Olav Høysæther Opheim DSIK 2

INF140-Introduction to Cybersecurity

Overview of Cybersecurity

Question 1.

There are five core security attributes in a computer system: confidentiality, integrity, authenticity, accountability, and availability. Here is the explanation of each security attribution:

Confidentiality is preserving data and protecting it to prevent unauthorized parties to view the data. Social security numbers are an example of confidential information that is very important to keep confidential.

Integrity is protecting data and preventing unauthorized parties from modifying, tampering, or destroying data. An example of integrity is at the doctor's, we should be able to trust doctors that they provide us with the correct medication. Let's say you got prescribed a medication, but then another doctor with access to change this data swaps your medication to something else. This may cause harm to this patient because of false information.

Authenticity is a term used to assure that a transaction, message, or other information is sent from the source it claims to be from, and that the data has not been tampered with. To prove *authenticity* proof of identity is needed.

Accountability is the principle that an individual is trusted with authorization to control equipment, data and the individual is accountable if loss or misuse of equipment or data. An example is that if a system or website gets breached by hackers the website needs to be able to trace the entities uniquely This is because they need to find out what the hackers were doing, and to be able to hold them legally accountable for their actions.

Availability is ensuring that authorized parties have access to information services and that those services are reliable and accessible upon demand. An example of this would be if a customer has an account with a stockbroker. The customer then tries to log in to sell his stocks because the customer has heard rumors of bad news, but he can’t log in because the servers are down. This causes the customer to take a huge financial loss because the availability of the stockbroker was poor.

(Stallings and Brown, 2017, p. 32-35)

Question 2.

With these security attributes there comes threats and consequences. I will now list and briefly define the different kinds of threat consequences and the types of threat action that cause these consequences:

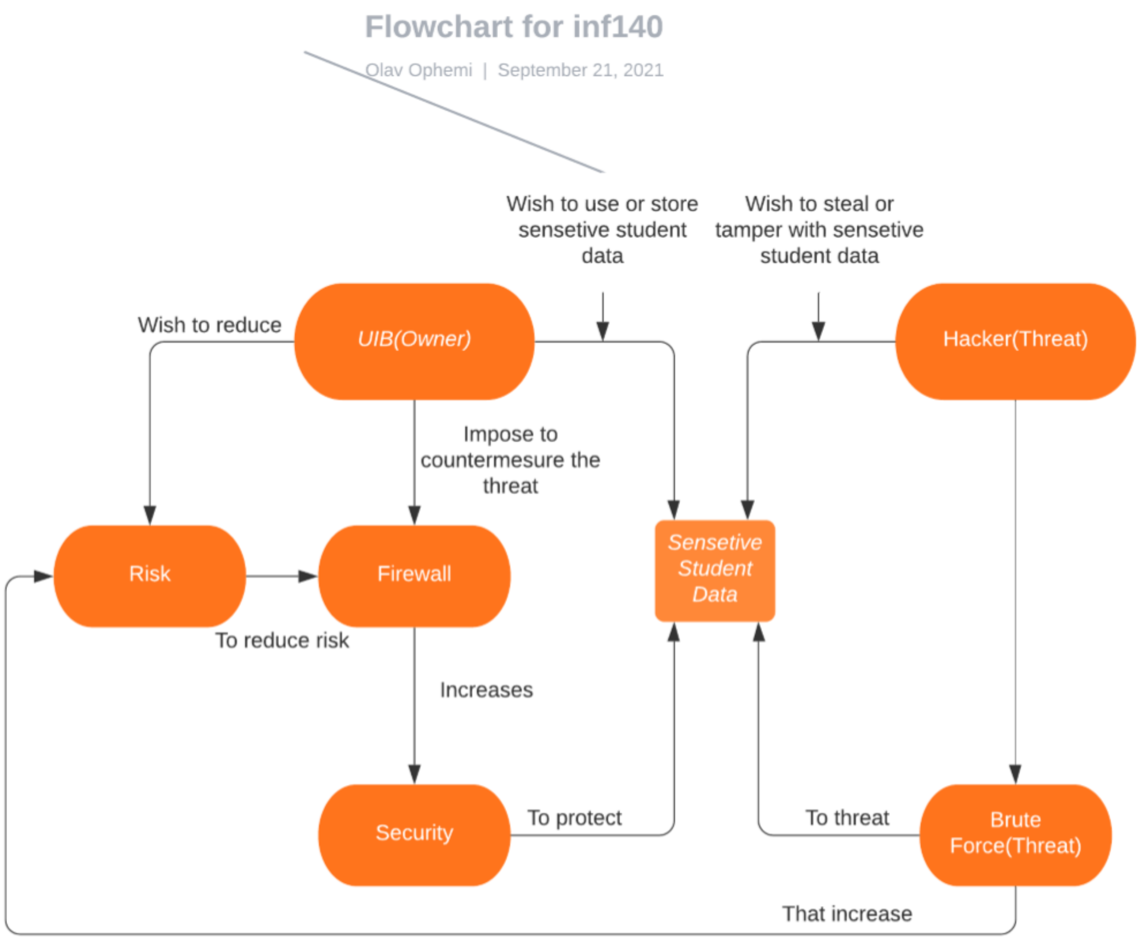
Threat Consequence	**Threat Action/Attack**
Unauthorized Disclosure Is the instance or event when an entity gains access to disclosed data which the entity should not be authorized to access.	**Exposure:** Sensitive data is directly released to unauthorized entity. **Interception:** Sensitive data traveling between authorized sources and destinations are directly accessed by unauthorized entity. **Inference:** When an unauthorized entity gains access to sensitive data indirectly by either reasoning from characteristics or if the entity only received parts of the sensitive data. **Intrusion:** The instance where an unauthorized entity manages to break past system security and gain access to sensitive data.
Deception Is the instance or event which might result to authorized entity receiving false data but believing that the data is true.	**Masquerade:** An unauthorized entity impersonates and authorized entity to gain access to a system or to perform a malicious act. **Falsification:** Event where false data tricks an authorized entity

	Repudiation: The instance when an entity tricks another entity by denying responsibility for an act
Disruption Is the instance or event where operations of system services and functions are interrupted or prevented from working the correct way.	**Incapacitation:** Deactivating a system component to prevent or interrupt system operation. **Corruption:** Accidentally changing system operations by modifying system data or functions. **Obstruction:** Threat action which interrupts and prevents system operation to deliver system services.
Usurpation Is the instance or event which results in an unauthorized entity receiving control of system services or functions.	**Misappropriation:** An entity takes upon himself unauthorized logical or physical control of a system resource. **Misuse:** Will result in a system component to perform a service or function that is harmful to the security of the system.

