Linnéuniversitetet
Kalmar Växjö

Linnaeus University

1DV700 - Computer Security
Assignment 1

Student: <Yuyao> <Duan>
Personal number: 19921104-1299
Student ID: yd222br@student.lnu.se

Setup Premises

No.	Category	Item
1	Operation System	MacOS Mojave
2	Web Browser	Google Chrome
3	IDE	Visual Studio Code (Python 3.8.3)
3	Web Tool	HexEd.it (https://hexed.it/) Transforming Hex to Binary
4	Web Tool	Convertbinary (https://www.convertbinary.com/to-text/) Binary Code to Text Translator
5	Web Tool	Decode(https://www.dcode.fr/monoalphabetic-substitution) Simple Substitution Cipher Decoder
6	Web Tool	Substitution Cipher Breaker (https://planetcalc.com/8047/) Simple Substitution Cipher Decoder
7	Web Tool	Dummy Text Generator (http://www.dummytextgenera-tor.com/#jump) Generating Random Text
8	Web Tool	Generate Prime Numbers (https://www.browser-ling.com/tools/prime-numbers) Generating Large Prime Numbers
9	Python Library	Matplotlib

Task 1

a) **Symmetric encryption** and **asymmetric encryption** are considered as different ways to encode (translating entire words) or encipher (translating letters or symbols individually) a message to make it not obvious [1]. The principle of a cryptosystem includes two elements: the encryption rules – **algorithms** and the **key**. The process to achieve encryption can be expressed as *C = E (K, P)*. In this expression, *C* indicates the result after encryption, *E* represents a set of encryption algorithms, and *K* is the selected algorithm from the set. In reality, to achieve encrypted information transfer includes three main stages:

The plaintext [(encryption)→] The ciphertext [(decryption)→] The plaintext

During this process, the symmetric encryption method indicates that using the same key for encryption and decryption; while in the asymmetric encryption system, two separated keys (an encryption key and a decryption key) are implemented respectively during these two stages [1].

During information transformation, **encryption algorithms** demonstrate a set of rules to encode plaintexts as a ciphertext [1]. The main purpose of encryption algorithms is to hide the meaning of original materials. By contrast, **hash algorithms** are applied in order to detect data integrity when unintentional transmission errors occurred or to prevent malicious modifications in the files or message from potential attackers. **Hash (**or **checksum** or **message digest)** play as the main role which can be considered as a seal to the file. The principle of hash algorithms is by implementing one-way functions which will prevent attackers from working backward to see the inputs leading to the digest value. Therefore, changes in the file can be detected and it is impossible for attackers to modify the hash value of the file [1].

Data **compression** refers to implement a technique (algorithm) to transform data format which requires less space for the data storage or less bandwidth for transferring data [2], [3]. This technique is mainly used for space saving. This technique is normally combined with data encryption algorithm to achieve storage space saving and data encryption [2], [3]. By comparison, **hashing** is a process to convert a given key into another value. The convert function which is called hashing function is a mathematical algorithm to generate the new value according to the key [4]. In data encryption field, hashing can be applied for transforming plaintext passwords as cryptographic hashes. So even a database is attacked, the critical information can be kept as encrypted data [4].

b) **Steganography** is defined as a technique which is used for hiding a secret data within an ordinary, non-secret file or message. The secret information, therefore, can be transferred without detection and be extracted in the destination [5]. Modern digital steganography technique normally includes two steps: firstly, the secret data will be encrypted or obfuscated (data encryption); secondly, the encrypted data will be inserted to a normal file based on some special algorithms [5]. **Digital watermarking** can be considered as a typical implementation of steganography technique [5]. However, instead of delivering secret message to the receiver, watermarking technique is normally used for identifying digital publications' belongings. By inserting a trademark or some other special data, the published digital files will be secretly tagged with certain designed digital labels [5]. The application of watermarking will contribute to copyright protection on the internet [5]. The idea of both steganography and digital watermarking can be considered as subsets of cryptography or **encryption** [1]. Generally speaking, encryption is the process of encoding a message, therefore the real content of the message can be protected from any third party [1]. Normally, encryption indicates to encode the whole file, message, information or another type of data. In contrast, steganography refers to hide a secret information within a normal file without being detected. When it is transferred, the normal file can also be encrypted again.

Encryption is widely used in everyday digital communicate, while steganography is used for more secretly delivering information, probably, under the normal encryption's cover. The real message still needs to further extraction even after the normal decryption. The purpose of using watermarking, however, is different from both encryption and steganography, instead of delivering message, this technique is used for identifying certain digital publications if infringement of permissions or copyrights happened [5].

Task 2

a) From the web page we can see that the author designed a type of steganography that will allow the user to hide a picture in another cover picture. The principle of this steganography is as follows: each channel of each pixel in an image is represented by an 8-bit value, while some bits are most significant for representing the image and some bits are least significant for representing the picture. In order to achieve steganography, the program can extract the most significant bits of each pixel of the image which the user want to hide and insert them to replace the **least significant bits** of each pixel in the "cover picture". The result image which combines both bits of pixels of the message picture and the cover picture is the implementation of steganography.

The limitation of this method is: the hidden information can be identified by performing edge detection on bit plane which is not difficult to achieve. By implementing edge detection for a normal image, from the most significant bits to the least significant, we can find that the least significant bits actually look like very random-looking [6]. In comparison, if a picture includes some hidden information, less random-looking may occur when edge detection reaching to least significant bits [6].

b) In addition to the **least significant bit** technique, there are a few other techniques are used in steganography. **Palette based technique** uses images as malware carriers [7]. Attackers can hide an encrypted message and then hide it in a stretched palette of the cover image [7]. Due to the malware is normally encrypted, it may take a lot of time to decrypt for threat identifying. Furthermore, another technique is **secure cover selection** which is one of the most difficult method to be detected [7]. Attackers will prepare the images as the malware carriers which the blocks of the images have same blocks of the malware [7]. The malware will then be carefully fitted into the images [7]. Comparing the least significant bit technique, the carrier image from secure cover selection technique is identical to the original picture which can easily cheat the detection software [7].

c) This task is solved by identifying **least significant bit** in "Secret.bmp" file. The steps for solving this task are as follows:

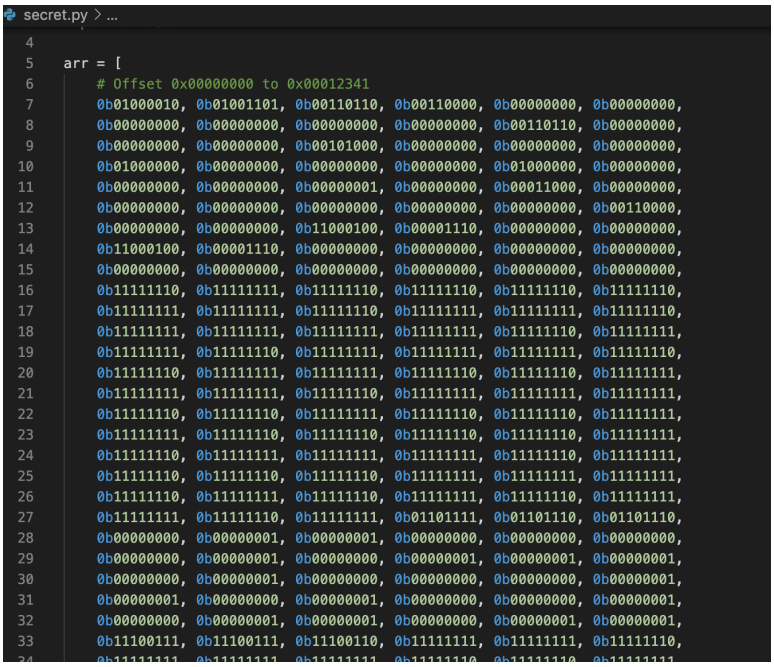

Figure 1. Binary codes of Secret.bmp

First of all, using online tool "hexed.it" (https://hexed.it/) to open this file and extracting the data to Visual Studio Code as binary format (see Figure 1). Then we can see that the least significant bits start to appear since line 16 where the least significant bits from the first byte of line 16 until the next

8 least significant bits combining as “01000011”. By implementing the online tool “convertbinary.com” (https://www.convertbinary.com/to-text/), which shows that “01000011” equals ASCII text “C”, therefore the secret seems like to appear. After keeping extracting least significant bits from the data, it eventually shows the final result of the secret message (see Figure 2).

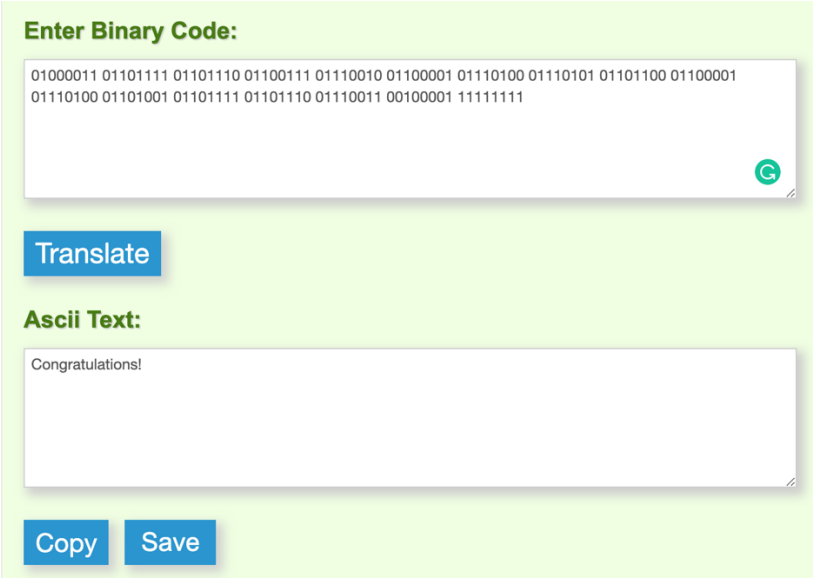

Figure 2. Decoding the secret message

After checking the rest binary codes of “Secret.bmp” file, no least significant bits were identified from the file after “11111111”. We then can conclude that the secret text from “Secret.bmp” file is “Congratulations!”

Task 3

a) As we can see the secret message is “RK ERKT EHURMXD”. By implementing the simple substitution key from this task, we can see that “RK ERKT EHURMXD” equals “in vino veritas” (A further investigation on “in vino veritas” which probably indicates a famous restaurant/bar).

b) It is possible decrypted this message by someone who does not have the key. The reason for that is because even the number of keys for substitution cipher is around $2^{88.4}$ but there are actually a lot of potential clues to decrypt the English words [8]. There are many statistical properties of English text which can make it quite easy to identify the keys [8]. This means the frequency distribution of the letters in an English text can be a good starting point to decrypt the cipher.

