CONTENTS

I. PROJECT DESCRIPTION

AIM OF THE PROJECT

The Aim of the project is to implement a system to count the “n” most popular keywords used in the search engine “DuckDuckGo” at any given time. Implementation is done using Max Fibonacci Heap structure to find out the “n” most popular keywords.

DATA STRUCTURES USED FOR IMPLEMENTATION

The project uses two data structures for the implementation:

- **Hash Table**: **Keywords** are the keys for the hash table and **hashvalue** are the frequencies associated with the keyword
- **Max Fibonacci Heap**: Used to keep track of the frequencies of Keywords. Keyword is the key for the hash table and hashvalue is pointer to the corresponding node in the Fibonacci heap.

FIBONACCI HEAP

Fibonacci heap consist of a collection of heap-ordered trees. It is a data structure implemented for priority queue operations. Compared to many other priority queue data structures it has a better amortized running time.

Cascading Cut in Fibonacci Heap: The number of cascading cuts during a sequence of heap operations is bounded by the number of delete and increase key operations.

REASON FOR USING MAX FIBONACCI HEAP

As keywords appear in the input keywords stream, increase key operation and extract max operations have to be performed many times to find out the “n” most popular keywords. Hence Max Fibonacci heap is the suitable data structure.

Time Complexity for Fibonacci Heap

	Amortized	Actual
Insert	O(1)	O(1)
Remove	O(logn)	O(n)
Remove Max(or Min)	O(logn)	O(n)
Meld	O(1)	O(1)
Increase Key(or Decrease)	O(1)	O(n)

PROGRAM STRUCTURE

The program consists of 2 java classes.

1) **keywordcounter** : This is the main class where input is read and the output is written.

2) **FibHeap** : This class is used to instantiate the object of node, methods and functions of the Heap class.

WORK FLOW

- Input file is read in **keywordcounter.class** and a hashmap named **hashmap_fib** is created.
- If the keyword is already present in the hashmap then its **hashvalue**(frequency) is increased
- If the keyword is not present then keywordcounter.class performs insert node operations on an instance of the **FibHeap.class** by creating new nodes using **FibHeap.FibNode**
- If the input is a **query** (digit) keywordcounter.class performs remove max(upto “n” times) and places the removed nodes in array **removedNodes**
- Output is printed in the outputfile.txt
- After printing the output the nodes are reinserted back into the hashmap.
- If the input is **stop** the program ends.

WORKING ENVIRONMENT

1. **IDE** : Eclipse(2018-09)

2. **Operating System**: Windows 10

3. **Hardware Requirement**:
   Hard Disk space: Minimum 4GB
   Memory: 512 MB

II. FUNCTION DESCRIPTION AND PROTOTYPE

The detailed overview of FibHeap.java

a) public int isEmpty(FibNode node)

Description	This function checks whether a Node of type FibNode exists or not. If the node exists (not equal to null) the flag is set to 1 else if node does not exists in the Fibonacci heap then flag value is 0.
Parameter	Node of type FibNode
Return value	Integer value is the return value. Value of flag is returned

b) private void sibling_link(FibNode m, FibNode n)

Description	Insert node n between node m and node m.right. Do n.right = m.right; Check if n.right is null or not. If n.right is not null then n.right.left = n. Else if it is null do nothing.This will ensure the properties of doubly linked list among sibling nodes.
Parameter	Two nodes m and n of type FibNode are the input parameters.
Return value	Void(nothing is returned)

c) private void left_right_merge(FibNode node)

Description	This function is used to remove the input node by connecting its siblings together(node.left.right = node.right; node.right.left = node.left)
Parameter	Node of type FibNode is the input parameter. This node has to be deleted.
Return value	Void(nothing)

d) private FibNode compare_hash(FibNode node, FibNode maxnode)

Description	This function is used to compare the hashvalue (frequencies) of two nodes: node and maxnode. It returns the node which has highest hashvalue
Parameter	Input parameter are two nodes of type FibNode whose hash values are to be compared
Return value	FibNode whose hashvalue is highest is returned

e) public void increasevalue(FibNode node, int value)

