ALFECHE, NIGELL LABORA May 19, 2016

CHU IM, MICAH NICOLE J.

GABATO, CHAELLE MERCADER

CMSC 56 - A

Analysis Paper

For our project, we created 4 programs all-in all. First is the anagram checker, a program that would accept two strings and would compare each string if they both contain exactly the same characters. Second is the Caesar cipher that was broken down to two parts: 1) Encrypting 2) Deciphering. Both programs would accept a string and a key that would be the number of shifting to be made to encrypt or decrypt a sentence. The last program is the Automated Teller Machine that would able you to withdraw, transfer, inquire balance, in a quick and convenient manner.

Regarding the time complexity of the programs, all of them have a time complexity of O(n). The anagram checker compared to a brute force algorithm, is similar since the checking would have to traverse through every character in the string until a desired output has been found. Caesar Cipher is also a brute force algorithm since the code would traverse through every characters in the string until all characters have been shifted to desired key.

All of the programs have limitations of their own. The anagram checker is limited to only one pair of strings. The Caesar cipher is limited only to letters and not numbers. Most of the processes made by the ATM, are computer generated. The account and pin is randomized. And old accounts could not be retrieved.

In real life, the anagram checker could be used in playing word games. When you are limited to some letters and you can’t change the letters presented to you, the anagram checker would be helpful to you in forming words. One concrete example is the game word factory. You have a limited time and limited letters presented to you. You would have to form words within the time constrain. The anagram checker would be helpful to you if you would play this game. Caesar cipher was and is still being used in the real world. People use Caesar cipher in passing messages to other people. If you want a private conversation with another person, Caesar ciphering your message would be a good option. Your real message would be encrypted and if you look at it, it would be gibberish and hard to understand. Only when you decipher it, you would be able to read the text. And also the receiver will only be able to decipher it if she knows the shift key. So it would be good to use Caesar ciphers when passing confidential or important messages. But it won’t be efficient if you will use it as a password for it would be so easy to crack the code. The Automated Teller Machine could be used for monitoring your bank accounts. It could help you manage your savings account and would help you keep track of your transactions. Sometimes we forget our latest transactions and when the time comes that our bank accounts are empty, we are dumbfounded. With the Automated Teller Machine program, you would be able to know when and how much you have withdrawn or deposited into your account.

The programs that we made were just simple and could be improved in so many ways. The anagram checker for example could be improved by allowing multiple inputs of strings and all of the inputs will be check if it is an anagram of the other strings. The Caesar cipher would have been better if it could also decipher numbers, other symbols or pictures. If it could decipher not only letters, but also numbers, then it would be practical for passwords. The message encrypted would be harder to crack if it contains other characters aside from the English alphabet. The Automated Teller Machine could have been better if it had a database. This database could contain a person’s personal account. It could also store more accounts if it had a database.

We concluded that some programs does not necessarily have to be that complicated. Sometimes, a simple code is just enough to solve a certain problem. And if you are certain that the code is working, then that’s the time where you would think of improving your code. There are different ways or algorithms in solving a particular problem. You just have to think of the most efficient code to meet your needs. The more efficiency means the faster your program could solve a problem. The faster your program in solving a problem, the better.