Furthermore, the structures of English words do have regular patterns which will help the decipher to find the key, such as most frequent digraphs, trigraphs, doubles etc. [8]. Other patterns include one-letter words in English (a, I), frequent two-letter words (of, to, in, it, is, be, as, at, so, we, he, by, or, on, do, if, me, my, up, an, go, no, us, am), frequent three-letter words (the, and, for, are, but, not, you, all, any, can, had, her, was, one, our, out, day, get, has, him, his, how, man, new, now, old, see, two, way, who, boy, did, its, let, put, say, she, too, use), and frequent four-letter words will contribute to decrypt simple substitution cipher (that, with, have, this, will, your, from, they, know, want, been, good, much, some, time) [8].

In addition, the text should follow English grammar to make it understandable, therefore the decipher can use this principle to effectively infer the encrypted text.

c) The starting point to decrypt the message “QMJ BPZ B XPJZ RZWJPAXQ LAD” can be the single “B”. Since we discussed in b), in an English text, there are only two English words with single alphabet which are “a” and “I”. Since “BPZ” starts with B, we can infer here either a word starting with “a??” or “i??”. Based on English language habits, we firstly test “BPZ” as “are” in the text. In this logic, the text can make sense when “QMJ” is subject as “you”. “You are…” can be consider as the most reasonable guess since there is no other appropriate subjects for this grammar. Based on these speculations, “XPJZ” can be decrypted as “?rue” and “RZWJPAXQ” can be decrypted as “?e?ur??y”. It is easy to infer “XPJZ” as “true” since the combination which also leads to a clearer clue for “RZWJPAXQ” which stands for “?e?ur?ty”. By now, it is not difficult to decode “?e?ur?ty” as “security” and therefore “LAD” can be decrypted as “?i?”.
Based on previous decryption, we could find the following relation (see Table 1):

Table 1: The relation between alphabets and key alphabets

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
i	a		?						u		?	o			r	y	s					c	t		e

In order to decode the last word, we need to confirm L and D. But comparing the relation and decrypted context, it could be difficult to confirm the relation for other characters. Another strategy to decrypt can be guessing the word based on meaning of the text. Together with the meaning of the text and related possible combinations for “?i?”, “bin” is inferred as the possible word for the text. The message “QMJ BPZ B XPJZ RZWJPAXQ LAD” is decoded as “**You are a true security bin**”, which “bin” makes this sentence meaningful.

The conclusion about this task is that it can be difficult to decode simple substitution cipher sometimes when the content is very short. This kind of text is easier to decode when the content is very large.

Task 4

Task 4 is regarding how to implement at least two simple encryption methods (substitution and transposition method) and then to implement decryption method. The grogram “Task4.py” includes seven functions.

The first function is “receive_key ()”. The idea of this function is to accept a number between 1 and 25 as the key for substitution method to use. The user can decide which number as the key for substation encryption.

After receiving the key from user, the principle of the second function “encryption_substi ()” is based on “Caesar Cipher” [8]. The reason to design the key between 1 and 25 for this substitution encryption is due to the encryption alphabet list based on each letter of the original alphabet list with a same shifting. The key received from “receive_key ()” is used for this shift. The reason for this key between 1 and 25 is due to the valid shifts for “Caesar Cipher” in the interval of “[1, 25]” where “0” and “26” do not make sense for the encryption shifting. Furthermore, since there are only 26 alphabets in English, therefore, designing the key interval with “[1, 25]” will be valid for this encryption. The changing shift is implemented by the following process: first of all, the program will identify the character is uppercase- or lowercase-alphabet or neither of them (non-alphabet characters will not be processed); after this, if the character is uppercase English letter, the program will identify its ASCII code and adding the key value and subtracting the “A”’s ASCII code (65); furthermore, the real shifting number will be found after modulo 26 which by adding 65 will be the encrypted letter. The same principle applies for lowercase English letters where only changing 65 to 97 (ASCII code of “a”).

The transposition encryption will encrypt the text based on substituted encryption. The key for this function is used for to exchange order of the subtext. The length of the key will decide how to slice plaintext. In this case, the key is “(1,4,3,2)”, each subtext will be a length of four. The last subtext in the sliced list will be removed before changing position since the length of the last one may not equal to four. Then using the key to change alphabets’ positions. The original’s first one will be encrypted as the first one and the original’s fourth alphabet will be changed to the second position etc.

The decryption method includes two steps: the first step is to decrypt transposition encryption and then for the substitution encryption. For decoding the transposition encryption, the function just applies “encryption_trans ()” again with the same key, then the plaintext will be generated. After that, the user needs to input the key which is selected by the user during the substitution encryption. By implementing the key, the system will shift back the encrypted alphabet list based on the key number, therefore the program can retrieve the original letters.

The program starts with a “while true loop” in order to continuously asking user which method to perform. Selection 1 and 2 are designed for Task 4, and selection 3 and 4 are designed for Task 5.
For Task 4, after selecting 1, the user needs to set up a key based on the system prompt. The result will be a processed file in the same path as the program named “message_encryption.txt”. By selecting 2, the encrypted cipher from “message_encryption.txt” will be decrypted to “message_plaintext.txt”. The user can write the content with any character.

Task 5

Task 5 shares the same program as Task 4. In Task 5, the user should copy the downloaded file "plaintext.txt" to the same path as the program. After that, according to the requirements of Task 5, a few paragraphs and a name were added in this .txt file for later use. By selecting 3, the content from "plaintext.txt" will be encrypted by both substitution and transposition methods (product cipher) of the program which will be outputted as "encryption.text" in the same path as the program. The user can decode the encrypted content by the selection 4 in the program. The plaintext will be then saved in a "generate_plaintext.txt" file which has the same content as the original "plaintext.txt".

The encrypted file is renamed as "Yuyao_Duan.txt" which the key for substitution encryption is "14" (see Figure 3.).

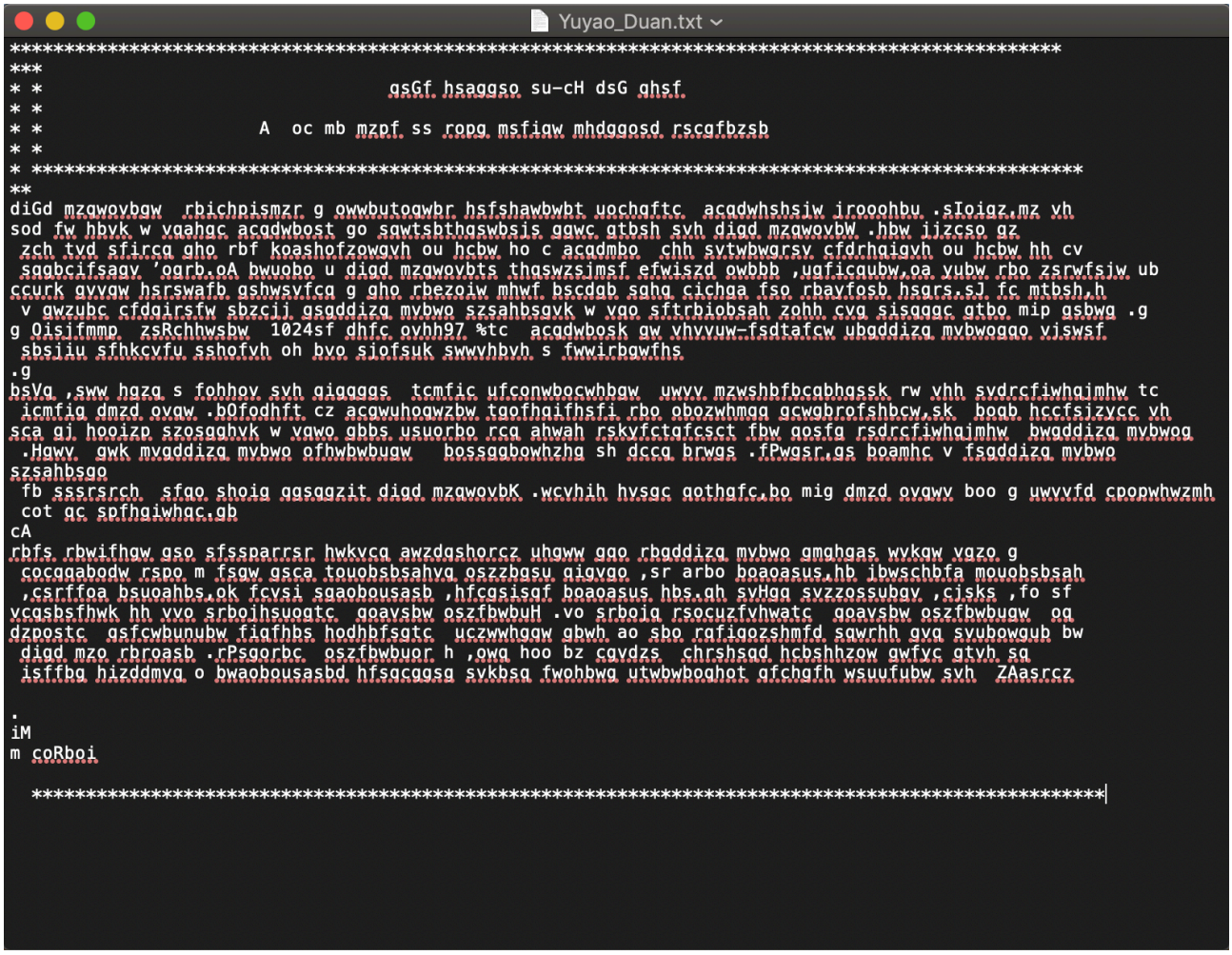

Figure 3. Encrypted file "Yuyao_Duan.txt" based on product cipher

Task 6

a) Decryption “Chen Ningrui.txt”

In this task, I will perform crypto analysis to a cipher file called “Chen Ningrui.txt”. This analysis is based on knowing about the structure of the original .txt file. As we know, the header of this file includes the following content: “Secret message - Top Secret May only be read by security passed personnel”. After analyzing “Chen Ningrui.txt”, we can find that this file was implemented with substitution method. We can see that each of the encrypted word has same number of letters as the original one. Therefore, to research the relation between the original alphabets and the key alphabets can be very important for this task.