Description	This function is used to increase the hashvalue(frequency or value) of keyword (FibNode node) .If the parent of the node(keyword) is not null and hashvalue of node is greater than that of its parent, then node is cut and cascading cut is performed on its parent if its childcut_value is true before the cut. If the hashvalue of the node is greater than the existing maxFibnode then update this node as maxFibnode using the function compare_hash
Parameter	Input parameter is the keyword (node of type FibNode) whose hashvalue has to be increased to value.
Return value	Void (nothing)

f) public void insertnode(FibNode newnode)

Description	This function is used to insert a new node of type FibNode into Fibonacci heap. First check if the maxFibnode is null(Empty heap).If the maxFibnode is null make this new node to be inserted as max . If the max exists then link this new node as right sibling to maxFibnode using sibling_link function. After insertion check the hashvalue of both newly inserted node and maxFibnode and make the highest hashvalue one as max. After the insertion the total numbers of nodes are increased by 1.
Parameter	Input parameter is the keyword(node of type FibNode) which has to be inserted into the Fibonacci heap
Return value	Void(nothing). The newnode is inserted into the Fibonacci heap.

g) public FibNode maxnode_remove()

Description	This function is used to remove the maxFibnode. The children are removed one by one from the maxnode and are added to the root list of heap. On these nodes we call function degreewiseMerge().
Parameter	Null
Return value	maxFibnode whose hashvalue is highest is returned

h) public void Nodecut(FibNode p, FibNode gp)

Description	This function cuts the node p from gp, and decreases the degree of node gp. It adds the node p to root list of heap and sets its childcut_value to false. It also checks if node gp degree is 0, it sets gp.child value to Null. If degree is not 0, then check if the node p was assigned gp.child and then change the value of gp.child to sibling of node p
Parameter	Input parameter are nodes p and gp(nodes are of type FibNode).
Return value	Void (nothing).

i) public void Node_CascadingCut(FibNode newnode)

Description	This function is used to perform cascading cut based on the childcut _value. If the newnode has parent then perform cascading cut based on childcut value up to the root. If childcut value of newnode is true (it has previously lost a child) then perform Nodecut() on it and perform Cascading cut on its parent. If childcut value of newnode is false (it did not lose child before this) set its childcut_value to true.
Parameter	Input parameter is the node of type FibNode.
Return value	Void (nothing)

j) public void child_create(FibNode newchild, FibNode newparent)

Description	This function is used to make node newchild child of the node newparent. This function is useful in Merge_Pairwise()
Parameter	Input parameter are two nodes of type FibNode
Return value	Void (nothing)

k) public void Merge_Pairwise()

Description	This function performs degree wise merge. It melds those nodes in the root list of the heap who have same degree using the function child_create.
Parameter	Null
Return value	Void

The detailed overview of fibNode in FibHeap.java

Class Variable	**Type**	**Description**
keyword	String	Contains the input keyword.
hashvalue	Integer	Signifies the frequency of corresponding keyword.
Degree	Integer	Signifies the number of children a node has (The number of nodes that a node can have in its next level)
childcut_value	Boolean	If childcut_value is true, it means that node has previously lost its child. If the value is false it means that no child has been removed from that node.
left	FibNode	Points to the left node in the doubly linked circular list.
right	FibNode	Points to the right node in the doubly linked circular list.
parent	FibNode	Points to the parent node
child	FibNode	Points to the child node.

a) FibNode(String keyword,Long hashvalue)

Description	Constructor which initializes the keyword and its hashvalue
Parameter	String and Long
Return value	-

b) public String getKeyword()

Description	Returns the keyword of the node
Parameter	Null
Return value	String

c) public long gethashvalue()

Description	Returns the hashvalue associated with the keyword
Parameter	Null
Return value	long

d) public FibNode getParent()

Description	Returns the parent of the node(keyword)
Parameter	Null
Return value	FibNode

e) public void setParent(FibNode parent)

Description	Assign the parent to the keyword
Parameter	FibNode
Return value	Void

f) public FibNode getLeft()

Description	Returns the left sibling of the node
Parameter	Null
Return value	FibNode

g) public void setLeft(FibNode left)

Description	Assign the left sibling to the node
Parameter	FibNode
Return value	Void

h) public FibNode getRight()

