CSCE 221 Cover Page
Programming Assignment #6
Due **April 29** by midnight to eCampus

First Name	Hunter	Last Name	Cleary	UIN	625001547
User Name	hncleary	E-mail address	hncleary@tamu.edu		

Please list all sources in the table below including web pages which you used to solve or implement the current homework. If you fail to cite sources you can get a lower number of points or even zero. According to the University Regulations, Section 42, scholastic dishonesty are including: acquiring answers from any unauthorized source, working with another person when not specifically permitted, observing the work of other students during any exam, providing answers when not specifically authorized to do so, informing any person of the contents of an exam prior to the exam, and failing to credit sources used. Disciplinary actions range from grade penalties to expulsion read more: Aggie Honor System Office

Type of sources			
People			
Web pages (provide URL)	URLS Listed Below		
Printed material	Data Structures and Algorithms (Textbook)	Programming P&P C++ (Stroustrup)	
Other Sources			

https://en.wikipedia.org/wiki/Associative_array
https://www.hackerearth.com/practice/data-structures/hash-tables/basics-of-hash-tables/tutorial/
http://www.cplusplus.com/reference/sstream/stringstream/stringstream/
http://www.cplusplus.com/reference/vector/vector/erase/
http://www.cplusplus.com/reference/map/map/operator[]/
http://www.cplusplus.com/reference/map/map/count/

I certify that I have listed all the sources that I used to develop the solutions/codes to the submitted work.

"On my honor as an Aggie, I have neither given nor received any unauthorized help on this academic work."

Your Name Hunter Cleary Date 4/29/18

Programming Assignment #6

Due ***April 29*** to submit to eCampus

Report

1. Explain how your brackets operator works.

   (a) What is the running time expressed in terms of big-Oh asymptotic notation of your brackets operator?

   The brackets operator runs in $O(logn)$ time.

   (b) How could you improve the running time? Here don't think about micro optimizations, think about changing data structures or major algorithm changes.

   To improve the running time, implement a different data structure. If keys were replaced with integer values, the elements could be accessed in vector in constant time $O(1)$.

2. Explain what the advantages and disadvantages of implementing map with the following data structures

   (a) A vector of key_values b.

   A vector of key_values allows for a conceptually easy design. Iterating through data is simple. Buidling the vector of keys can be done in linear time. Very useful if keys are already sorted.

   (b) A tree of key_values.

      i. Regular binary

      Allows for key search in logarithmic time. Although building the tree will take more time, computing time is conserved when looking up keys.

      ii. Red Black

      Similar to the regular binary tree but is self-balancing. The managed height of the tree would result in faster key search times. This data structure is good to use if keys need to be deleted and added frequently.

      iii. AVL

      Another self-balancing binary tree. Managed height would result in faster key search times. Most useful when the task being completed requires extensive numbers of key searches.

      iv. 2-4

      Another self-balancing tree. Allows for multiple child nodes. Like red-black trees, it is most useful when a larger number of keys need to be deleted or added.

   (c) A Hash Table of key_values

Hash tables are useful when implementation of a hash function is needed. Would be most useful when the keys being inserted are not skewed, resulting in a lower number of collisions. Accessing keys would have a constant time average, but a linear worst case. Hash tables are efficient in storing bound keys in a structure, but lack the sorting of the other structures.