Based on understanding the original header, the decoding analysis will be presented in this paragraph: “Amkzmb” can be inferred as “Secret”; “umaaiom” can be inferred as “message”; “Bwx” can be inferred as “Top”; “Uig” can be inferred as “May”; “wvtg” can be inferred as “only”;
“jm” can be inferred as “be”; “zmil” can be inferred as “read”; “amkczqbg” can be inferred as “security”; “xiaaml” can be inferred as “passed”; “xmzawvvmt” can be inferred as “personnel”. Furthermore, we know that in the end will present the author’s name, then “Kpmv Vqvozcq” can be inferred as “Chen Ningrui”. We could find the relation between the encrypted alphabets and plaintext alphabets as follows (see Table 2):

Table 2: The relation between alphabets and key alphabets

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
i	j	k	l	m		o	p	q			t	u	v	w	x		z	a	b	c				g	

After finding this relation, we could identify the potential pattern among them. We can find that even the second line is reordered as a key alphabet list (see Table 2), between the key alphabets we can still find the orders between identified letters and blanks. Further decryption about the blanks from above as follows (see Table 3):

Table 3: The deduction from Table 2

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
i	j	k	l	m	**n**	o	p	q	**r**	**s**	t	u	v	w	x	**y**	z	a	b	c	**d**	**e**	**f**	g	**h**

Therefore, we get the key (“ijklmnopqrstuvwxyzabcdefgh”) for the cipher (see Table 3). By implement the online tool “decode” (https://www.dcode.fr/monoalphabetic-substitution) with this key, we could decode the secret text from “Chen Ningrui.txt” as follows:
(Main body of this cipher)

MEXICAN DRUG LORD JOAQUIN “EL CHAPO”; GUZMAN HAS COMPLAINED IN COURT ABOUT HIS CONDITIONS OF CUSTODY IN A US JAIL. GUZMAN IS BEING HELD IN A MAXIMUM SECURITY PRISON IN NEW YORK AFTER HE WAS EXTRADITED LAST MONTH.THE NOTORIOUS KINGPIN ESCAPED TWICE FROM PRISON IN MEXICO, ONCE IN A LAUNDRY BASKET AND MOST RECENTLY THROUGH A TUNNEL IN HIS CELL. HIS LAWYERS SAY HE HAS BEEN DENIED MARITAL VISITS AND IS LARGELY KEPT IN SOLITARY CONFINEMENT. THE CLAIMS AROSE AT A FEDERAL COURT IN BROOKLYN AT A HEARING FOR GUZMAN, WHO HAS PLEADED NOT GUILTY TO CHARGES THAT HE RAN THE WORLD'S LARGEST DRUG-TRAFFICKING ORGANISATION DURING A DECADES-LONG CAREER. HE FACES LIFE IN PRISON IF CONVICTED. GUZMAN'S WIFE EMMA CORONEL, A 27-YEAR-OLD FORMER BEAUTY QUEEN AND MOTHER OF HIS TWINS, FLEW FROM MEXICO TO ATTEND THE HEARING. HIS LAWYERS SAID IT WAS FIRST TIME MS CORONEL HAD SEEN HER HUSBAND SINCE HIS SURPRISE EXTRADITION TWO WEEKS AGO.GUZMAN, 59, IS REPORTEDLY ON 23-HOUR LOCKDOWN IN A SPECIAL UNIT OF THE MANHATTAN CORRECTIONAL CENTER. “WE UNDERSTAND

THE NEED FOR SECURITY BUT WE THINK IT HAS GONE ABOVE AND BEYOND," ; SAID MICHELLE GELERNT, ONE OF HIS COURT-APPOINTED LAWYERS. DISTRICT JUDGE BRIAN COGAN POINTED OUT THAT THE "HISTORY OF THE DEFENDANT IS SOMEWHAT UNUSUAL" ; - A REFERENCE TO HIS PAST ESCAPES - AND SAID THE FEDERAL BUREAU OF PRISONS SHOULD DECIDE WHAT CONDITIONS GUZMAN FACED AND WHO HE COULD SEE.WHO IS 'EL CHAPO' GUZMAN? EL CHAPO: FIVE THINGS HIS WIFE REVEALED DRUG LORD EL CHAPO EXTRADITED TO USGUZMAN'S SINALOA CARTEL ALLEGEDLY SMUGGLED HUNDREDS OF TONNES OF COCAINE, HEROIN, MARIJUANA AND METHAMPHETAMINES TO THE US WHILE WAGING WAR WITH OTHER GANGS. THE CARTEL IS ACCUSED OF CARRYING OUT THOUSANDS OF MURDERS AND KIDNAPPINGS, AND BRIBING OFFICIALS. GUZMAN - WIDELY KNOWN BY HIS NICKNAME EL CHAPO, WHICH MEANS "SHORTY" ; - IS BELIEVED TO HAVE AMASSED A BILLION-DOLLAR FORTUNE THROUGH THE DRUGS TRADE. MEXICAN PRESIDENT ENRIQUE PENA NIETO HAD INITIALLY RESISTED EXTRADITING HIM TO THE US, INSISTING THAT HE SHOULD FACE JUSTICE AT HOME. BUT AFTER GUZMAN WAS RECAPTURED IN JANUARY 2016, PENA NIETO CHANGED HIS MIND ON EXTRADITION AND ORDERED OFFICIALS TO SPEED UP THE PROCESS.

According to the result we can see that the key alphabets analysis is correct and the plaintext does make sense. The conclusion of this task is that substitution cipher can be decoded based on some known plaintext material. The key can be inferred through finding sufficient relations between the plaintext and the encrypted text. Once the key alphabet list is decoded, the rest of the encryption cipher can be decrypted.

b) Decryption "Yasser Almodhi ciphertext.txt"

After analyzing the content of this text, I identified this file was encrypted by simple substitution encryption which is based on the principle of "Caesar Cipher". I decrypted this file based on my own program for Task 4. I modified the selection 4 of the program by removing the decryption codes for transposition encryption (see Task_6_b.py). After this, I tested the possible keys from "1" to "25" and identified "17" is the key for this simple substitution encryption. The interesting part of this encryption file is that the author mentioned the key for this file is "17". One thing to mention here is that there are double "THE" before "ASSIGNMENT" where can be confirmed from the encryption file with double "KYZ". This proves that there was no error during decryption.

SECRET MESSAGE - TOP SECRET

MAY ONLY BE READ BY SECURITY PASSED PERSONNEL

WE DID THE THE ASSIGNMENT IN ONLY 2 DAYS. IT WAS HARD ESPECIALLY THE THIRD QUESTION.

THE SECRET KEY IS SEVENTEEN.

YASSER ALMODHI,MOHAMMED ALMNAEA.

c) Decryption "Substitution_rasmus_skedinger_Seldin_Music.txt"

After analyzing the content of this text, I identified this file was encrypted by substitution encryption. Due to the author did not encrypted the header and name of this file, therefore it cannot be decrypted

by analyzing header and name. Furthermore, this file cannot be decrypted by my own program, therefore, the author did not use "Caesar Cipher" for substitution. This encrypted file is decrypted by online tool "Substitution cipher breaker" (https://planetcalc.com/8047/). The decryption condition for using this tool is that the content of the encrypted text should be sufficient i.e. this tool cannot decrypt the content like Task 3. The algorithms behind this online tool demand large among of data to correct the decryption result where this file has large content by chance. The decrypted content by this online tool as follows and the key alphabet list for this encryption is "BZYXGWVUETASJFLKRHCDIMQPON":
(Main body of this cipher)

I AM HAPPY TO JOIN WITH YOU TODAY IN WHAT WILL GO DOWN IN HISTORY AS THE GREATEST DEMONSTRATION FOR FREEDOM IN THE HISTORY OF OUR NATION FIVE SCORE YEARS AGO A GREAT AMERICAN IN WHOSE SYMBOLIC SHADOW WE STAND TODAY SIGNED THE EMANCIPATION PROCLAMATION THIS MOMENTOUS DECREE CAME AS A GREAT BEACON LIGHT OF HOPE TO MILLIONS OF NEGRO SLAVES WHO HAD BEEN SEARED IN THE FLAMES OF WITHERING INJUSTICE IT CAME AS A JOYOUS DAYBREAK TO END THE LONG NIGHT OF THEIR CAPTIVITY BUT ONE HUNDRED YEARS LATER THE NEGRO STILL IS NOT FREE ONE HUNDRED YEARS LATER THE LIFE OF THE NEGRO IS STILL SADLY CRIPPLED BY THE MANACLES OF SEGREGATION AND THE CHAINS OF DISCRIMINATION ONE HUNDRED YEARS LATER THE NEGRO LIVES ON A LONELY ISLAND OF POVERTY IN THE MIDST OF A VAST OCEAN OF MATERIAL PROSPERITY ONE HUNDRED YEARS LATER THE NEGRO IS STILL LANGUISHED IN THE CORNERS OF AMERICAN SOCIETY AND FINDS HIMSELF AN EXILE IN HIS OWN LAND AND SO WE VE COME HERE TODAY TO DRAMATIZE A SHAMEFUL CONDITION IN A SENSE WE VE COME TO OUR NATION S CAPITAL TO CASH A CHECK WHEN THE ARCHITECTS OF OUR REPUBLIC WROTE THE MAGNIFICENT WORDS OF THE CONSTITUTION AND THE DECLARATION OF INDEPENDENCE THEY WERE SIGNING A PROMISSORY NOTE TO WHICH EVERY AMERICAN WAS TO FALL HEIR THIS NOTE WAS A PROMISE THAT ALL MEN YES BLACK MEN AS WELL AS WHITE MEN WOULD BE GUARANTEED THE UNALIENABLE RIGHTS OF LIFE LIBERTY AND THE PURSUIT OF HAPPINESS IT IS OBVIOUS TODAY THAT AMERICA HAS DEFAULTED ON THIS PROMISSORY NOTE INSOFAR AS HER CITIZENS OF COLOR ARE CONCERNED INSTEAD OF HONORING THIS SACRED OBLIGATION AMERICA HAS GIVEN THE NEGRO PEOPLE A BAD CHECK A CHECK WHICH HAS COME BACK MARKED INSUFFICIENT FUNDS BUT WE REFUSE TO BELIEVE THAT THE BANK OF JUSTICE IS BANKRUPT WE REFUSE TO BELIEVE THAT THERE ARE INSUFFICIENT FUNDS IN THE GREAT VAULTS OF OPPORTUNITY OF THIS NATION AND SO WE VE COME TO CASH THIS CHECK A CHECK THAT WILL GIVE US UPON DEMAND THE RICHES OF FREEDOM AND THE SECURITY OF JUSTICE WE HAVE ALSO COME TO THIS HALLOWED SPOT TO REMIND AMERICA OF THE FIERCE URGENCY OF NOW THIS IS NO TIME TO ENGAGE IN THE LUXURY OF COOLING OFF OR TO TAKE THE TRANQUILIZING DRUG OF GRADUALISM NOW IS THE TIME TO MAKE REAL THE PROMISES OF DEMOCRACY NOW IS THE TIME TO RISE FROM THE DARK AND DESOLATE VALLEY OF SEGREGATION TO THE SUNLIT PATH OF RACIAL JUSTICE NOW IS THE TIME TO LIFT OUR NATION FROM THE QUICKSANDS OF RACIAL INJUSTICE TO THE SOLID ROCK OF BROTHERHOOD NOW IS THE TIME TO MAKE JUSTICE A REALITY FOR ALL OF GOD S CHILDREN IT WOULD BE FATAL FOR THE NATION TO OVERLOOK THE URGENCY OF THE MOMENT THIS SWELTERING SUMMER OF THE NEGRO S LEGITIMATE DISCONTENT WILL NOT PASS UNTIL THERE IS AN INVIGORATING AUTUMN OF FREEDOM AND EQUALITY NINETEEN SIXTY THREE IS NOT AN END BUT A BEGINNING AND THOSE WHO HOPE THAT THE NEGRO NEEDED TO BLOW OFF STEAM AND WILL NOW BE CONTENT WILL HAVE A RUDE