Description	Returns the right sibling of the node
Parameter	Null
Return value	FibNode

i) public void setRight(FibNode right)

Description	Assign the right sibling to the node
Parameter	FibNode
Return value	Void

j) public FibNode getChild()

Description	Returns the child of the node
Parameter	Null
Return value	FibNode

k) public void setChild(FibNode child)

Description	Assign the child to the node
Parameter	FibNode
Return value	void

l) public boolean isChildCut()

Description	Returns the child cut value of the node
Parameter	Null
Return value	boolean

m) public void setChildCut(boolean childCut)

Description	Returns the child cut value of the node
Parameter	boolean
Return value	void

The detailed overview of keywordcounter.java

This class contains the main method, **public static void** main (String [ ] args)

- Reading input is done in try block. In the main method, input is read from the input_file .In the input file, keywords appear one per each line and start with $ sign and “n” most popular keywords are written to the output_file.

- HashMap<String,FibHeap.FibNode> hashmap_fib = new HashMap() is used to create a Hash table named hashmap_fib. Keywords are key in hashmap_fib and their corresponding frequencies are termed as hash_value

- Pattern matching in Java is to match a regular expression (pattern) against input text. The java package java.util.regex is used for pattern matching. Pattern.compile () is used to match the regular expression against input text.

- The Matcher instance is used to find matches of the pattern in the input text. The Matcher class has a matches () method that tests if the specified pattern matches the input text.

  Three kinds of pattern are specified for query, keyword and stop and matched with the input string.

  pattern_1: This pattern is to match query

  pattern_2: This pattern is to match input keyword

  pattern_3: This pattern is to match stop

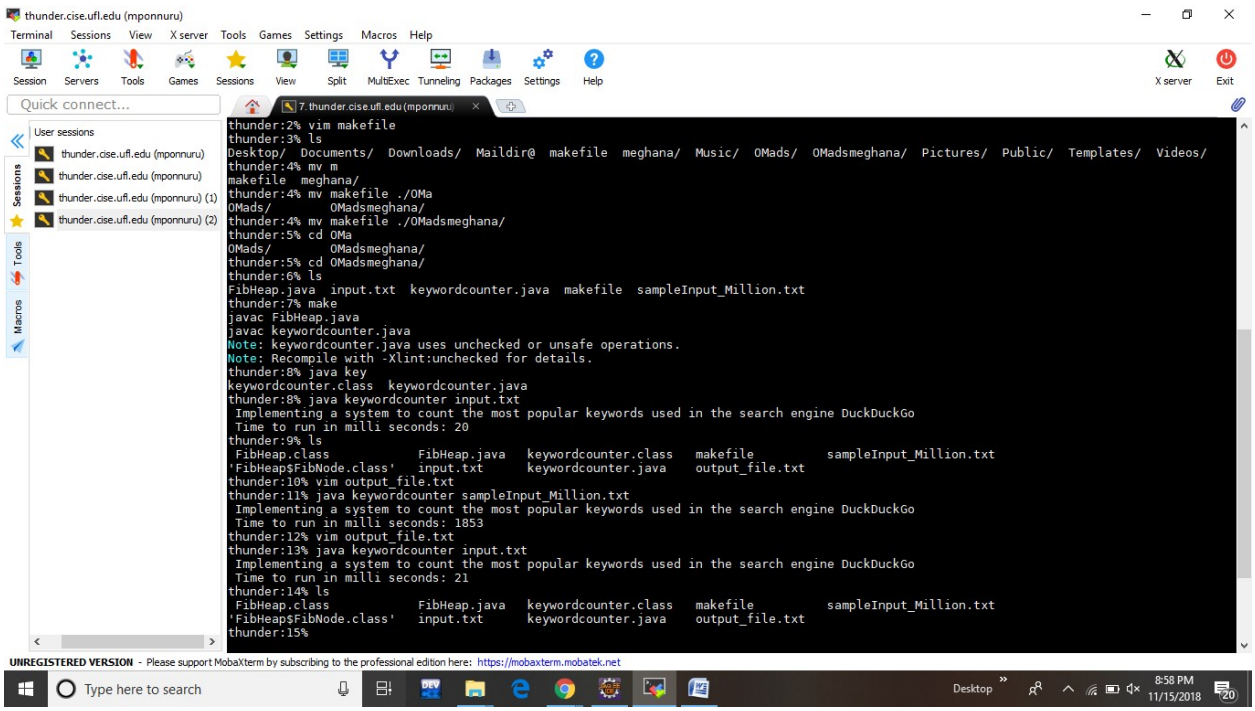
- When **stop** appears the loop breaks
- When **query("n")** appears remove the "n" keywords with highest hashvalues using hashmap_fib. Remove and put the removed nodes in array named removedNodes. After performing the query, extracted keywords are reinserted into hash table hashmap_fib
- When new keyword appears, its keyword and hashvalue are correspondingly stored in hashtable named hashmap_fib, but when old keyword appears, its old hashvalue is incremented by new hashvalue.