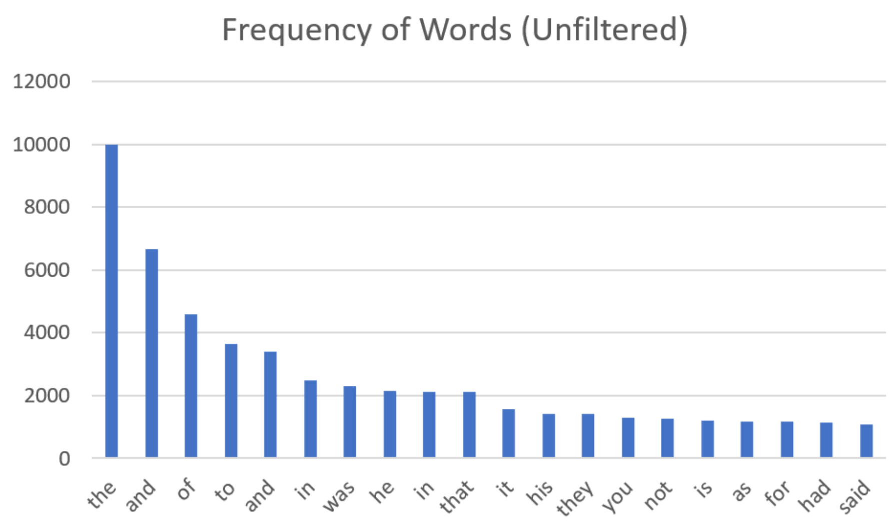
3. Explain one real life use case for a map, you should say specifically why it would benefit from using a map over a simpler data structure like a vector. This doesn't have to be something that exists currently, it's just an idea for how you could use a map. Example (don't use this one) storing word frequencies for a large document.

   A real life usage for a map data structure would be an interest profile for users on a website. The map could keep track of the frequency of related data that has been viewed in clusters. Using the frequencies, a profile could be created so that the content and advertisements shown to the user are catered to their interests.

Charts and Tables

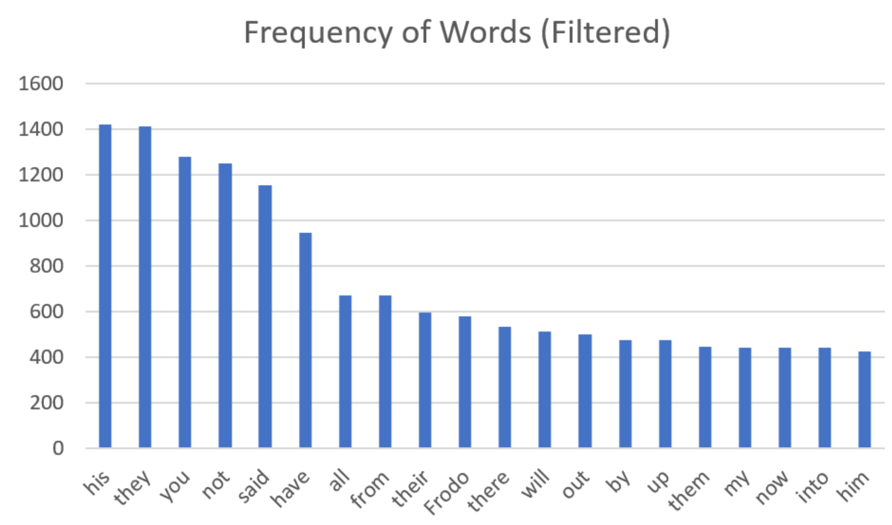
Unfiltered

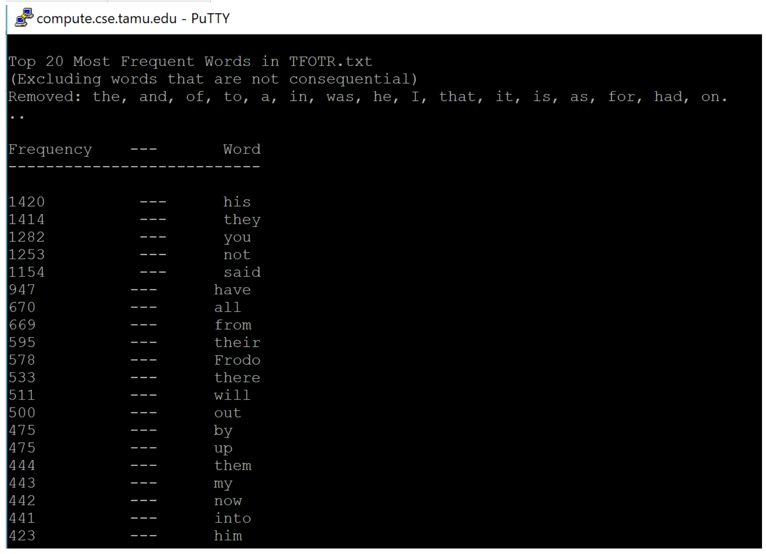
Word	Frequency
the	10009
and	6659
of	4604
to	3652
and	3405
in	2472
was	2316
he	2145
in	2134
that	2124
it	1563
his	1420
they	1414
you	1282
not	1253
is	1210
as	1180
for	1164
had	1154
said	1085