AWAKENING IF THE NATION RETURNS TO BUSINESS AS USUAL AND THERE WILL BE NEITHER REST NOR TRANQUILITY IN AMERICA UNTIL THE NEGRO IS GRANTED HIS CITIZENSHIP RIGHTS THE WHIRLWINDS OF REVOLT WILL CONTINUE TO SHAKE THE FOUNDATIONS OF OUR NATION UNTIL THE BRIGHT DAY OF JUSTICE EMERGES BUT THERE IS SOMETHING THAT I MUST SAY TO MY PEOPLE WHO STAND ON THE WARM THRESHOLD WHICH LEADS INTO THE PALACE OF JUSTICE IN THE PROCESS OF GAINING OUR RIGHTFUL PLACE WE MUST NOT BE GUILTY OF WRONGFUL DEEDS LET US NOT SEEK TO SATISFY OUR THIRST FOR FREEDOM BY DRINKING FROM THE CUP OF BITTERNESS AND HATRED WE MUST FOREVER CONDUCT OUR STRUGGLE ON THE HIGH PLANE OF DIGNITY AND DISCIPLINE WE MUST NOT ALLOW OUR CREATIVE PROTEST TO DEGENERATE INTO PHYSICAL VIOLENCE AGAIN AND AGAIN WE MUST RISE TO THE MAJESTIC HEIGHTS OF MEETING PHYSICAL FORCE WITH SOUL FORCE THE MARVELOUS NEW MILITANCY WHICH HAS ENGULFED THE NEGRO COMMUNITY MUST NOT LEAD US TO A DISTRUST OF ALL WHITE PEOPLE FOR MANY OF OUR WHITE BROTHERS AS EVIDENCED BY THEIR PRESENCE HERE TODAY HAVE COME TO REALIZE THAT THEIR DESTINY IS TIED UP WITH OUR DESTINY AND THEY HAVE COME TO REALIZE THAT THEIR FREEDOM IS INEXTRICABLY BOUND TO OUR FREEDOM WE CANNOT WALK ALONE AND AS WE WALK WE MUST MAKE THE PLEDGE THAT WE SHALL ALWAYS MARCH AHEAD WE CANNOT TURN BACK THERE ARE THOSE WHO ARE ASKING THE DEVOTEES OF CIVIL RIGHTS WHEN WILL YOU BE SATISFIED WE CAN NEVER BE SATISFIED AS LONG AS THE NEGRO IS THE VICTIM OF THE UNSPEAKABLE HORRORS OF POLICE BRUTALITY WE CAN NEVER BE SATISFIED AS LONG AS OUR BODIES HEAVY WITH THE FATIGUE OF TRAVEL CANNOT GAIN LODGING IN THE MOTELS OF THE HIGHWAYS AND THE HOTELS OF THE CITIES WE CANNOT BE SATISFIED AS LONG AS THE NEGRO S BASIC MOBILITY IS FROM A SMALLER GHETTO TO A LARGER ONE WE CAN NEVER BE SATISFIED AS LONG AS OUR CHILDREN ARE STRIPPED OF THEIR SELF HOOD AND ROBBED OF THEIR DIGNITY BY SIGNS STATING FOR WHITES ONLY WE CANNOT BE SATISFIED AS LONG AS A NEGRO IN MISSISSIPPI CANNOT VOTE AND A NEGRO IN NEW YORK BELIEVES HE HAS NOTHING FOR WHICH TO VOTE NO NO WE ARE NOT SATISFIED AND WE WILL NOT BE SATISFIED UNTIL JUSTICE ROLLS DOWN LIKE WATERS AND RIGHTEOUSNESS LIKE A MIGHTY STREAM I AM NOT UNMINDFUL THAT SOME OF YOU HAVE COME HERE OUT OF GREAT TRIALS AND TRIBULATIONS SOME OF YOU HAVE COME FRESH FROM NARROW JAIL CELLS AND SOME OF YOU HAVE COME FROM AREAS WHERE YOUR QUEST QUEST FOR FREEDOM LEFT YOU BATTERED BY THE STORMS OF PERSECUTION AND STAGGERED BY THE WINDS OF POLICE BRUTALITY YOU HAVE BEEN THE VETERANS OF CREATIVE SUFFERING CONTINUE TO WORK WITH THE FAITH THAT UNEARNED SUFFERING IS REDEMPTIVE GO BACK TO MISSISSIPPI GO BACK TO ALABAMA GO BACK TO SOUTH CAROLINA GO BACK TO GEORGIA GO BACK TO LOUISIANA GO BACK TO THE SLUMS AND GHETTOS OF OUR NORTHERN CITIES KNOWING THAT SOMEHOW THIS SITUATION CAN AND WILL BE CHANGED LET US NOT WALLOW IN THE VALLEY OF DESPAIR I SAY TO YOU TODAY MY FRIENDS AND SO EVEN THOUGH WE FACE THE DIFFICULTIES OF TODAY AND TOMORROW I STILL HAVE A DREAM IT IS A DREAM DEEPLY ROOTED IN THE AMERICAN DREAM I HAVE A DREAM THAT ONE DAY THIS NATION WILL RISE UP AND LIVE OUT THE TRUE MEANING OF ITS CREED WE HOLD THESE TRUTHS TO BE SELF EVIDENT THAT ALL MEN ARE CREATED EQUAL I HAVE A DREAM THAT ONE DAY ON THE RED HILLS OF GEORGIA THE SONS OF FORMER SLAVES AND THE SONS OF FORMER SLAVE OWNERS WILL BE ABLE TO SIT DOWN TOGETHER AT THE TABLE OF BROTHERHOOD I HAVE A DREAM THAT ONE DAY EVEN THE STATE OF MISSISSIPPI A STATE SWELTERING WITH THE HEAT OF INJUSTICE SWELTERING WITH THE HEAT OF OPPRESSION

WILL BE TRANSFORMED INTO AN OASIS OF FREEDOM AND JUSTICE I HAVE A DREAM THAT MY FOUR LITTLE CHILDREN WILL ONE DAY LIVE IN A NATION WHERE THEY WILL NOT BE JUDGED BY THE COLOR OF THEIR SKIN BUT BY THE CONTENT OF THEIR CHARACTER I HAVE A DREAM TODAY I HAVE A DREAM THAT ONE DAY DOWN IN ALABAMA WITH ITS VICIOUS RACISTS WITH ITS GOVERNOR HAVING HIS LIPS DRIPPING WITH THE WORDS OF INTERPOSITION AND NULLIFICATION ONE DAY RIGHT THERE IN ALABAMA LITTLE BLACK BOYS AND BLACK GIRLS WILL BE ABLE TO JOIN HANDS WITH LITTLE WHITE BOYS AND WHITE GIRLS AS SISTERS AND BROTHERS I HAVE A DREAM TODAY I HAVE A DREAM THAT ONE DAY EVERY VALLEY SHALL BE EXALTED AND EVERY HILL AND MOUNTAIN SHALL BE MADE LOW THE ROUGH PLACES WILL BE MADE PLAIN AND THE CROOKED PLACES WILL BE MADE STRAIGHT AND THE GLORY OF THE LORD SHALL BE REVEALED AND ALL FLESH SHALL SEE IT TOGETHER THIS IS OUR HOPE AND THIS IS THE FAITH THAT I GO BACK TO THE SOUTH WITH WITH THIS FAITH WE WILL BE ABLE TO HEW OUT OF THE MOUNTAIN OF DESPAIR A STONE OF HOPE WITH THIS FAITH WE WILL BE ABLE TO TRANSFORM THE JANGLING DISCORDS OF OUR NATION INTO A BEAUTIFUL SYMPHONY OF BROTHERHOOD WITH THIS FAITH WE WILL BE ABLE TO WORK TOGETHER TO PRAY TOGETHER TO STRUGGLE TOGETHER TO GO TO JAIL TOGETHER TO STAND UP FOR FREEDOM TOGETHER KNOWING THAT WE WILL BE FREE ONE DAY AND THIS WILL BE THE DAY THIS WILL BE THE DAY WHEN ALL OF GOD S CHILDREN WILL BE ABLE TO SING WITH NEW MEANING MY COUNTRY TIS OF THEE SWEET LAND OF LIBERTY OF THEE I SING LAND WHERE MY FATHERS DIED LAND OF THE PILGRIM S PRIDE FROM EVERY MOUNTAINSIDE LET FREEDOM RING AND IF AMERICA IS TO BE A GREAT NATION THIS MUST BECOME TRUE AND SO LET FREEDOM RING FROM THE PRODIGIOUS HILLTOPS OF NEW HAMPSHIRE LET FREEDOM RING FROM THE MIGHTY MOUNTAINS OF NEW YORK LET FREEDOM RING FROM THE HEIGHTENING ALLEGHENIES OF PENNSYLVANIA LET FREEDOM RING FROM THE SNOW CAPPED ROCKIES OF COLORADO LET FREEDOM RING FROM THE CURVACEOUS SLOPES OF CALIFORNIA BUT NOT ONLY THAT LET FREEDOM RING FROM STONE MOUNTAIN OF GEORGIA LET FREEDOM RING FROM LOOKOUT MOUNTAIN OF TENNESSEE LET FREEDOM RING FROM EVERY HILL AND MOLEHILL OF MISSISSIPPI FROM EVERY MOUNTAINSIDE LET FREEDOM RING AND WHEN THIS HAPPENS AND WHEN WE ALLOW FREEDOM RING WHEN WE LET IT RING FROM EVERY VILLAGE AND EVERY HAMLET FROM EVERY STATE AND EVERY CITY WE WILL BE ABLE TO SPEED UP THAT DAY WHEN ALL OF GOD S CHILDREN BLACK MEN AND WHITE MEN JEWS AND GENTILES PROTESTANTS AND CATHOLICS WILL BE ABLE TO JOIN HANDS AND SING IN THE WORDS OF THE OLD NEGRO SPIRITUAL FREE AT LAST FREE AT LAST THANK GOD ALMIGHTY WE ARE FREE AT LAST

d) Decryption “Tommy_Duff.txt”

After analyzing the content, this file is encrypted by substitution principle. Due to it has sufficient length for “Substitution cipher breaker”, the plaintext can be decoded by this tool again. The key alphabet list for this encryption is: “VUTSRXPONMLKJIHGFEDCBAZYQW”.