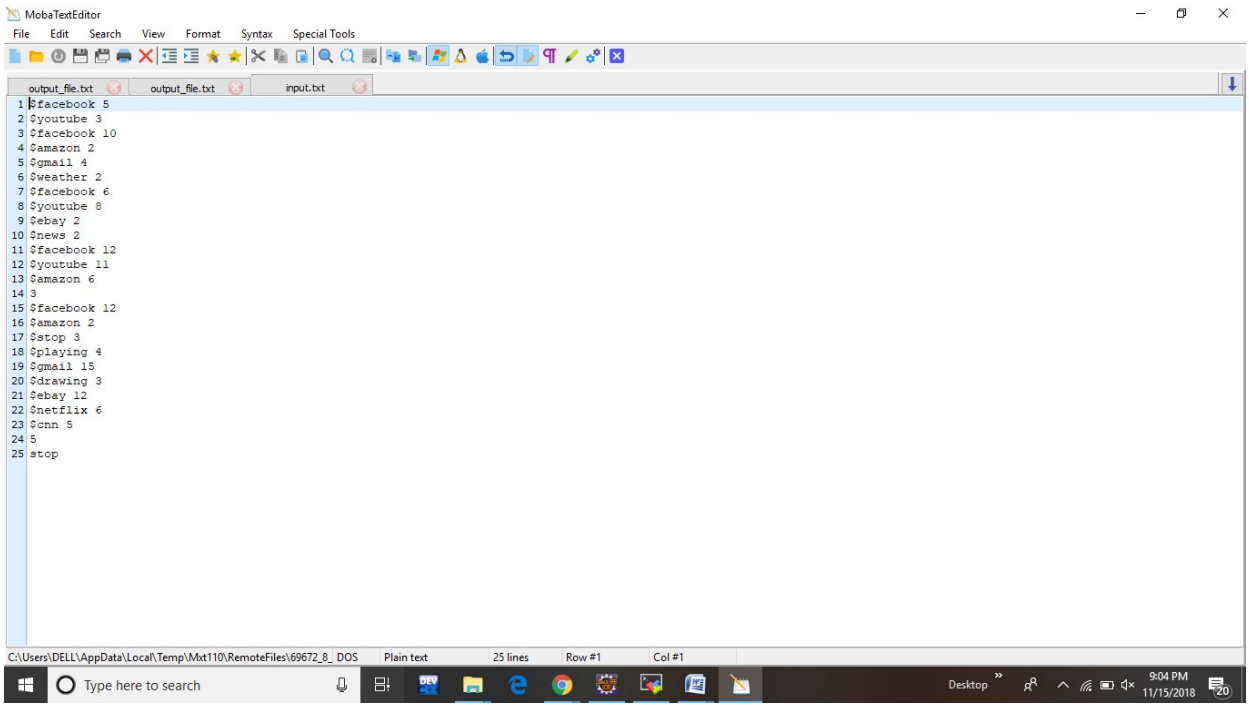
COMPILE AND RUN INSTRUCTIONS

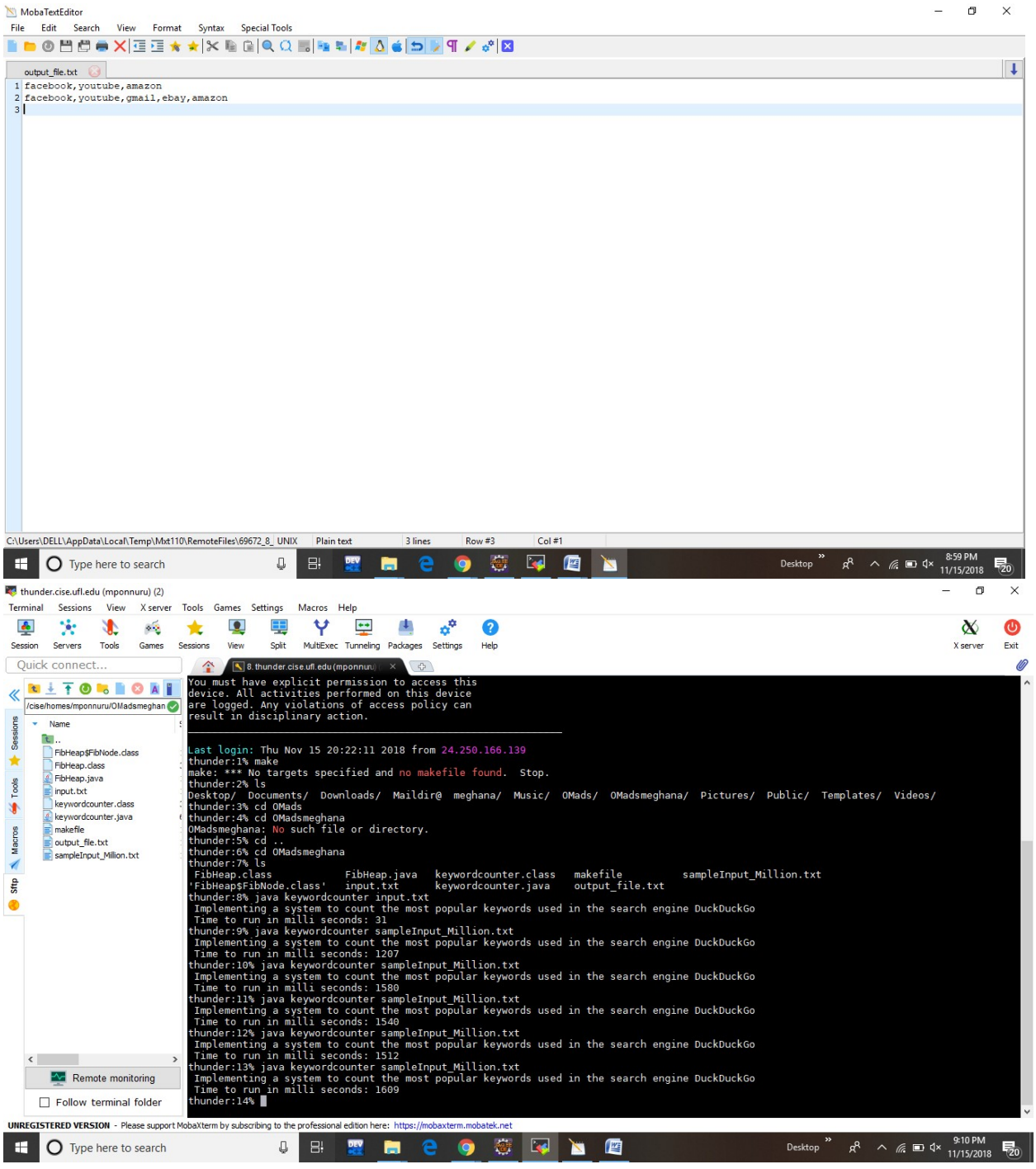
- The project has been tested and compiled on thunder.cise.ufl.edu
- Open the cmd and change directory to where the files are located.
  On cmd run the make file.
  java keywordcounter file_name