(Stallings and Brown, 2017, p. 40)

Question 3.

By referring to Figure 1.2 (Section 1.1, Chapter 1) I have come up with an example that involves all blocks in the figure. I will now go through the relations of entities and relevant techniques, and it will include a drawing of a similar figure that consists of concrete entities, techniques, and countermeasures:



Figur 1:Float chart (Opheim, 2021)

Explanation of the figure:

If we look at the figure I have created, we can see that "UIB" the owner of the "Sensitive Student Data" wishes to use or store that data. They also wish to reduce the risk since a new "Hacker" is increasing the threat by using different types of hacks or methods to try to break past the security. So UIB then wants to reduce that risk and threat and decided to impose a countermeasure which is a "Firewall". This "Firewall" will reduce the risk and increase the security to protect the "Sensitive Student Data" from threats that wish to steal or tamper with the "Sensitive Student Data". (Stallings and Brown, 2017, p. 38-39)

Cryptographic Tools

Question 4.

Known info:

Key 1 = jisuan

Key 2 = 3415726

Message: Cyber security is actually determined by the weakest link

Variable names shortened:

K1 = Key 1, K2 = Key 2, M = Message, T1 = Text 1, T2 = Text 2, T3 = Text 3, C = Cipher

Round 1; Encryption

Text 1

Info used:

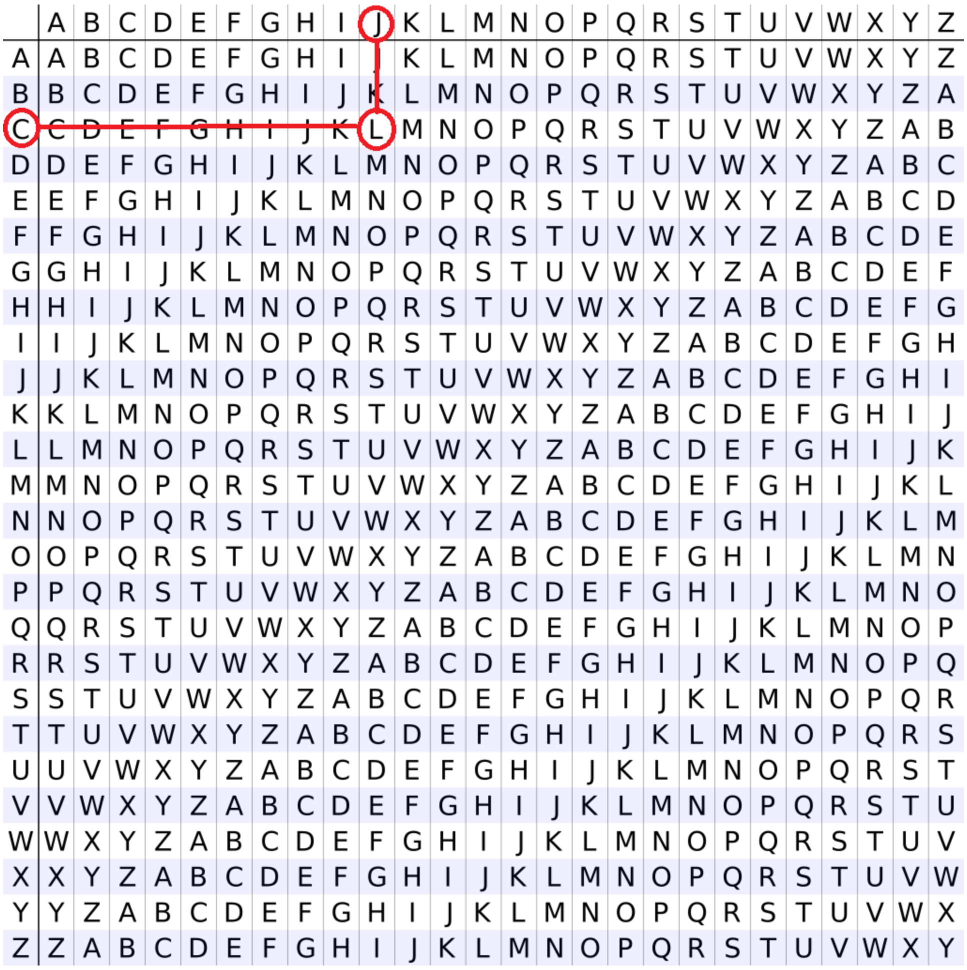
M and K1

Vigenère cipher

To encrypt the message, we need to use the Vigenère cipher, and you can see the picture below that displays the tableau. We use this tableau to encrypt the message.

How to find T1

To find T1 we need to use the tableau. The way I used it was to start with the first letter of the message which is "C" and then start from the top of the column and make my way down until I find row "C", then we use the first letter of K1 which is "J" and then follow the top row until we find column "J". Once row "C" and column "J" is found we can see that if we draw a line, (Horizontal line from the rows and vertical lines from the columns). At the point where the lines crosses are the letter we are looking for "L". Then we do these again until we have done each letter of the message.



	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
A	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
B	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	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	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
I	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
J	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	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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

Figur 2:The Vigenère Square (Fields, 2011)

(M, but the spaces are removed)

M: cybersecurityisactuallydeterminedbytheweakestlink

(K1, but we repeat the K1 untill K1 length = M length)

K1: jisuanjisuanjisuanjisuanjisuanjisuanjisuanjisuanj

Result:

T1: lgtyrfnkmlighqkucgdidfyqnbwlmvwmvvygqmoyaxnalfiat

Text 2

Info used:

T1 and K2

Transposition cipher

Now that we have figured out Text 1 we can use this as input together with key 2 to figure out Text 2.

How to find T2

The way T1 was figured out was to write K2 in the first row and that then decides that the row length should be 7 since the K2 length was 7. Then we proceed to write the T1 in the rows from left to right starting at the top. After every letter is inputted, we start by looking at the row with number "1" at the top. We go from top to bottom and write letter by letter, then we proceed to do the same with numbers "2", "3" and so on until we have done the last number "7". After this we have received the T2 and can now proceed to round 2.