SECRET MESSAGE - TOP SECRET

MAY ONLY BE READ BY SECURITY PASSED PERSONNEL

I HAD VISIONS, I WAS IN THEM

I WAS LOOKING INTO THE MIRROR

TO SEE A LITTLE BIT CLEARER

ROTTENNESS AND EVIL IN ME
FINGERTIPS HAVE MEMORIES
MINE CAN'T FORGET THE CURVES OF YOUR BODY
AND WHEN I FEEL A BIT NAUGHTY
I RUN IT UP THE FLAGPOLE AND SEE WHO SALUTES
(BUT NO ONE EVER DOES)
I'M NOT SICK BUT I'M NOT WELL
AND I'M SO HOT CAUSE I'M IN HELL
BEEN AROUND THE WORLD AND FOUND
THAT ONLY STUPID PEOPLE ARE BREEDING
THE CRETINS CLONING AND FEEDING
AND I DON'T EVEN OWN A TV
PUT ME IN THE HOSPITAL FOR NERVES
AND THEN THEY HAD TO COMMIT ME
YOU TOLD THEM ALL I WAS CRAZY
THEY CUT OFF MY LEGS NOW I'M AN AMPUTEE, GOD DAMN YOU
I'M NOT SICK BUT I'M NOT WELL
AND I'M SO HOT CAUSE I'M IN HELL
I'M NOT SICK BUT I'M NOT WELL
AND IT'S A SIN TO LIVE SO WELL
I WANT TO PUBLISH ZINES
AND RAGE AGAINST MACHINES
I WANT TO PIERCE MY TONGUE
IT DOESN'T HURT, IT FEELS FINE
THE TRIVIAL SUBLIME
I'D LIKE TO TURN OFF TIME
AND KILL MY MIND
YOU KILL MY MIND
PARANOIA, PARANOIA
EVERYBODY'S COMING TO GET ME
JUST SAY YOU NEVER MET ME
I'M GOING UNDERGROUND WITH THE MOLES DIGGING HOLES
HEAR THE VOICES IN MY HEAD
I SWEAR TO GOD IT SOUNDS LIKE THEY'RE SNORING
BUT IF YOU'RE BORED THEN YOU'RE BORING
THE AGONY AND THE IRONY, THEY'RE KILLING ME

I'M NOT SICK BUT I'M NOT WELL
AND I'M SO HOT CAUSE I'M IN HELL
I'M NOT SICK BUT I'M NOT WELL
AND IT'S A SIN TO LIVE SO WELL
TOMMY DUFF

e) Decryption “Substitution_Ahmadreza Vakilalroayayi.txt”
After analyzing the content, this file is encrypted by substitution principle. Due to it has sufficient length for “Substitution cipher breaker”, the plaintext can be decoded by this tool again. The key alphabet list for this encryption is: “BDGHEIJKLAMNOFPQRSTUVCWZYX”.

SECRET MESSAGE - TOP SECRET
MAY ONLY BE READ BY SECURITY PASSED PERSONNEL
THIS IS A SUMMARY OF SINUHE'S STORY "THE PHYSICIAN" MY NAME IS SINOHE WHO WROTE THIS BOOK, I AM NOT WRITING THIS BOOK TO WORSHIP GODS AND PHARAOH BECAUSE I AM TIRED OF PHARAOH AND WHOLE GODS, THERE IS NO ENTHUSIASM FOR ME TO BECOME FAMOUS AND RICH BY WRITING THIS BOOK. I AM JUST WRITING FOR MY OWN , FIRST OF ALL I HAVE TO SAY THAT I AM NOT PRAYING TO ANY PHARAOHS AND GODS CAUSE I DONT BELIEVE IN THEIR HUMANITY SINCE THERE IS A PHARAOH THERE WILL BE NO EQUALITY BETWEEN SLAVES AND KINGS LIKE PHARAOHS , WHAT I BELIEVE IS ABOUT EQUALITY BETWEEN US AND PHARAOHS , I SAW A LOT IN MY LIFETIME I SAW A SON WAS KILLING HIS OWN FATHER IN FRONT OF MY EYES , I SAW UPRISING OF POOR PEOPLE AGAINST RICHE AND GODS , IN PAST I WAS THE CLOSEST PERSON TO PHARAOH AND SLAVEHOLDERS WAS FLATTERING ME BY SENDING ME A LOTS OF VALUABLE THINGS. THE MAN WHO I CALLED FATHER WAS A PHYSICIAN OF POOR PEOPLE IN THEBES WHICH IS THE MOST BEAUTIFUL AND BIGGEST CITY IN THE WORLD, AND MY MOTHER BELIEVE THAT I AM A GIFT FROM GODS BECAUSE THEY DONT HAVE CHILDREN TILL THEY WAS SENIOR ONE DAY SHE TOOK ME FROM NIL RIVER, AND THEY DID NOT KNOW THIS GIFT WILL GOING TO MAKE A HUGE MISERY IN THE FUTURE FOR THEM.

f) Decryption “Sai_Shashank_Maktala_Ciphertext.txt”
After analyzing the content, this file is encrypted by substitution principle. Due to it has sufficient length for “Substitution cipher breaker”, the plaintext can be decoded by this tool again. The key alphabet list for this encryption is: “ZYXWVUTSRQPONMLKJIHGFEDCBA”.
(Main body of this cipher)

IT’S LIKE A BILLBOARD FOR DISILLUSIONMENT AND MISTRUST, AND IT’S EVERYWHERE: “EPSTEIN DIDN’T KILL HIMSELF.” THE PHRASE HAS BEEN SLAPPED ON BEER CANS, PRINTED ON SWEATSHIRTS, BLURTED AS A NONSEQUITUR IN CABLE NEWS INTERVIEWS, SCRAWLED ON POSTERS HELD UP AT SOUTHERN COLLEGE FOOTBALL GAMES AND ON SAN FRANCISCO BAR BATHROOM WALLS CRAMPED WITH SHARPIED PHONE NUMBERS AND PROFANITY. IT'S POPPED UP ON CLASSICAL MUSIC SUBREDDITS, WHERE IT WAS DETERMINED THAT THE PHRASE IS IN 7/8 TIME. IT’S BEEN NOT-SO-SECRETLY SPELLED OUT IN THE IMPEACHMENT-RELATED TWEETS OF REPRESENTATIVE PAUL GOSAR. (YOU MAY REMEMBER HIM AS THE GUY WHO COULDN’T GET AN ENDORSEMENT FROM HIS OWN SIBLINGS WHILE RUNNING FOR OFFICE. THIS WEEK, HE’S THE GUY WHOSE LAST 23 POSTS ARE A PARANOID ACROSTIC.) IT’S BECOME A JOKE ON DATING APPS, AND IT’S BEEN

AN INCESSANT TALKING POINT FOR SOCIAL MEDIA USERS LEFT, RIGHT, AND SLANTWAYS.

SOURCE: HTTPS://WWW.WIRED.COM/STORY/EPSTEIN-DIDNT-KILL-HIMSELF-CONSPIRACY/

g) Decryption "RaphaePageDylanEbert.txt"
After analyzing the content, this file is encrypted by substitution principle. Due to it has sufficient length for "Substitution cipher breaker", the plaintext can be decoded by this tool again. The key alphabet list for this encryption is: "IQKLMNOPVRSTUFWXYZABCDEJGH".

SECRET MESSAGE - TOP SECRET
MAY ONLY BE READ BY SECURITY PASSED PERSONNEL
ND I WILL STRIKE DOWN UPON THEE WITH GREAT FENGEANCE AND JURIOUS ANGER THOSE WHO WOULD ATTEMPT TO POISON AND DESTROY MY BROTHERS. AND YOU WILL KNOW MY NAME IS THE LORD WHEN I LAY MY FENGEANCE. UPON THEE.

EBERT DYLAN - PAGE RAPHAEL

h) Decryption "Pouya Khast.txt"
After analyzing the content, this file is encrypted by substitution principle. Due to it has sufficient length for "Substitution cipher breaker", the plaintext can be decoded by this tool again. The key alphabet list for this encryption is: "JYZABCDEFGHIXKLMNOPQRSTUVW".

SECRET MESSAGE - TOP SECRET
MAY ONLY BE READ BY SECURITY PASSED PERSONNEL
WHAT EMERGES IS THE PICTURE OF A RATIONALIST. FOR INSTANCE, DA VINCI WAS ONE OF THE FIRST TO QUESTION AND CONCLUDED THAT THESE COULD NOT HAVE ~EEN DEPOSITED IN A FORTY DAY FLOOD. HE LOOKED AT RIVER VALLEYS AND DID THE MATH; THEY COULD ONLY HAVE ~EEN ERODED OVER HUGE HORIZONS OF TIME. POUYA KHAST

i) Decryption "plaintext_Jean-Pierre_Salum.txt"
After analyzing the content, this file is encrypted by substitution principle. Due to it has sufficient length for "Substitution cipher breaker", the plaintext can be decoded by this tool again. The key alphabet list for this encryption is: "JMPSVYBEHKNQTWZCFILORUXADG".
(Main body of this cipher)

M HAD A PROGRAMMABLE DRUM MACHINE WITH PEGS (CAMS) THAT BUMP INTO LITTLE LEVERS THAT OPERATE THE PERCUSSION. THE DRUMMER COULD BE MADE TO PLAY DIFFERENT RHYTHMS AND DIFFERENT DRUM PATTERNS BY MOVING THE PEGS TO DIFFERENT LOCATIONS. ANOTHER SOPHISTICATED PROGRAMMABLE MACHINE BY AL-JAZARI WAS THE CASTLE CLOCK.

THE JACQUARD LOOM, DEVELOPED IN 1801, IS OFTEN QUOTED AS A SOURCE OF PRIOR ART. THE MACHINE USED A SERIES OF PASTEBOARD CARDS WITH HOLES PUNCHED IN THEM. THE HOLE PATTERN REPRESENTED THE PATTERN THAT THE LOOM HAD TO FOLLOW IN WEAVING CLOTH. THE LOOM COULD PRODUCE ENTIRELY DIFFERENT WEAVES USING DIFFERENT SETS OF CARDS. THE USE OF PUNCHED CARDS WAS ALSO ADOPTED BY CHARLES BABBAGE AROUND 1830, TO CONTROL HIS ANALYTICAL ENGINE.

THIS INNOVATION WAS LATER REFINED BY HERMAN HOLLERITH WHO, IN 1896 FOUNDED THE TABULATING MACHINE COMPANY (WHICH BECAME IBM). HE INVENTED THE HOLLERITH PUNCHED CARD, THE CARD READER, AND THE KEY PUNCH MACHINE. THESE INVENTIONS WERE THE FOUNDATION OF THE MODERN INFORMATION PROCESSING INDUSTRY. THE ADDITION OF A PLUG-BOARD TO HIS 1906 TYPE I TABULATOR ALLOWED IT TO DO DIFFERENT JOBS WITHOUT HAVING TO BE REBUILT (THE FIRST STEP TOWARD PROGRAMMING). BY THE LATE 1940S THERE WERE A VARIETY OF PLUG-BOARD PROGRAMMABLE MACHINES, CALLED UNIT RECORD EQUIPMENT, TO PERFORM DATA PROCESSING TASKS (CARD READING). THE EARLY COMPUTERS WERE ALSO PROGRAMMED USING PLUG-BOARDS.
A BOX OF PUNCH CARDS WITH SEVERAL PROGRAM DECKS.