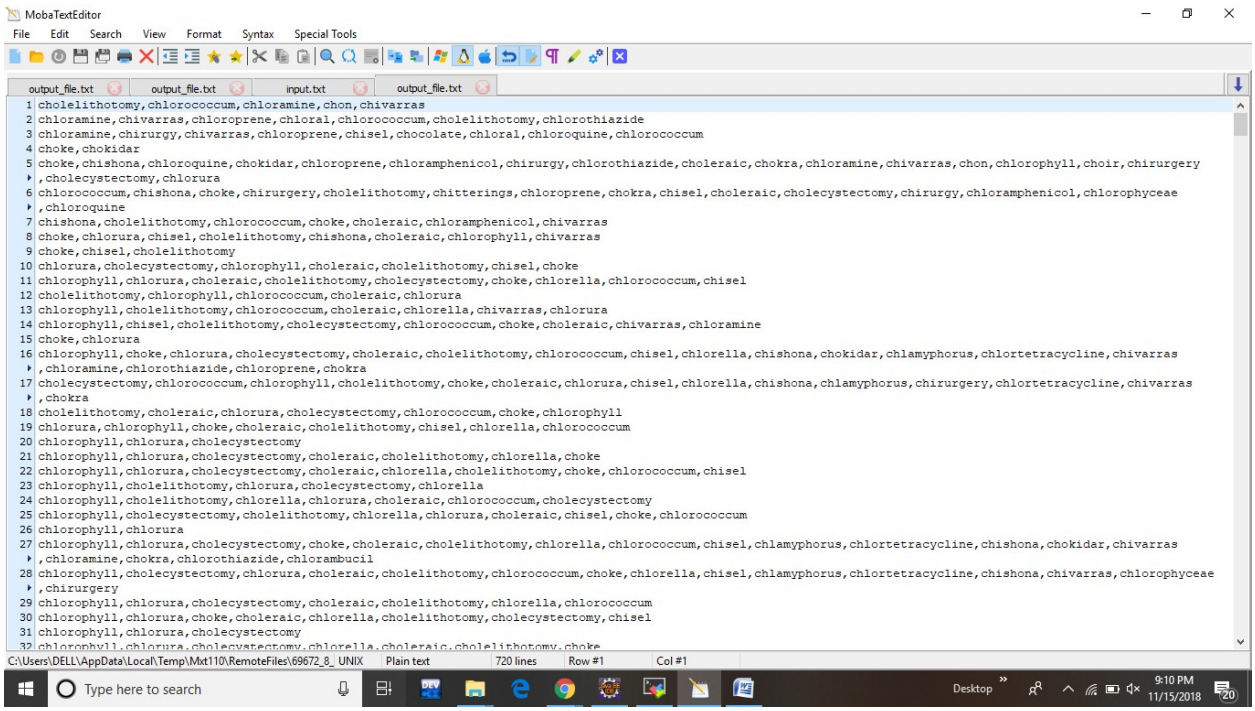
RESULTS

- The order of the output can be different as there are many nodes with same hashvalues.
- The code was run for 1 million input and time required was found to be 1609 milli seconds.


thunder.cise.ufl.edu (mponnuru)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
User sessions
thunder.cise.ufl.edu (mponnuru)
thunder.cise.ufl.edu (mponnuru)
thunder.cise.ufl.edu (mponnuru) (1)
thunder.cise.ufl.edu (mponnuru) (2)
thunder:2% vim makefile
thunder:3% ls
Desktop/ Documents/ Downloads/ Maildir@ makefile meghana/ Music/ OMads/ OMadsmeghana/ Pictures/ Public/ Templates/ Videos/
thunder:4% mv m
makefile meghana/
thunder:4% mv makefile ./OMa
OMads/ OMadsmeghana/
thunder:4% mv makefile ./OMadsmeghana/
thunder:5% cd OMa
OMads/ OMadsmeghana/
thunder:5% cd OMadsmeghana/
thunder:6% ls
FibHeap.java input.txt keywordcounter.java makefile sampleInput_Million.txt
thunder:7% make
javac FibHeap.java
javac keywordcounter.java
Note: keywordcounter.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
thunder:8% java key
keywordcounter.class keywordcounter.java
thunder:8% java keywordcounter input.txt
Implementing a system to count the most popular keywords used in the search engine DuckDuckGo
Time to run in milli seconds: 20
thunder:9% ls
FibHeap.class FibHeap.java keywordcounter.class makefile sampleInput_Million.txt
'FibHeap$FibNode.class' input.txt keywordcounter.java output_file.txt
thunder:10% vim output_file.txt
thunder:11% java keywordcounter sampleInput_Million.txt
Implementing a system to count the most popular keywords used in the search engine DuckDuckGo
Time to run in milli seconds: 1853
thunder:12% vim output_file.txt
thunder:13% java keywordcounter input.txt
Implementing a system to count the most popular keywords used in the search engine DuckDuckGo
Time to run in milli seconds: 21
thunder:14% ls
FibHeap.class FibHeap.java keywordcounter.class makefile sampleInput_Million.txt
'FibHeap$FibNode.class' input.txt keywordcounter.java output_file.txt
thunder:15%
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
Type here to search
Desktop
8:58 PM
11/15/2018


