Script started on Thu 17 May 2018 12:40:21 PM PDT
[?1034hbash-4.2$ pwd
/afs/cats.ucsc.edu/users/s/aljilee/CMPS101S18PA3/asg3
bash-4.2$ ls -l
total 1889
-rw-r--r--. 1 aljilee users   2177 May 17 12:34 anagram.java
-rw-r--r--. 1 aljilee users   4399 May 17 12:34 FindAnagrams.java
-rw-r--r--. 1 aljilee users   874 May 17 12:34 NoteToGrader.txt
-rw-r--r--. 1 aljilee users   0 May 17 12:40 pa3submissionfile.txt
-rw-r--r--. 1 aljilee users   190 May 17 12:34 README.txt
-rw-r--r--. 1 aljilee users 1923525 May 17 12:34 wordList.txt
bash-4.2$ cat README.txt
anagram.java   Contains the anagram class
FindAnagrams.java Contains the anagrams test and the find anagram algorithm
wordList.txt   Contains the list of words to find anagrams frombash-4.2$ cat NOT [K [KoteToGrader.txt
My approach to the program was pretty simple. In order to test if a word is an anagram of another, I compared the words to see if they had the same value. I first stored each word of the word list text file in an array. Next, I assigned each letter of the alphabet a prime number. I used prime numbers because words are concatenated letters. When you multiply 2 prime numbers together, you can only recreate that number by multiplying those same 2 numbers. After assigning each letter a value, I inserted each word into the hash table one character at a time, getting a numerical value for each word and storing it in an array. When a user inputs a word, it gets the numerical value of it by inserting it into the hash table. It then loops through the array with values and if 2 numbers are equivalent, I know they are anagrams, so I print the word out from the word array. bash-4.2$ cat anagram.java
//CREATED: ALEXANDER LEE 05/14/2018

```
import java.io.*;
import java.util.*;
import java.util.Scanner.*;
import java.util.List.*;
import java.util.Hashtable.*;

class Anagram
{
   String word;

   //Char array constructor
   public Anagram(char[] word)
   {
      String str = String.valueOf(word);
      init(str);
   }

   //String constructor
   public Anagram(String word)
   {
      init(word);
   }

   //Initializes anagram type
   public void init(String word)
   {
```

```
    this.word = word;
}

//Prints anagram
public void print()
{
    for (int i = 0; i < word.length(); i++)
    {
        System.out.print(word.charAt(i));
    }
}

//Compares two words to see if they are anagrams
public boolean compare(Anagram word2, Hashtable <Character, Integer> h)
{
    char c1;
    char c2;
    int charVal1;
    int charVal2;
    int wordVal1 = 1;
    int wordVal2 = 1;

    //Don't need to go through hashtable if string length not equal
    if (word.length() != word2.word.length())
    {
        print();
        System.out.print(" & ");
        word2.print();
        System.out.print(" are not anagrams");
        return false;
    }

    //Map hashtable if word lengths equal
    for (int i = 0; i < word.length(); i++)
    {
        c1 = word.charAt(i);
        c2 = word2.word.charAt(i);
        charVal1 = h.get(c1);
        charVal2 = h.get(c2);
        wordVal1 = wordVal1 * charVal1;
        wordVal2 = wordVal2 * charVal2;
    }
    //Print if anagram
    if (wordVal1 == wordVal2)
    {
        print();
        System.out.print(" & ");
        word2.print();
        System.out.print(" are anagrams");
        return true;
    }
    //Print if not anagram
```

```
        print();
        System.out.print(" & ");
        word2.print();
        System.out.print(" are not anagrams");
        return false;
    }

    //Returns anagram word
    public String returnWord()
    {
        return word;
    }
}

bash-4.2$ javac FindAnagrams.java anagram.java
bash-4.2$ ls -l
total 1895
-rw-r--r--. 1 aljilee users    1619 May 17 12:40 Anagram.class
-rw-r--r--. 1 aljilee users    2177 May 17 12:34 anagram.java
-rw-r--r--. 1 aljilee users    3175 May 17 12:40 FindAnagrams.class
-rw-r--r--. 1 aljilee users    4399 May 17 12:34 FindAnagrams.java
-rw-r--r--. 1 aljilee users     874 May 17 12:34 NoteToGrader.txt
-rw-r--r--. 1 aljilee users       0 May 17 12:40 pa3submissionfile.txt