THE INVENTION OF THE VON NEUMANN ARCHITECTURE ALLOWED COMPUTER PROGRAMS TO BE STORED IN COMPUTER MEMORY. EARLY PROGRAMS HAD TO BE PAINSTAKINGLY CRAFTED USING THE INSTRUCTIONS OF THE PARTICULAR MACHINE, OFTEN IN BINARY NOTATION. EVERY MODEL OF COMPUTER WOULD BE LIKELY TO NEED DIFFERENT INSTRUCTIONS TO DO THE SAME TASK. LATER ASSEMBLY LANGUAGES WERE DEVELOPED THAT LET THE PROGRAMMER SPECIFY EACH INSTRUCTION IN A TEXT FORMAT, ENTERING ABBREVIATIONS FOR EACH OPERATION CODE INSTEAD OF A NUMBER AND SPECIFYING ADDRESSES IN SYMBOLIC FORM (E.G. ADD X, TOTAL). IN 1954 FORTRAN, THE FIRST HIGHER LEVEL PROGRAMMING LANGUAGE, WAS INVENTED. THIS ALLOWED PROGRAMMERS TO SPECIFY CALCULATIONS BY ENTERING A FORMULA DIRECTLY (E.G. Y = X*2 + 5*X + 9). THE PROGRAM TEXT, OR SOURCE, WAS CONVERTED INTO MACHINE INSTRUCTIONS USING A SPECIAL PROGRAM CALLED A COMPILER. MANY OTHER LANGUAGES WERE DEVELOPED, INCLUDING ONES FOR COMMERCIAL PROGRAMMING, SUCH AS COBOL. PROGRAMS WERE MOSTLY STILL ENTERED USING PUNCH CARDS OR PAPER TAPE. (SEE COMPUTER PROGRAMMING IN THE PUNCH CARD ERA). BY THE LATE 1960S, DATA STORAGE DEVICES AND COMPUTER TERMINALS BECAME INEXPENSIVE ENOUGH SO PROGRAMS COULD BE CREATED BY TYPING DIRECTLY INTO THE COMPUTERS. TEXT EDITORS WERE DEVELOPED THAT ALLOWED CHANGES AND CORRECTIONS TO BE MADE MUCH MORE EASILY THAN WITH PUNCH CARDS.

AS TIME HAS PROGRESSED, COMPUTERS HAVE MADE GIANT LEAPS IN THE AREA OF PROCESSING POWER. THIS HAS BROUGHT ABOUT NEWER PROGRAMMING LANGUAGES THAT ARE MORE ABSTRACTED FROM THE UNDERLYING HARDWARE. ALTHOUGH THESE MORE ABSTRACTED LANGUAGES REQUIRE ADDITIONAL OVERHEAD, IN MOST CASES THE HUGE INCREASE IN SPEED OF MODERN COMPUTERS HAS BROUGHT ABOUT LITTLE PERFORMANCE DECREASE COMPARED TO EARLIER COUNTERPARTS. THE BENEFITS OF THESE MORE ABSTRACTED LANGUAGES IS THAT THEY ALLOW BOTH AN EASIER LEARNING CURVE FOR PEOPLE LESS FAMILIAR WITH THE OLDER LOWER-LEVEL PROGRAMMING LANGUAGES, AND THEY ALSO ALLOW A MORE EXPERIENCED PROGRAMMER TO DEVELOP SIMPLE APPLICATIONS QUICKLY. DESPITE THESE BENEFITS, LARGE COMPLICATED PROGRAMS, AND PROGRAMS THAT ARE MORE DEPENDENT ON SPEED STILL REQUIRE THE FASTER AND RELATIVELY LOWER-LEVEL LANGUAGES WITH TODAY'S HARDWARE. (THE SAME CONCERNS WERE RAISED ABOUT THE ORIGINAL FORTRAN LANGUAGE.)

THROUGHOUT THE SECOND HALF OF THE TWENTIETH CENTURY, PROGRAMMING WAS AN ATTRACTIVE CAREER IN MOST DEVELOPED COUNTRIES. SOME FORMS OF

PROGRAMMING HAVE BEEN INCREASINGLY SUBJECT TO OFFSHORE OUTSOURCING (IMPORTING SOFTWARE AND SERVICES FROM OTHER COUNTRIES, USUALLY AT A LOWER WAGE), MAKING PROGRAMMING CAREER DECISIONS IN DEVELOPED COUNTRIES MORE COMPLICATED, WHILE INCREASING ECONOMIC OPPORTUNITIES IN LESS DEVELOPED AREAS. IT IS UNCLEAR HOW FAR THIS TREND WILL CONTINUE AND HOW DEEPLY IT WILL IMPACT PROGRAMMER WAGES AND OPPORTUNITIES.

HAOFEI YAN HY222AP

j) Decryption “PetkoGerdzhikovSub.txt”

After analyzing the content, this file is encrypted by substitution principle. Due to it has sufficient length for “Substitution cipher breaker”, the plaintext can be decoded by this tool again. The key alphabet list for this encryption is: “FGHIBQKLMNDOPJEARSTCUVWXYZ”.

SECRET MESSAGE - TOP SECRET
MAY ONLY BE READ BY SECURITY PASSED PERSONNEL

IN A COLUMNAR TRANSPOSITION, THE MESSAGE IS WRITTEN OUT IN ROWS OF A FIXED LENGTH, AND THEN READ OUT AGAIN COLUMN BY COLUMN, AND THE COLUMNS ARE CHOSEN IN SOME SCRAMBLED ORDER. BOTH THE WIDTH OF THE ROWS AND THE PERMUTATION OF THE COLUMNS ARE USUALLY DEFINED BY A KEYWORD. FOR EXAMPLE, THE WORD ZEBRAS IS OF LENGTH 6 (SO THE ROWS ARE OF LENGTH 6), AND THE PERMUTATION IS DEFINED BY THE ALPHABETICAL ORDER OF THE LETTERS IN THE KEYWORD. IN THIS CASE, THE ORDER WOULD BE "6 3 2 4 1 5". IN A REGULAR COLUMNAR TRANSPOSITION CIPHER, ANY SPARE SPACES ARE FILLED WITH NULLS; IN AN IRREGULAR COLUMNAR TRANSPOSITION CIPHER, THE SPACES ARE LEFT BLANK. FINALLY, THE MESSAGE IS READ OFF IN COLUMNS, IN THE ORDER SPECIFIED BY THE KEYWORD. THIS IS VERY SENSITIVE INFORMATION PLEASE USE IT WITH CAUTION!

PETKO GERDZHIKOV

Task 7

a) The simple hash function is designed to generate hash values when receiving inputs. In order to achieve uniformity in this function, a unique value needs to be produced before mod 256. One common method is to apply prime numbers to calculate hash [9]. In this function (see Figure 4.), the

```
def hash_function(string,p):
    hash = 0
    k = 0
    for i in string:
        k += 1
        hash += ((p**(len(string)-k)) * ord(i))
    hash = hash % 256
    return hash
```

Figure 4. An implementation of simple hash function

prime number is used to create a large unique value. The large unique figure will be achieved through adding a unique weight to each of the character's ASCII code. The weight is calculated by a prime number and a dynamic changing value between different characters in a string. Each time certain character's weight will be consisted of the prime number with a power of "k" which is a changing number in each round. Through this method, a unique large number will be generated. According to the requirement of the task, the hash value should be 8-bits. Therefore, the large value should modulo 256 to yield the final hash value.

b) **Property one – Uniformity:**
Uniformity is defined as similar input will generate the result with same possibility [10]. This can be considered as the output of the hash function which is distributed evenly when the function receiving similar input [10]. The test for the first property based on the hints from instruction of "Task 7 b". Several test files with different amount of random words (10, 50, 100, 500, 1000, 10000) are generated by the only tool "Dummy Text Generator" (http://www.dummytextgenerator.com/#jump).

The prime number which is demanded by this hash function is generated by the online tool "Generate Prime Numbers" (https://www.browserling.com/tools/prime-numbers). A large prime number – "701" (comparing the size of buckets, in this case is 256) is yielded and selected for this hash function. All the hash values of the input text will be saved in a new list "lst_hash" for further analysis. To present the test, the occurrences of each word's hash value of the text need to be added up. This is achieved by the following function (see Figure 5.):

```
b = collections.Counter(lst_hash)
dic = {number: value for number, value in b.items()}
x = [i for i in dic.keys()]
y = []
for i in dic.keys():
    y.append(dic.get(i))
```

Figure 5. Summing up hash value occurrences

All the hash values of the input text will be saved in a new list "lst_hash" for further analysis. All the codes of property one test are saved in the file "Task_7_perporty_1.py". The presentations of test results based on matplotlib are as follows:

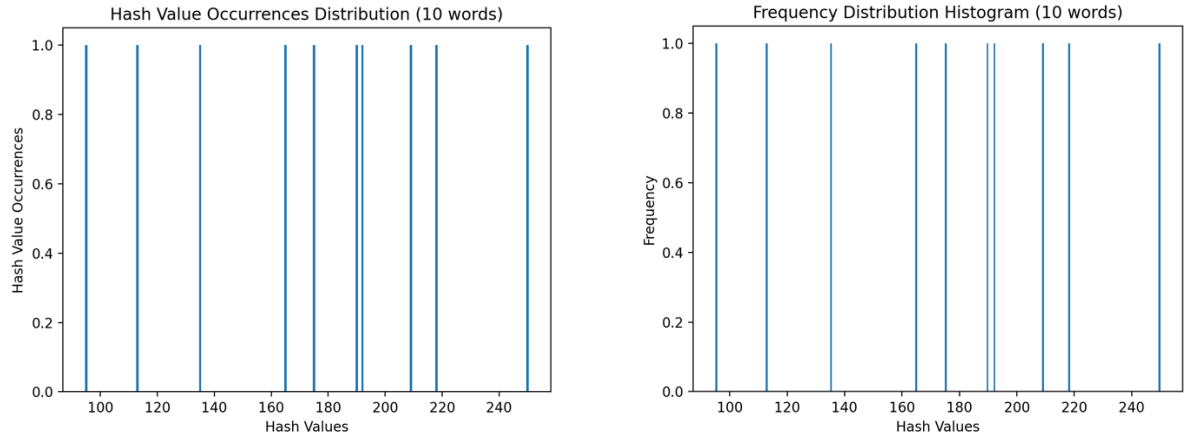

Figure 6. Hash value occurrences distribution and frequency distribution histogram (10 words)

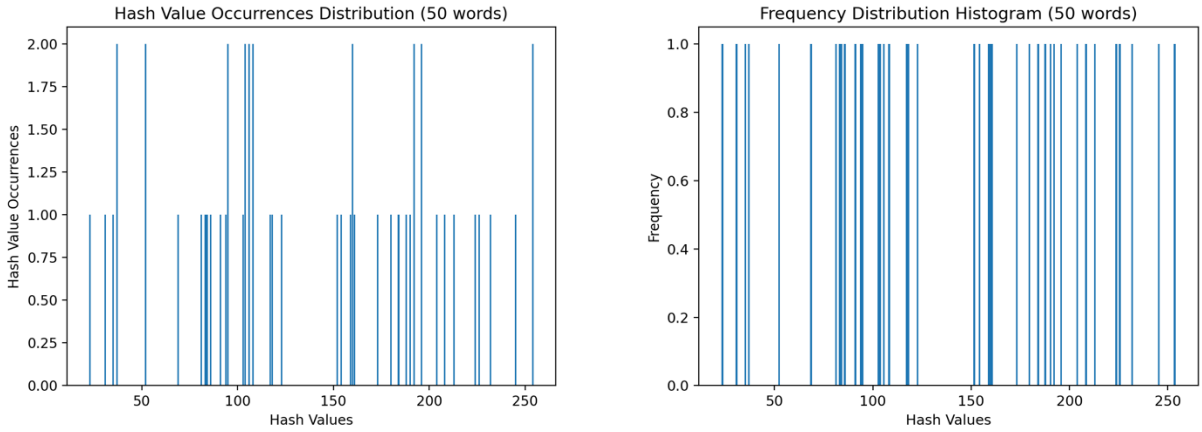

Figure 7. Hash value occurrences distribution and frequency distribution histogram (50 words)