MobaTextEditor
File Edit Search View Format Syntax Special Tools
output_file.txt
output_file.txt
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
C:\Users\DELL\AppData\Local\Temp\Mxt110\RemoteFiles\69672_8_ DOS
Plain text
25 lines
Row #1
Col #1
Type here to search
Desktop
9:04 PM
11/15/2018


MobaTextEditor
File Edit Search View Format Syntax Special Tools
output_file.txt
1 facebook,youtube,amazon
2 facebook,youtube,gmail,ebay,amazon
3
C:\Users\DELL\AppData\Local\Temp\Mxt110\RemoteFiles\69672_8_ UNIX Plain text 3 lines Row #3 Col #1
Type here to search
Desktop
8:59 PM
11/15/2018
thunder.cise.ufl.edu (mponnuru) (2)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
X server Exit
Quick connect...
8. thunder.cise.ufl.edu (mponnuru)
/cise/homes/mponnuru/OMadsmeghan
Name
..
FibHeap$FibNode.class
FibHeap.class
FibHeap.java
input.txt
keywordcounter.class
keywordcounter.java
makefile
output_file.txt
sampleInput_Million.txt
Sessions
Tools
Macros
Sftp
Remote monitoring
Follow terminal folder
You must have explicit permission to access this
device. All activities performed on this device
are logged. Any violations of access policy can
result in disciplinary action.
Last login: Thu Nov 15 20:22:11 2018 from 24.250.166.139
thunder:1% make
make: *** No targets specified and no makefile found. Stop.
thunder:2% ls
Desktop/ Documents/ Downloads/ Maildir@ meghana/ Music/ OMads/ OMadsmeghana/ Pictures/ Public/ Templates/ Videos/
thunder:3% cd OMads
thunder:4% cd OMadsmeghana
OMadsmeghana: No such file or directory.
thunder:5% cd ..
thunder:6% cd OMadsmeghana
thunder:7% ls
FibHeap.class FibHeap.java keywordcounter.class makefile sampleInput_Million.txt
'FibHeap$FibNode.class' input.txt keywordcounter.java output_file.txt
thunder:8% java keywordcounter input.txt
Implementing a system to count the most popular keywords used in the search engine DuckDuckGo
Time to run in milli seconds: 31
thunder:9% java keywordcounter sampleInput_Million.txt
Implementing a system to count the most popular keywords used in the search engine DuckDuckGo
Time to run in milli seconds: 1207
thunder:10% java keywordcounter sampleInput_Million.txt
Implementing a system to count the most popular keywords used in the search engine DuckDuckGo
Time to run in milli seconds: 1580
thunder:11% java keywordcounter sampleInput_Million.txt
Implementing a system to count the most popular keywords used in the search engine DuckDuckGo
Time to run in milli seconds: 1540
thunder:12% java keywordcounter sampleInput_Million.txt
Implementing a system to count the most popular keywords used in the search engine DuckDuckGo
Time to run in milli seconds: 1512
thunder:13% java keywordcounter sampleInput_Million.txt
Implementing a system to count the most popular keywords used in the search engine DuckDuckGo
Time to run in milli seconds: 1609
thunder:14%
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
Type here to search
Desktop
9:10 PM
11/15/2018