3	4	1	5	7	2	6
L	G	T	Y	R	F	N
K	M	L	I	G	H	Q
K	U	C	G	D	I	D
F	Y	Q	N	B	W	L
M	V	W	M	V	V	Y
G	Q	M	O	Y	A	X
N	A	L	F	I	A	T

T1: lgtyrfnkmlighqkucgdidfyqnbwlmvwmvvygqmoyaxnalfiat

K2: 3415726

Result:

T2: tlcqwmlfhiwvaalkkfmgngmuyvqayignmofnqdlyxtrgdbvyi

Round 2; Encryption

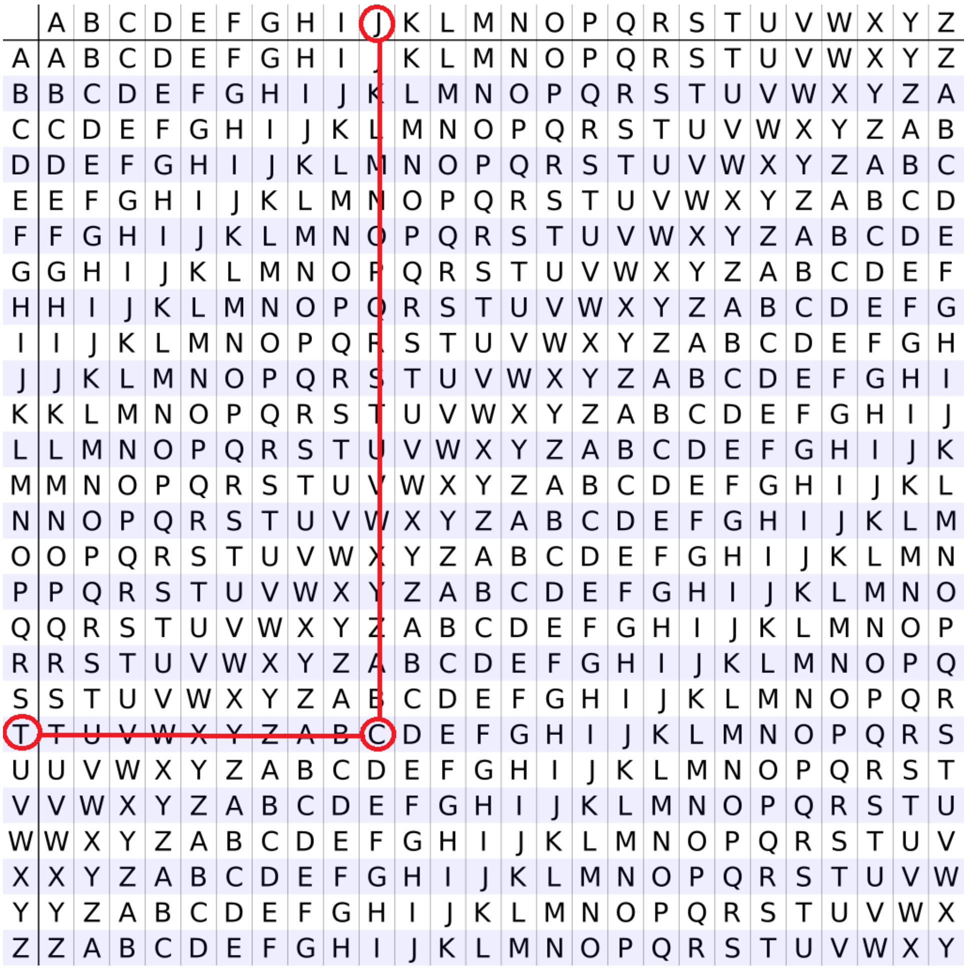
Text 3

Info used:

T2 and K1

Vigenère cipher & how to find T3

In round 2 we use text 2 as input and do the same process as we did in round 1 with the message M. We first find the letter “j” at the top row and then find letter “t” at column 1 moving down. Once both letters are found we draw a line, and we can see that the lines cross at letter “c” which is the first letter of text 3. Then we do this process for every letter.



	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
A	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
B	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	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	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
I	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
J	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	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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

T2: tlcqwmlfhiwvaalkkfmgngmuyvqayignmofnqdlyxtrgdbvyi

K1: jisuanjisuanjisuanjisuanjisuanjisuanjisuanjisuanj

Result

T3: ctukwzunzcwijideksvofamhhdiuyvpveifazldsxgaovvvlr

Ciphertext:

Info used:

T3 and K2

Transposition cipher & how to find ciphertext

Now once we have found the T3 we use that as input in the transposition cipher to find the final cipher text together with K2 and the length of K2 decides the box length which is 7, and since we know that T3 is 49 letters long we can take 49/7 = 7 and that means that the columns are also 7 boxes in length.

3	4	1	5	7	2	6
C	T	U	K	W	Z	U
N	Z	C	W	I	J	I
D	E	K	S	V	O	F
A	M	H	H	D	I	U
Y	V	P	V	E	I	F
A	Z	L	D	S	X	G
A	O	V	V	V	L	R

T3: ctukwzunzcwijideksvofamhhdiuyvpveifazldsxgaovvvlr

K2: 3415726

Result:

When using the same method as in Round 1, Text 2 we figure out the ciphertext:

C = uckhplvzjoiixlcndayaatzemvzokwshvdvuifufgrwivdesv

Decryption:

To prove that the encryption works we will decrypt the ciphertext and get the original message. To decrypt we must do the same process but in the opposite way, and that means that we will start with the transposition cipher, then Vigenère, then transposition, and then last Vigenère.

Round 2; Decryption

From ciphertext to text 3

Info used:
K2 and C

Transposition cipher

We start the decryption by using the transposition cipher, but we input the data a bit differently. Since we only know K2 and C means we need to do the transposition cipher in another way.