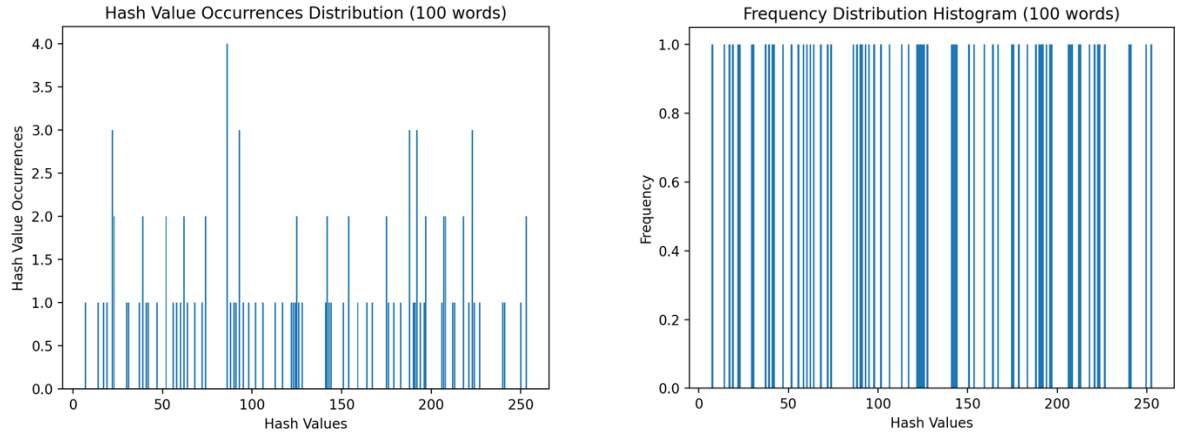

Figure 8. Hash value occurrences distribution and frequency distribution histogram (100 words)

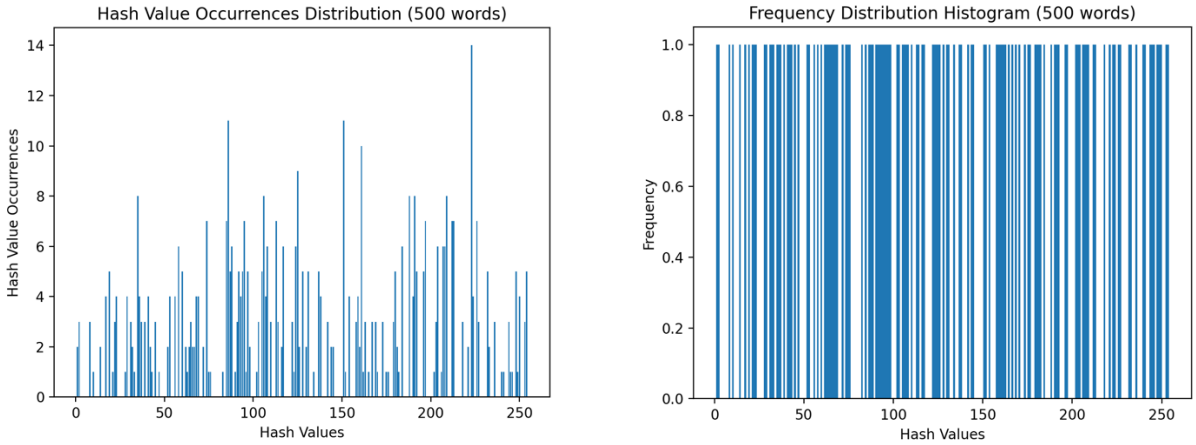

Figure 9. Hash value occurrences distribution and frequency distribution histogram (500 words)

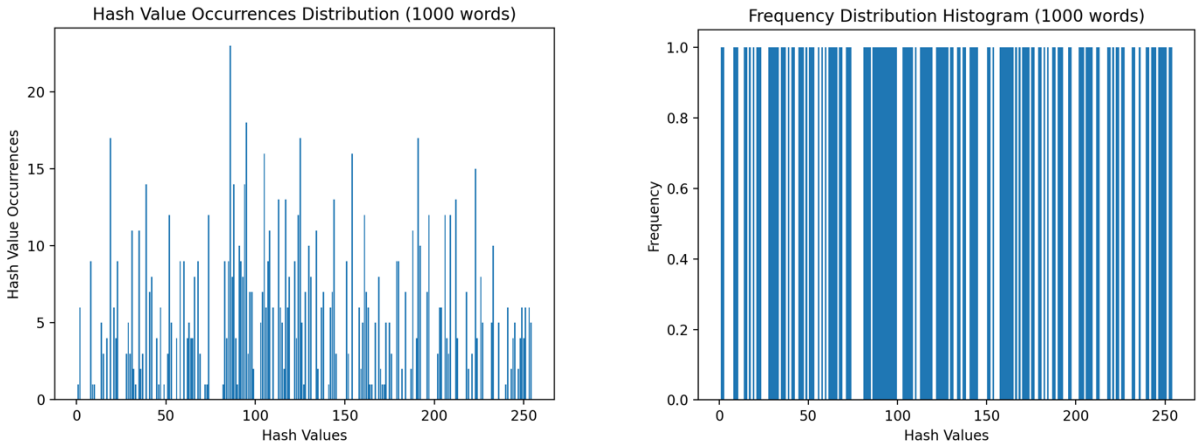

Figure 10. Hash value occurrences distribution and frequency distribution histogram (1000 words)

From Figure 6 to 10, we can see that this hash function's performance with different amount of words. The left side "Hash Value Occurrences Distribution" diagrams give an overview of typical hash values with how many occurrences. All hash values are well distributed when a total amount of 10 words are processed by this function. This trend is followed by "Hash Value Occurrences Distribution (50 words)", while only a few bucket collisions happened around hash value 50, 100, 150, and 200. When total inputs increase up to 100 words, the uniformity of this hash function can be found among most of the data since most of the occurrences remaining at "1.0". From the "500 words" test, we can see that the data is distributed evenly overall where only a few hash values have more occurrences than the others, typically hash values around 100, 150, and 225. A better uniformity of this function is found when test words reaching to 1000 words. On the right side, the "Frequency Distribution Histogram" can assist to analyze this trend. The sparse degree of the blue bars describes how good the uniformity of this function is. From 100 to 1000 words, we can see that more words input the better sparseness the function achieved which can be also proved by the following standard deviation statistics table. This table is realized by the following function (see Figure 11.):

```
# standard deviation function
def mean(lst):
    sum = 0
    for i in lst:
        sum += int(i)
    return round( float(sum / len(lst) ), 4 )
def std(lst):
    avg_lst = mean(lst)
    sum = 0
    for i in lst:
        sum += ( ( int(i) - avg_lst ) ** 2 )
    std = sqrt ( sum / len(lst) )
    return round (std, 4)
```

Figure 11. The standard deviation function for the statistic table

From 500 words (2.416) to 1000 words (4.2898), the standard deviation of the tests shows an overall good uniformity of this function. As we can see from "Frequency Distribution Histogram (1000 words)", there are only a few white bars in this diagram where most of parts show a good uniformity.

Table 4: The standard deviation statistics of the occurrences among different hash value buckets

Input strings	Standard deviation
10 words	0.0
50 words	0.433
100 words	0.6725
500 words	2.416
1000 words	4.2898

Property two - Testing and measurement:
In order to test the second property, the similar inputs are demanded which can be achieved by adding "AAA", "AAB", "AAC" … "AAZ" to each of the word in the test text. The process is realized by the following function (see Figure 12.). All the modified words will be saved in a new list for the test.

```
# save the words in lst
lst = read_words("1000_words_uniformity.txt")
lst2 = []
for i in lst:
    ending = ["AAA","AAB","AAC","AAD","AAE","AAF","AAG","AAH","AAI","AAJ",
              "AAK","AAL","AAM","AAN","AAO","AAP","AAQ","AAR","AAS","AAU",
              "AAV","AAW","AAX","AAY","AAZ"]
    lst2.append(i + random.choice(ending))
```

Figure 12. Generating similar strings

Based on the new strings, the test result can be found as the following diagrams:

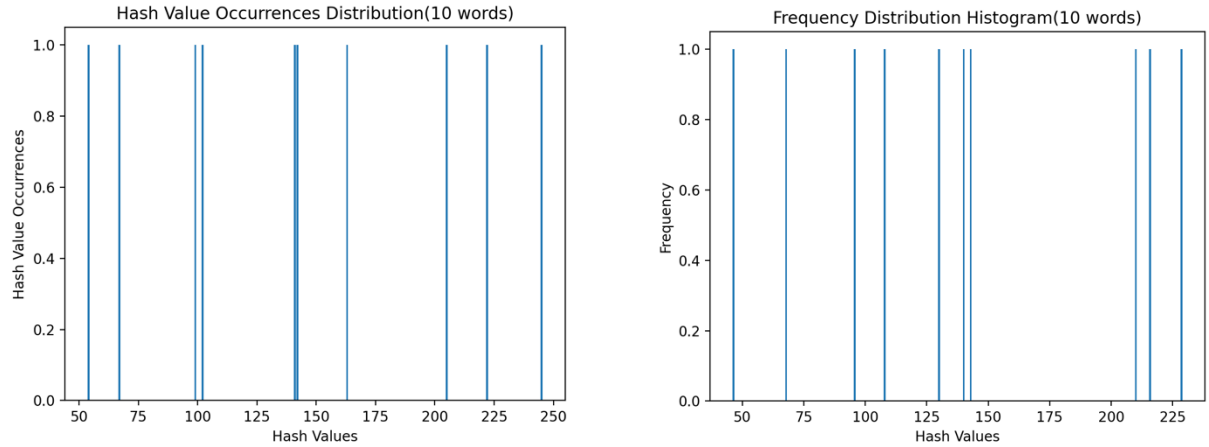

Figure 13. Hash value occurrences distribution and frequency distribution histogram (10 words)