MobaTextEditor
File Edit Search View Format Syntax Special Tools
output_file.txt
output_file.txt
input.txt
output_file.txt
1 cholelithotomy,chlorococcum,chloramine,chon,chivarras
2 chloramine,chivarras,chloroprene,chloral,chlorococcum,cholelithotomy,chlorothiazide
3 chloramine,chirurgy,chivarras,chloroprene,chisel,chocolate,chloral,chloroquine,chlorococcum
4 choke,chokidar
5 choke,chishona,chloroquine,chokidar,chloroprene,chloramphenicol,chirurgy,chlorothiazide,choleraic,chokra,chloramine,chivarras,chon,chlorophyll,choir,chirurgery
,cholecystectomy,chlorura
6 chlorococcum,chishona,choke,chirurgery,cholelithotomy,chitterings,chloroprene,chokra,chisel,choleraic,cholecystectomy,chirurgy,chloramphenicol,chlorophyceae
,chloroquine
7 chishona,cholelithotomy,chlorococcum,choke,choleraic,chloramphenicol,chivarras
8 choke,chlorura,chisel,cholelithotomy,chishona,choleraic,chlorophyll,chivarras
9 choke,chisel,cholelithotomy
10 chlorura,cholecystectomy,chlorophyll,choleraic,cholelithotomy,chisel,choke
11 chlorophyll,chlorura,choleraic,cholelithotomy,cholecystectomy,choke,chlorella,chlorococcum,chisel
12 cholelithotomy,chlorophyll,chlorococcum,choleraic,chlorura
13 chlorophyll,cholelithotomy,chlorococcum,choleraic,chlorella,chivarras,chlorura
14 chlorophyll,chisel,cholelithotomy,cholecystectomy,chlorococcum,choke,choleraic,chivarras,chloramine
15 choke,chlorura
16 chlorophyll,choke,chlorura,cholecystectomy,choleraic,cholelithotomy,chlorococcum,chisel,chlorella,chishona,chokidar,chlamyphorus,chlortetracycline,chivarras
,chloramine,chlorothiazide,chloroprene,chokra
17 cholecystectomy,chlorococcum,chlorophyll,cholelithotomy,choke,choleraic,chlorura,chisel,chlorella,chishona,chlamyphorus,chirurgery,chlortetracycline,chivarras
,chokra
18 cholelithotomy,choleraic,chlorura,cholecystectomy,chlorococcum,choke,chlorophyll
19 chlorura,chlorophyll,choke,choleraic,cholelithotomy,chisel,chlorella,chlorococcum
20 chlorophyll,chlorura,cholecystectomy
21 chlorophyll,chlorura,cholecystectomy,choleraic,cholelithotomy,chlorella,choke
22 chlorophyll,chlorura,cholecystectomy,choleraic,chlorella,cholelithotomy,choke,chlorococcum,chisel
23 chlorophyll,cholelithotomy,chlorura,cholecystectomy,chlorella
24 chlorophyll,cholelithotomy,chlorella,chlorura,choleraic,chlorococcum,cholecystectomy
25 chlorophyll,cholecystectomy,cholelithotomy,chlorella,chlorura,choleraic,chisel,choke,chlorococcum
26 chlorophyll,chlorura
27 chlorophyll,chlorura,cholecystectomy,choke,choleraic,cholelithotomy,chlorella,chlorococcum,chisel,chlamyphorus,chlortetracycline,chishona,chokidar,chivarras
,chloramine,chokra,chlorothiazide,chlorambucil
28 chlorophyll,cholecystectomy,chlorura,choleraic,cholelithotomy,chlorococcum,choke,chlorella,chisel,chlamyphorus,chlortetracycline,chishona,chivarras,chlorophyceae
,chirurgery
29 chlorophyll,chlorura,cholecystectomy,choleraic,cholelithotomy,chlorella,chlorococcum
30 chlorophyll,chlorura,choke,choleraic,chlorella,cholelithotomy,cholecystectomy,chisel
31 chlorophyll,chlorura,cholecystectomy
32 chlorophyll,chlorura,cholecystectomy,chlorella,choleraic,cholelithotomy,choke
C:\Users\DELL\AppData\Local\Temp\Mxt110\RemoteFiles\69672_8_ UNIX
Plain text
720 lines
Row #1
Col #1
Type here to search
Desktop
9:10 PM
11/15/2018