-rw-r--r--. 1 aljilee users     190 May 17 12:34 README.txt
-rw-r--r--. 1 aljilee users 1923525 May 17 12:34 wordList.txt
bash-4.2$ cat F indAnagrams.java
//CREATED: ALEXANDER LEE 05/14/2018

import java.io.*;
import java.util.*;
import java.util.Scanner.*;
import java.util.List.*;
import java.util.Hashtable.*;

public class FindAnagrams
{
    public static void main(String args[]) throws FileNotFoundException
    {
        //Store all contents of wordList.txt in array
        File wordList = new File("wordList.txt");
        Scanner scanner = new Scanner(wordList);
        List <String> lines = new ArrayList <String>();
        scanner.nextLine();
        while (scanner.hasNextLine())
        {
            lines.add(scanner.nextLine());
        }
        String[] wordL = lines.toArray(new String[0]);

        //Create hashtable that maps all letters of alphabet to prime number
        int isPrime = 0;
        int num = 2;
```

```
int counter = 1;
int prime[] = new int[26];
prime[0] = 2;
//Get first 26 prime numbers
for (int i = 2; i <= 26; )
{
   for (int j = 2; j <= Math.sqrt(num); j++)
   {
      if (num % j == 0)
      {
         isPrime = 0;
         break;
      }
   }
   if (isPrime != 0)
   {
      prime[counter] = num;
      counter++;
      i++;
   }
   isPrime = 1;
   num++;
}
//Create hashtable
counter = 0;
Hashtable <Character, Integer> charVal = new Hashtable <Character, Integer>();
for (char alphabet = 'a'; alphabet <= 'z'; alphabet++)
{
   charVal.put(alphabet, prime[counter]);
   counter++;
}

//Give every word a numerical value depending on its letters
double wordVal[] = new double[wordL.length];
double wordV = 1;
double charV = 0;
String word;
char c;

for (int i = 0; i < wordL.length; i++)
{
   word = wordL[i];
   for (int j = 0; j < word.length(); j++)
   {
      c = word.charAt(j);
      charV = charVal.get(c);
      wordV = wordV * charV;
   }
   wordVal[i] = wordV;
   wordV = 1;
}
```

```
//Anagram ADT testing
System.out.println("Anagram ADT Test");
char anagramWord[] = {'r', 'a', 'c', 'e', 's'};
System.out.println("String constructor test");
Anagram a1 = new Anagram("scare");     //String constructor test
Anagram a2 = new Anagram(anagramWord); //Char array test
Anagram a3 = new Anagram("scary");
a1.print();                            //Print test
System.out.println();
System.out.println("Char array constructor test");
a2.print();
System.out.println();
System.out.println("Anagrams test 1");
a1.compare(a2, charVal);               //Compare anagrams test
System.out.println();
System.out.println("Anagrams test 2");
a2.compare(a3, charVal);
System.out.println();
String randomWord = a1.returnWord(); //Return word test
System.out.println("Return word test");
System.out.println(randomWord);

//Find list of anagrams from wordList.txt
char a;
//Read user input and print out anagrams
do
{
   System.out.println("type a string of letters");
   double userVal = 1;
   Scanner in = new Scanner(System.in);
   String input = in.nextLine();
   String userIn[] = input.split(" ");

   for (int i = 0; i < userIn.length; i++)
   {
      input = userIn[i];
      for (int j = 0; j < input.length(); j++)
      {
         c = input.charAt(j);
         charV = charVal.get(c);
         userVal = userVal * charV;
      }
      for (int x = 0; x < wordVal.length; x++)
      {
         if (userVal == wordVal[x] && !input.equals(wordL[x]))
         {
            System.out.println(wordL[x]);
         }
      }
      userVal = 1;
   }
   System.out.println("Do another (y/n)?");
```

```
            a = in.next().charAt(0);
        } while (a == 'Y' || a == 'y');
    }
}
```

```
bash-4.2$ j  [Kjavac   [K  [K Find Anagrams.  [K
Anagram ADT Test
String constructor test
scare
Char array constructor test
races
Anagrams test 1
scare & races are anagrams
Anagrams test 2
races & scary are not anagrams
Return word test
scare
type a string of letters
scare
acres
cares
carse
escar
races
serac
Do another (y/n)?
n
bash-4.2$ exit
exit

Script done on Thu 17 May 2018 12:41:23 PM PDT
```