How to find T3

So, since we know key 2 we know the length of the rows which is 7 and we know that the ciphertext length is 49 and 49/7 = 7 so that means each column is 7 in length. Since we know the cipher design we know that in the transposition cipher “text 3” is inputted letter by letter and start in the top right corner and fills each empty box from left to right, but the ciphertext is written/read out by starting with column “1” and goes down that column so that means that the first 7 letters of the ciphertext are supposed to be inputted in the column number “1”. By doing this once all the letters are inputted, we will be able to read out text 3. So, by doing this we receive the following table:

3	4	1	5	7	2	6
C	T	U	K	W	Z	U

N	Z	C	W	I	J	I
D	E	K	S	V	O	F
A	M	H	H	D	I	U
Y	V	P	V	E	I	F
A	Z	L	D	S	X	G
A	O	V	V	V	L	R

C: uckhplvzjoiixlcndayaatzemvzokwshvdvuifufgrwivdesv

K2: 3415726

Result:

We know that text 3 is inputted normal by starting at the top right and moving to then left. So that means that we now can read out Text 3 which is:

Text 3: ctukwzunzcwijideksvofamhhdiuyvpveifazldsxgaovvvlr

From text 3 to text 2

Info used:

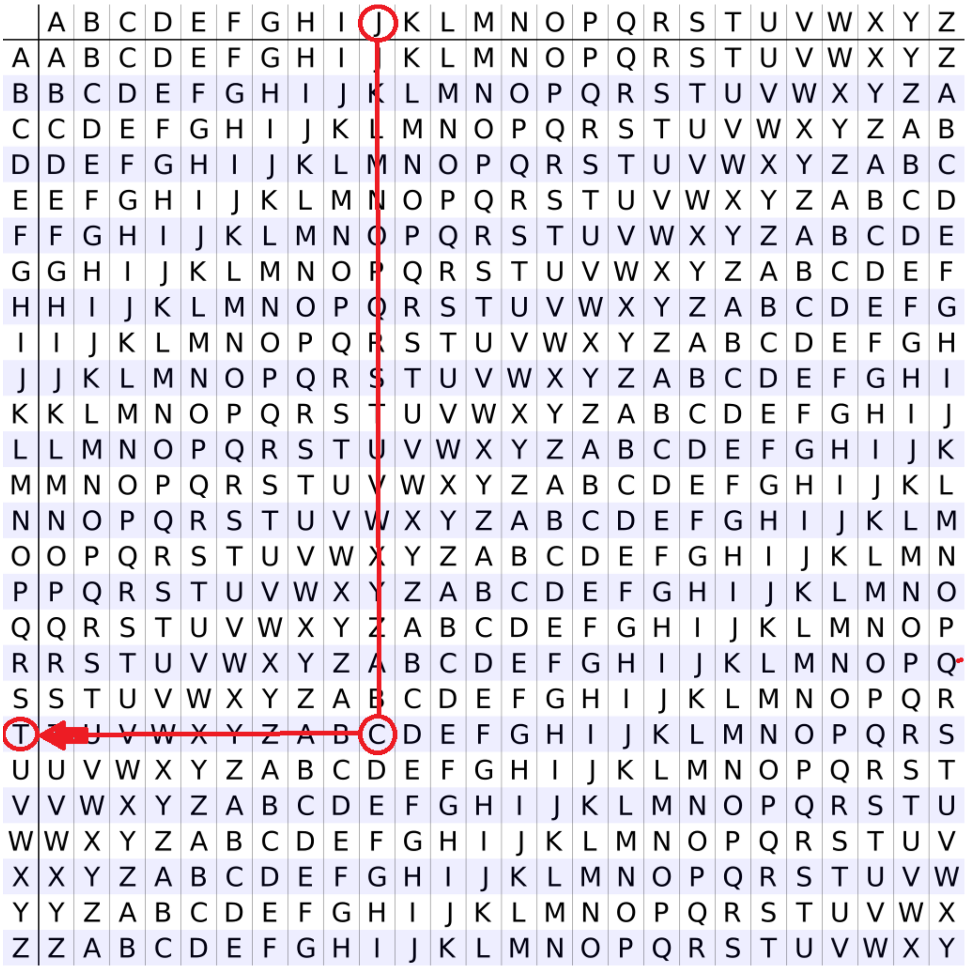
T3 and K1

Vigenère chiper

Since we found Text 3 we can now use that to decrypt the next text 2. Text 2 was found by Vigenère cipher. To find text 2 we must use key 1 as shown in the cipher design.

How to find T2

We know that the Vigenère cipher is found by using the tableau and first finding the first letter of the message in the column and then proceeded to follow that row until the top row letter is the same as the first letter of the key 1. Since we are decrypting, we need to do the opposite, so we start at the top column and find the letter “j” since that’s the first letter of the key, then we follow that column downwards until we find the first letter of Text 3. Then we follow that row to the left until we are at the last letter at that row and that will give us the first letter of text x. Then we do this process for each letter until each letter has been found and we will receive the following text:



	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
A	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
B	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	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	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
I	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
J	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	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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

T3: ctukwzunzcwijideksvofamhhdiuyvpveifazldsxgaovvvlr

K1: jisuanjisuanjisuanjisuanjisuanjisuanjisuanjisuanj

Result:

T2: tlcqwmlfhiwvaalkkfmgngmuyvqayignmofnqdlyxtrgdbvyi

Round 1:

From text 2 to text 1

Info used

T2 and K2

Transposition chiper:

Like in round 2 of decryption we need to do the transposition cipher a different way since we only know T2 and K2.

How to find T1

Like we did in round 2 to find text 3 we need to input text 2 by starting with column number “1” and then inputting the 7 first letters starting from the top and moving down then once column “1” is filled we move on to column”2” and so on. Once we have filled in all of the letters, we will be able to read out text 1 by starting at the top right.

3	4	1	5	7	2	6
L	G	T	Y	R	F	N
K	M	L	I	G	H	Q
K	U	C	G	D	I	D
F	Y	Q	N	B	W	L
M	V	W	M	V	V	Y
G	Q	M	O	Y	A	X
N	A	L	F	I	A	T

T2: tlcqwmlfhiwvaalkkfmgngmuyvqayignmofnqdlyxtrgdbvyi

K2: 3415726

Result:

T1: lgtyrfnkmlighqkucgdidfyqnbwlmvwmvvygqmoyaxnalfiat

From text 1 to message

Info used

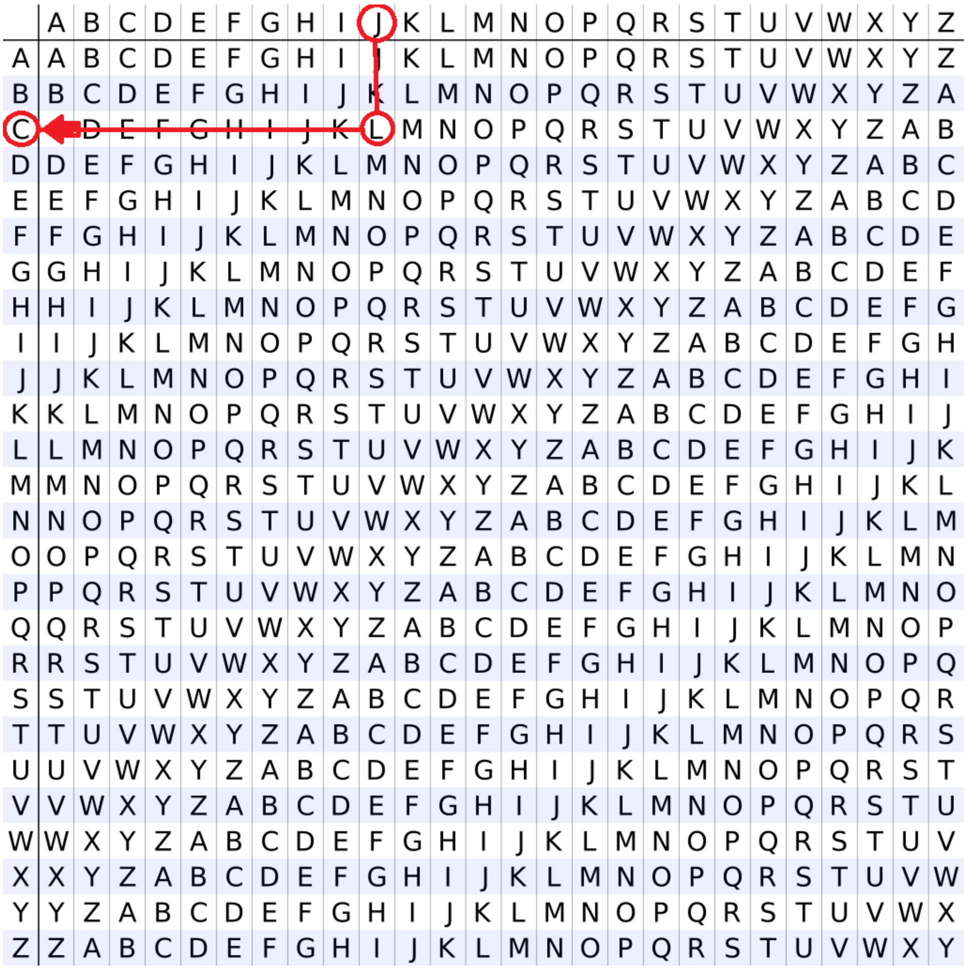
T1 and K1

Vigenère cipher:

In this last step to decrypt the ciphertext we need to use the Vigenère cipher one last time. We only know T1 and K1 so we need to do the Vigenère cipher a different way.

How to find M

To find the message we need to do a Vigenère cipher. We need to do the same as we did in round 2 to find text 2. We start with the first letter “j” in K1 and find that in the top row, then once we have found “j” we move down the column “j” and find the first letter of text 1 wich is “l”. Then when we have found “l” we follow that row to the left until we are at the last letter on that row which gives us the letter “c”. Then we do this process until all of the letters have been found.



	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
A	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
B	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	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	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
I	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
J	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	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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

T1: lgtyrfnkmlighqkucgdidfyqnbwlmvwmvvygqmoyaxnalfiat

K1: jisuanjisuanjisuanjisuanjisuanjisuanjisuanjisuanj

Result:

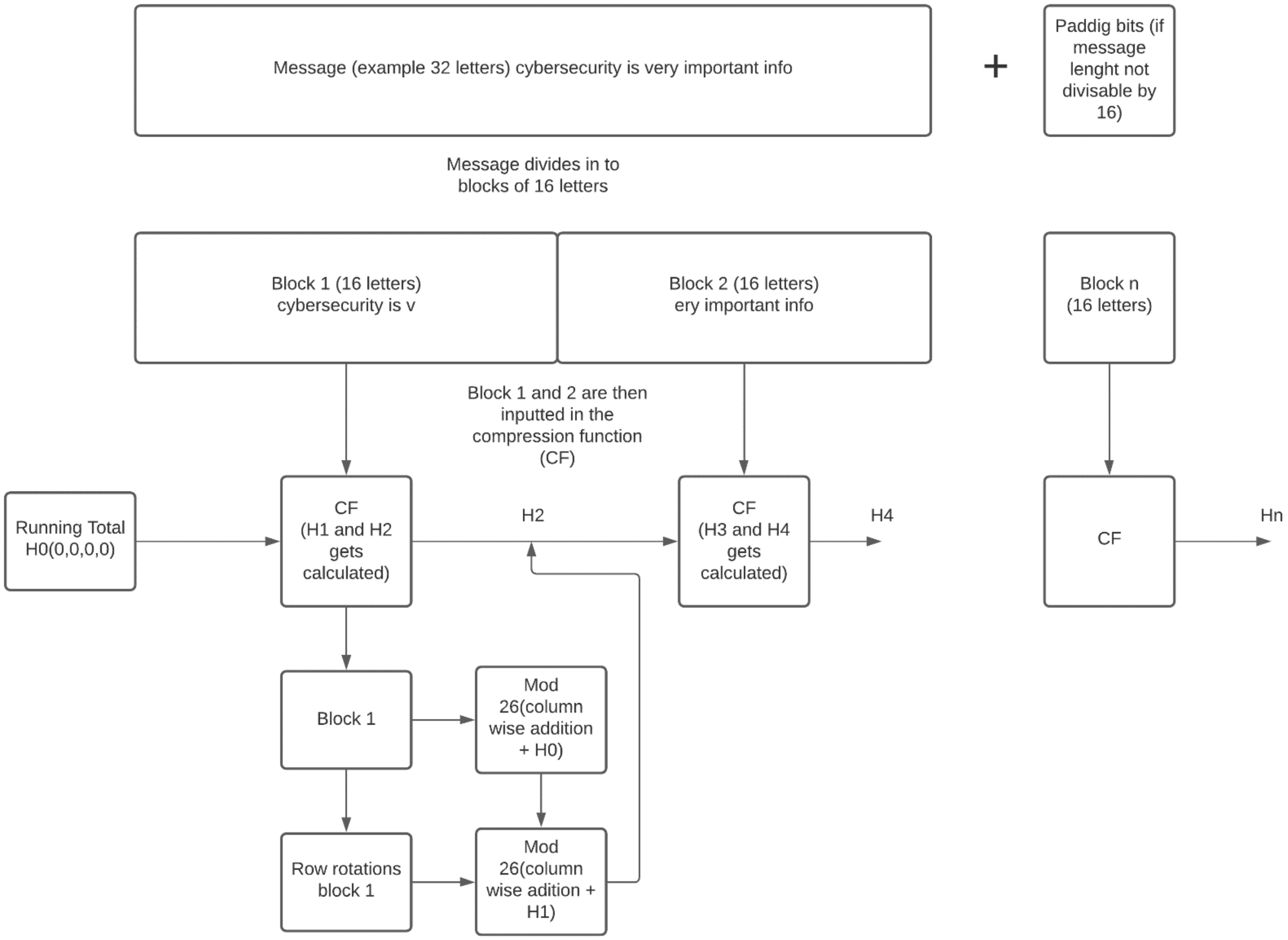
M: cybersecurityisactuallydeterminedbytheweakestlink

