$facebook 5
$youtube 3
$facebook 10
$amazon 2
$gmail 4
$weather 2
$facebook 6
$youtube 8
$ebay 2
$news 2
$facebook 12
$youtube 11
$amazon 6
3
$facebook 12
$amazon 2
$stop 3
$playing 4
$gmail 15
$drawing 3
$ebay 12
$netflix 6
$cnn 5
5
stop
```

Output file:



```
facebook,youtube,amazon
facebook,youtube,gmail,ebay,amazon
```

Conclusion:

The feature for finding the n most popular keywords to be searched has been implemented using the Max Fibonacci Heap data structure along with a HashMap. The feature enables us to find out the most popular keywords at point of time efficiently and fast.