Filtered

Word	Frequency
his	1420
they	1414
you	1282
not	1253
said	1154
have	947
all	670
from	669
their	595
Frodo	578
there	533
will	511
out	500
by	475
up	475
them	444
my	443
now	442
into	441
him	423



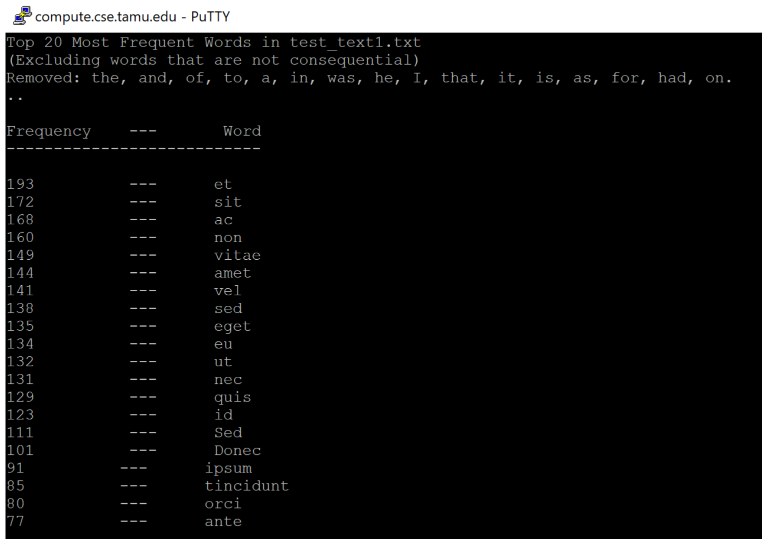


compute.cse.tamu.edu - PuTTY

```
Top 20 Most Frequent Words in TFOTR.txt
(Excluding words that are not consequential)
Removed: the, and, of, to, a, in, was, he, I, that, it, is, as, for, had, on.
..

Frequency    ---        Word
---------------------------

1420          ---      his
1414          ---      they
1282          ---      you
1253          ---      not
1154          ---      said
947          ---      have
670          ---      all
669          ---      from
595          ---      their
578          ---      Frodo
533          ---      there
511          ---      will
500          ---      out
475          ---      by
475          ---      up
444          ---      them
443          ---      my
442          ---      now
441          ---      into
423          ---      him
```



compute.cse.tamu.edu - PuTTY

```
Top 20 Most Frequent Words in test_text1.txt
(Excluding words that are not consequential)
Removed: the, and, of, to, a, in, was, he, I, that, it, is, as, for, had, on.
..

Frequency    ---        Word
---------------------------

193          ---      et
172          ---      sit
168          ---      ac
160          ---      non
149          ---      vitae
144          ---      amet
141          ---      vel
138          ---      sed
135          ---      eget
134          ---      eu
132          ---      ut
131          ---      nec
129          ---      quis
123          ---      id
111          ---      Sed
101          ---      Donec
91          ---      ipsum
85          ---      tincidunt
80          ---      orci
77          ---      ante
```

```
Frequency --- Word
10009          ---      the
6659          ---      and
4604          ---      of
3652          ---      to
3405          ---      a
2472          ---      in
2316          ---      was
2145          ---      he
2134          ---      I
2124          ---      that
1563          ---      it
1420          ---      his
1414          ---      they
1282          ---      you
1253          ---      not
1210          ---      is
1180          ---      as
1180          ---      for
1164          ---      had
1154          ---      said
1085          ---      on
1009          ---      with
955          ---      at
947          ---      have
933          ---      were
907          ---      but
867          ---      The
812          ---      be
690          ---      we
670          ---      all
669          ---      from
661          ---      He
604          ---      are
603          ---      if
601          ---      or
595          ---      their
578          ---      Frodo
533          ---      there
529          ---      But
511          ---      will
500          ---      out
490          ---      no
475          ---      by
475          ---      up
```