This shows us that our encryption is correct because we can decrypt it and receive the same message.

Question 5.

1) Create a float chart and explain it

In this task we were asked to draw a figure of the overall TTH logic and the compression function logic.


Message (example 32 letters) cybersecurity is very important info
+
Paddig bits (if message lenght not divisable by 16)
Message divides in to blocks of 16 letters
Block 1 (16 letters) cybersecurity is v
Block 2 (16 letters) ery important info
Block n (16 letters)
Block 1 and 2 are then inputted in the compression function (CF)
Running Total H0(0,0,0,0)
CF (H1 and H2 gets calculated)
H2
CF (H3 and H4 gets calculated)
H4
CF
Hn
Block 1
Mod 26(column wise addition + H0)
Row rotations block 1
Mod 26(column wise adition + H1)

Figur 3:TTH and CF drawing (Opheim, 2021)

Short explanation of the figure

So, if we start at the top, we see that we have a message consisting of a certain number of letters. If the number of letters is divisible by 16 we procced to divide the letters into boxes of 16 letters each, but if the number letters aren't divisible by 16 we need to add padding bits to make the number of letters divisible by 16. The added padding pits are nulls. Once the number of letters is divisible by 16 we divide letters into blocks consisting of 16 letters. Once the blocks are defined each block is inputted into the CF box (compression function). If we look at block 1 we can see that this block is inputted into CF and then the starting running total h0 (0,0,0,0) is added to the calculation of the running total for H1 and then H1 is added to the calculation for H2. The CF box leads us down to block 1 (these boxes below CF explain the CF process). Block 1 then proceeds to go two ways, first way is to the right where we calculate the running total for H1, the second way leads us down to where the rows in block 1 are rotated (row 1 rotates 1 step to the left, row 2 rotates 2 steps, row 3 rotates 3 steps and row

4 reverses). The first way and second way then meet in the same box where the rotated block 1 is inputted and then the running total of the rotated block 1 is calculated + H1(running total). The same CF process is done with block 2 but with a different running total.

2) How to calculate the 48-letter hash function

We were asked to calculate the hash function for the 48-lett message "He left twenty million us dollars to his beloved children".

Step 1. We calculated how many 4x4 blocks we needed so we took 48/16 = 3

Step 2. We created 3 4x4 blocks using an excel spreadsheet and then filled each square with one letter, and we did that until each square was filled.

Step 3. Once each square was filled, we created 3 more 4x4 blocks to the right of the other blocks and instead of writing letters these blocks was filled with numbers. We filled each square with a number and the number is defined by the letter, example:

A = 0, B = 1, C = 2 … and so on.

Step 4. We now start on creating the blocks for round 2, we created 3 blocks for letters and 3 for numbers, but they got filled in a different way. So like explained in the task the first row the first letter is moved to the left (out of the block, but that means it moves all the way to the back) so if you look at Round 2 the top block with letters you can see that the first row is written out E L E H, that's because the row was written H E L E first but then the "H" was moved to the back E L E "H". So as explained row 1 each letter moves 1 step to the left, row 2 each letter moves 2 steps to the left, row 3 each letter moves 3 steps to the left and the last row, row 4 the letters are reversed.

Step 5. Once we have moved the letters in all the 3 blocks, we can move on top the 3 blocks where the numbers are going to be. The same thing happens to the numbers they need to move as many steps as the letters, so that they line up with the same square as the letter they "represent" etc. if A = 0 and A is in square 2 and 0 is in square 3, then square 3 needs to move 1 step left.

Step 6. Now that all the blocks are filled out properly, we can move on to calculating the running total.

So, the numbers need to be added together in each column mod 26. So, let's take the first column in round 1:

7 + 5 + 4 + 12 = 28 – 26 (mod 26) = 2, and the same with the other columns in that box and that gives us the running total (2, 18, 8 ,9) as displayed in the image beside the text "Result 1). Now we can move over to round 2 and calculate the running total for those columns:

4+ 19 + 24 + 11 = 56 – 26 –26 = 6 and then we also needs to add on the running total form the column in round 1 so the result is then:

6 + 2 = 8

The same happens to the other columns so block 1 column 2 in round 2

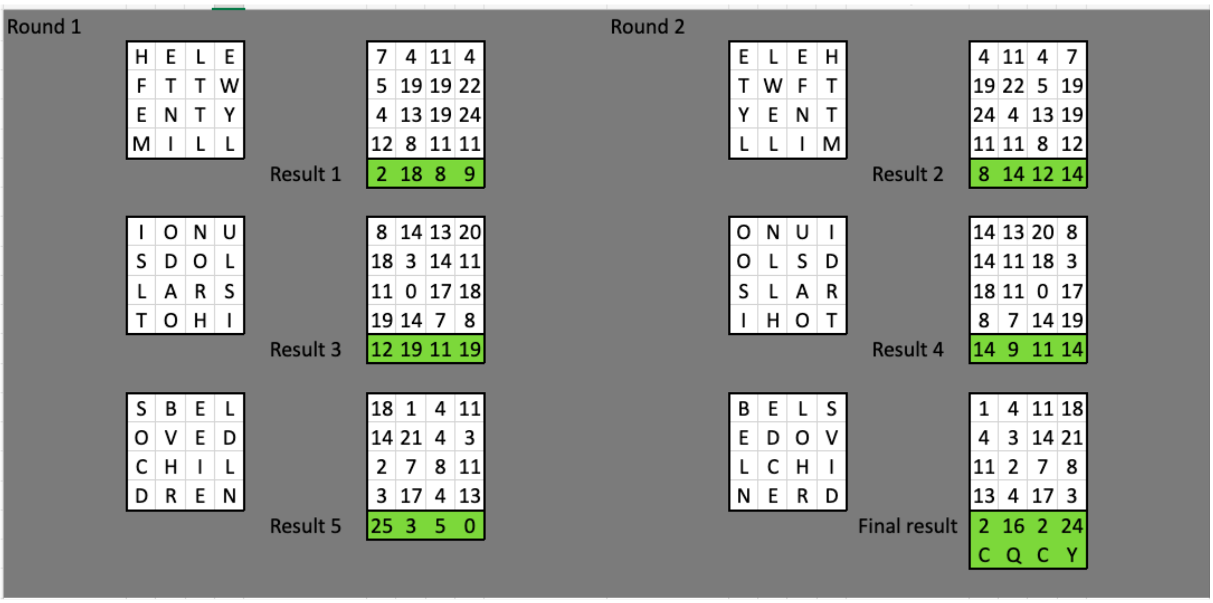
Answer + 18 since 18 is the running total in block 1 round 1 column 2

Once the running total is calculated in block 1 round 2 we can go back to round 2 but then down to block 2 and now the running total from block 1 round 2 will become the one that is added on.

Step 7. Once all the running totals are calculated we are left with the final result which is:

Final result

(2, 16, 2, 24) and we can translate the numbers to the letters C Q C Y which is the hash function for the 48-letter message.



Round 1

H	E	L	E
F	T	T	W
E	N	T	Y
M	I	L	L

	7	4	11	4
	5	19	19	22
	4	13	19	24
	12	8	11	11
Result 1	2	18	8	9

I	O	N	U
S	D	O	L
L	A	R	S
T	O	H	I

	8	14	13	20
	18	3	14	11
	11	0	17	18
	19	14	7	8
Result 3	12	19	11	19

S	B	E	L
O	V	E	D
C	H	I	L
D	R	E	N

	18	1	4	11
	14	21	4	3
	2	7	8	11
	3	17	4	13
Result 5	25	3	5	0

Round 2

E	L	E	H
T	W	F	T
Y	E	N	T
L	L	I	M

	4	11	4	7
	19	22	5	19
	24	4	13	19
	11	11	8	12
Result 2	8	14	12	14

O	N	U	I
O	L	S	D
S	L	A	R
I	H	O	T

	14	13	20	8
	14	11	18	3
	18	11	0	17
	8	7	14	19
Result 4	14	9	11	14

B	E	L	S
E	D	O	V
L	C	H	I
N	E	R	D

	1	4	11	18
	4	3	14	21
	11	2	7	8
	13	4	17	3
Final result	2	16	2	24
	C	Q	C	Y

Figur 4:48-letter hash calculación (Opheim, 2021)

3) How to demonstrate a weakness of TTH, by finding a new 48 letter block that produces the same hash function

I found a weakness in the TTH because as you see below on the picture we receive the same running total at the end, but the letters are not in the same order and that means that a message with different letter order can give the same hash value.

Explanation and step by step

As we know the process in round 2 is to move1 step to the left in row 1, 2 in row 2, and 3 in row 3 and reverse in row 4. So, let's say we got ABCD in row 1, and in round 2 after the letters has moved 1 step to the left the order will be BCDA. So, to get BCDA to go back to the original order, we need to shift it 3 times, and that means that we need to move the numbers row 1 to row 3 and the numbers row 3 to row 1. Because it is the "rows" that decide how many steps the letters/numbers should take. So, if we move the letters in row 1 to row 3, we can shift BCDA to ABCD. (This is just an explanation) Now let's move on to the step by step on how I managed to find a different 48-letter message that produces the same hash function

Step 1. We create 3 4x4 blocks for letters and 3 4x4 blocks for round 1 and the same for round 2 and we get the same layout as in the last task. Then instead of using the old message we use the message that was displayed at the end after the letters reorder.

Step 2. Once all the letters were inputted, we did the same as last time by writing in the number that represents the letter.

Step 3. Now that both the letters and numbers are filled out in Round 1 we can move on to Round 2, and it's not the tricky part begins. In Round 2 we first begin by filling in the same letters and numbers from round 2.

Step 4. Once all the letters and numbers are filled in, we start by swapping letters in row 1 with 3 and 3 with 1. (The same for numbers so numbers in row 1 go in row 3 and 3 to 1). Once we rows have been n rearranged, we can proceed to move them to the steps they are supposed to take.

Round 1 row 1 ELEH is now rearranged in Round 2 row 3 to HELE

Round 1 row 3 YENT is now rearranged in Round 2 Row 1 to ENTY

And so on...

Once all the letters have been rearranged, we can look at round 1 and see the message/letters:

ELEH TWFT YENT LLIM ONUI OLSD SLAR IHOT BELS EDOV LCHI NERD

For the numbers they need to move the same steps as the letter they represent.

Step 5. Now that all the numbers and letter are both filled and reorder the correct way, we can procced to calculate the running total the same way we did in the last task.

Step 6. Once all the running totals are calculated we receive the final running total that displays the hash function