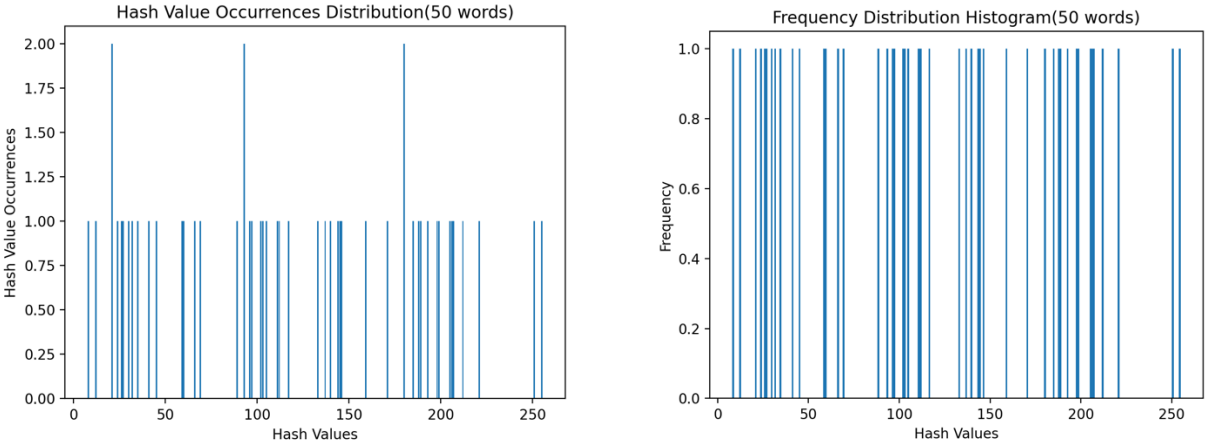

Figure 14. Hash value occurrences distribution and frequency distribution histogram (50 words)

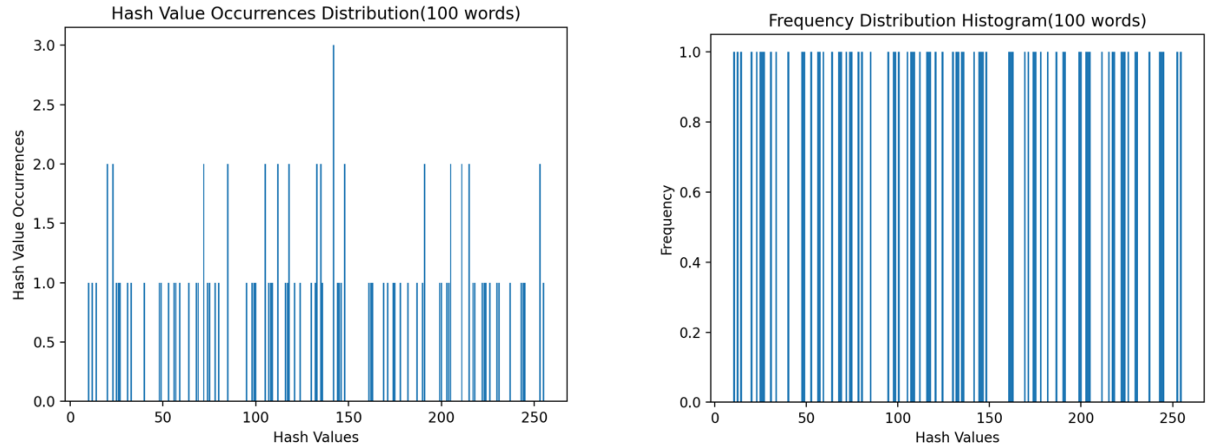

Figure 15. Hash value occurrences distribution and frequency distribution histogram (100 words)

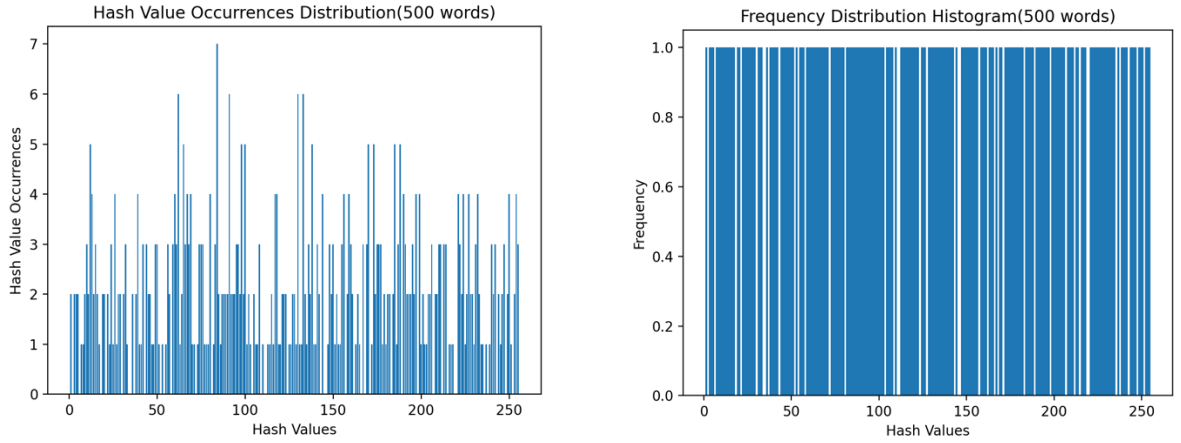

Figure 16. Hash value occurrences distribution and frequency distribution histogram (500 words)

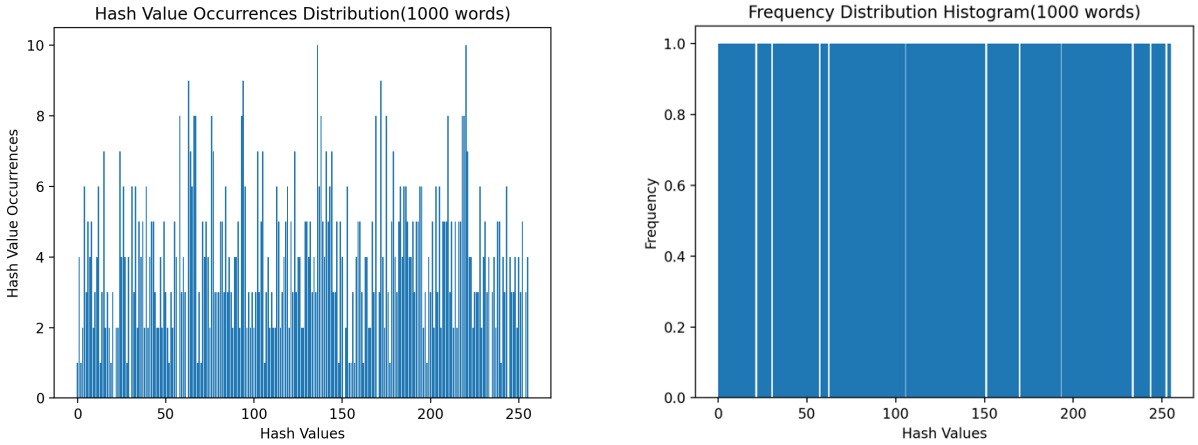

Figure 17. Hash value occurrences distribution and frequency distribution histogram (1000 words)

In the second test, we can see that a good uniformity of this function can be found since the input strings as 50 words. There are very small number of buckets emerged collision while most of the occurrences are evenly distributed among the different hash values (buckets). The similar trends but better performance of uniformity can be identified during the second test. The following table shows that a better standard deviation statistical data of the second test comparing to test one, which is based on the modified new strings.

Table 5: The standard deviation statistics of the occurrences among different hash value buckets

Input strings (modified)	Standard deviation
10 words	0.0
50 words	0.2444
100 words	0.4324
500 words	1.2598
1000 words	1.9335

As we can see, during the second test, a good uniformity of this function has achieved since 50 words input (0.2444 comparing to 0.433 of test 1). From the 500 words tests, the second tests revealed that its standard deviation is only around 50% (1.2598) as the first test (2.416). When the processed strings reaching up till 1000 words, a good uniformity can be proved by both the standard deviation figures (1.9335 comparing to 4.2898) and the "Frequency distribution histogram (1000 words)" (fewer white bars and more sparseness of the graph). Therefore, we can conclude that a good uniformity of this hash function is proved.

c) Secure hash functions are often embedded with secure hash algorithms which are a family of cryptographic functions designed to keep data secured [11]. There are three main features existing in secure hash algorithms: bitwise operations, modular additions, and compression functions. Furthermore, the results of secure hash functions are fixed-size strings which look nothing like the original [11]. The algorithms of secure functions normally perform as "one-way functions", which means that it is virtually infeasible to reverse the function's results back to initial inputs [11]. Moreover, Secure hash functions own three fundamental safety characteristics: pre-image resistance, second pre-image resistance, and collision resistance [11]. Secure hash functions are commonly used for passwords encryption as the server side [11].

Comparing these properties, the non-secure hash functions may perform well in terms of avoiding collision but the function itself may not perform as "one-way" which means the yielded results are invertible. Furthermore, the normal hash function may not include bitwise operations and compression functions. A common application of normal hash functions is to put objects into a hash table with as few collisions as possible, which normally do not need to perform neither bitwise operations nor compression. In addition, the results of normal hash functions are normally integers instead of strings. The easiest way to prove my hash function as non-secured is through identifying the results of the function. The outputs of this function are based on ASCII codes calculation. The generated results of the function are pure integers instead of encrypted strings. Moreover, the algorithm of this hash function does not include bitwise operations and compression functions which means the function's results can be inverted and the initial inputs can be found easily.

Bibliography

[1] C. Pfleeger, S. Pfleeger, and J. Margulies, "Security in Computing. 5th ed", [2015], United States of America: Pearson, 2015.

[2] M. Sharma and S. Gandhi, "Compression and Encryption: An Integrated Approach", [2012-07-05], url: [https://www.researchgate.net/publication/262979801_Compression_and_Encryption_An_Integrated_Approach]

[3] Indika, "Difference Between Data Compression and Data Encryption", [2011-07-08], url: [https://www.differencebetween.com/difference-between-data-compression-and-vs-data-encryption/]

[4] Edpresso Team, "What is hashing?", [2020], url: [https://www.educative.io/edpresso/what-is-hashing]

[5] M. Rouse, "Steganography", [2018-12], url: [https://searchsecurity.techtarget.com/definition/steganography].

[6] James Stanley, "How to defeat naive image steganography", [2016-04-27], url: [https://incoherency.co.uk/blog/stories/image-steganography.html]

[7] EC-Council, "WHAT IS STEGANOGRAPHY AND WHAT ARE ITS POPULAR TECHNIQUES?", [2018-05-22], url: [https://blog.eccouncil.org/what-is-steganography-and-what-are-its-popular-techniques/]

[8] Wikipedia, "Caesar cipher", [2020-11-11] url: [https://en.wikipedia.org/wiki/Caesar_cipher]

[9] Ore Asonibare, "Why are prime numbers used for constructing hash functions?", [2017-12-1], url: [https://www.quora.com/Why-are-prime-numbers-used-for-constructing-hash-functions]

[10] Wikipedia, "Hash function", [2020-11-19], url: [https://en.wikipedia.org/wiki/Hash_function#Properties]

[11] N. Landman, C. Williams, E. Ross, and J.Khim, "Secure Hash Algorithms", [2020-11-20], url: [https://brilliant.org/wiki/secure-hashing-algorithms/]