Final result

(2, 16, 2, 24) and we can translate that to the letters

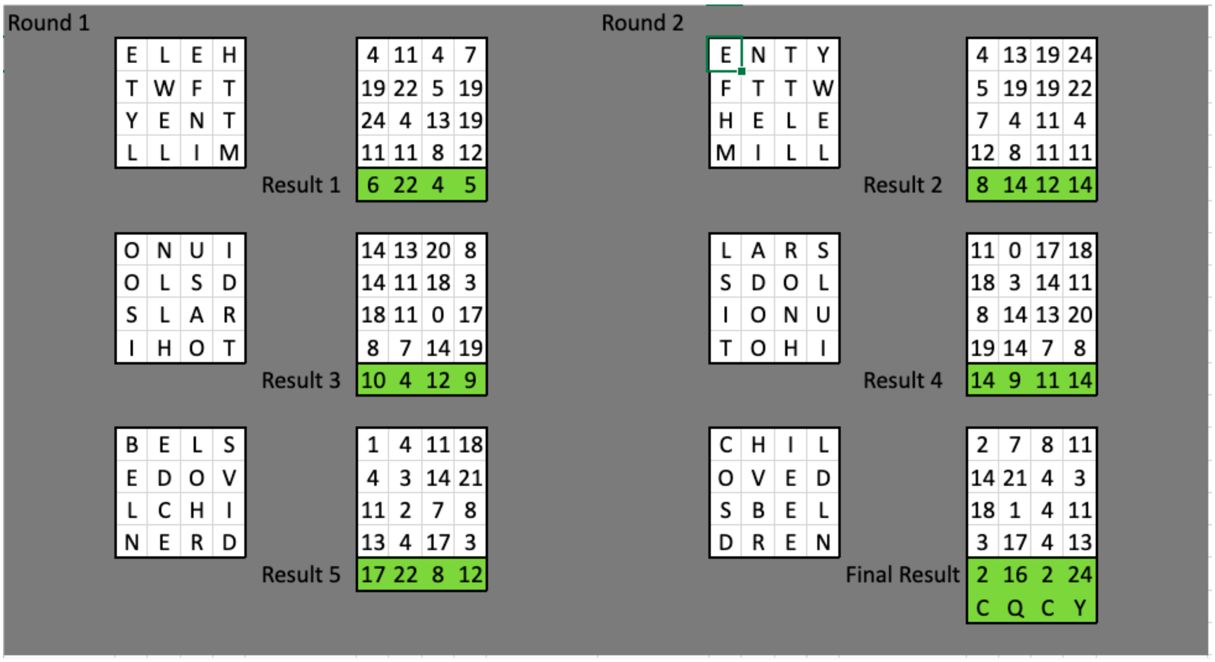
C Q C Y which is the same HASH as the last task

This means that the letters:

ELEH TWFT YENT LLIM ONUI OLSD SLAR IHOT BELS EDOV LCHI NERD

Produces the same hash function as the message:

He left twenty million us dollars to his beloved children



Round 1

E	L	E	H
T	W	F	T
Y	E	N	T
L	L	I	M

	4	11	4	7
	19	22	5	19
	24	4	13	19
	11	11	8	12
Result 1	6	22	4	5

Round 2

E	N	T	Y
F	T	T	W
H	E	L	E
M	I	L	L

	4	13	19	24
	5	19	19	22
	7	4	11	4
	12	8	11	11
Result 2	8	14	12	14

O	N	U	I
O	L	S	D
S	L	A	R
I	H	O	T

	14	13	20	8
	14	11	18	3
	18	11	0	17
	8	7	14	19
Result 3	10	4	12	9

L	A	R	S
S	D	O	L
I	O	N	U
T	O	H	I

	11	0	17	18
	18	3	14	11
	8	14	13	20
	19	14	7	8
Result 4	14	9	11	14

B	E	L	S
E	D	O	V
L	C	H	I
N	E	R	D

	1	4	11	18
	4	3	14	21
	11	2	7	8
	13	4	17	3
Result 5	17	22	8	12

C	H	I	L
O	V	E	D
S	B	E	L
D	R	E	N

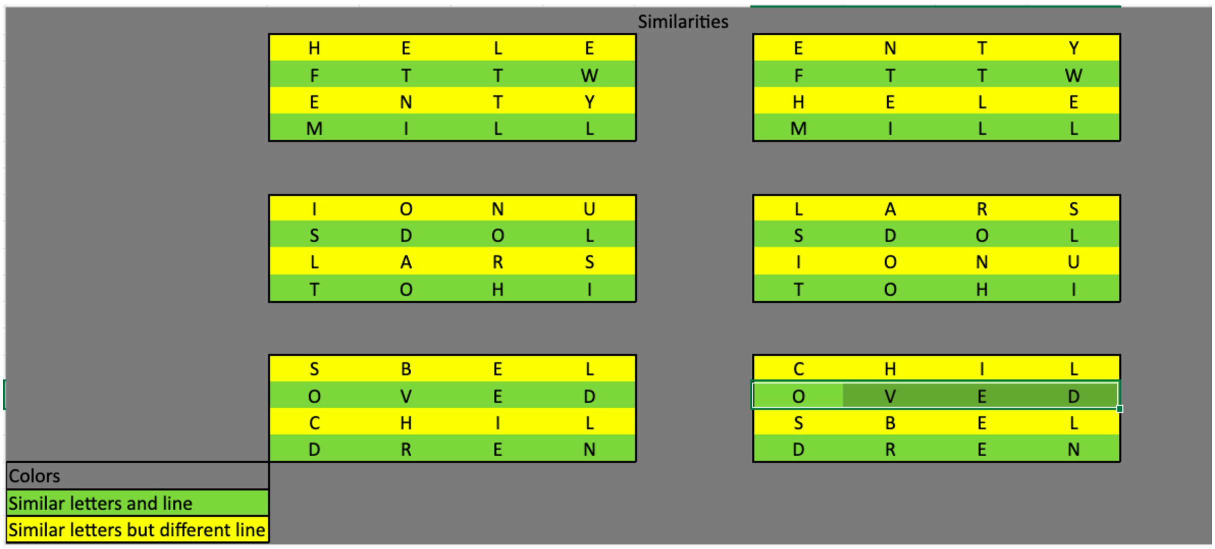
	2	7	8	11
	14	21	4	3
	18	1	4	11
	3	17	4	13
Final Result	2	16	2	24
	C	Q	C	Y

Figur 5:Demonstrating TTH weakness (Opheim, 2021)

Similarities

In the image below you can see that the green color displays similar letters on a similar line, and the yellow color displays similar letters but on different lines and that is because what we did was just to swap to lines to move the letters in to their original square, but to do that they needed to be moved to a different row.

In the image below we display what we explained that the rows marked yellow are those who have swapped but the green ones have not moved.



Similarities

H	E	L	E
F	T	T	W
E	N	T	Y
M	I	L	L

E	N	T	Y
F	T	T	W
H	E	L	E
M	I	L	L

I	O	N	U
S	D	O	L
L	A	R	S
T	O	H	I

L	A	R	S
S	D	O	L
I	O	N	U
T	O	H	I

S	B	E	L
O	V	E	D
C	H	I	L
D	R	E	N

C	H	I	L
O	V	E	D
S	B	E	L
D	R	E	N

Colors
🟩 Similar letters and line
🟨 Similar letters but different line

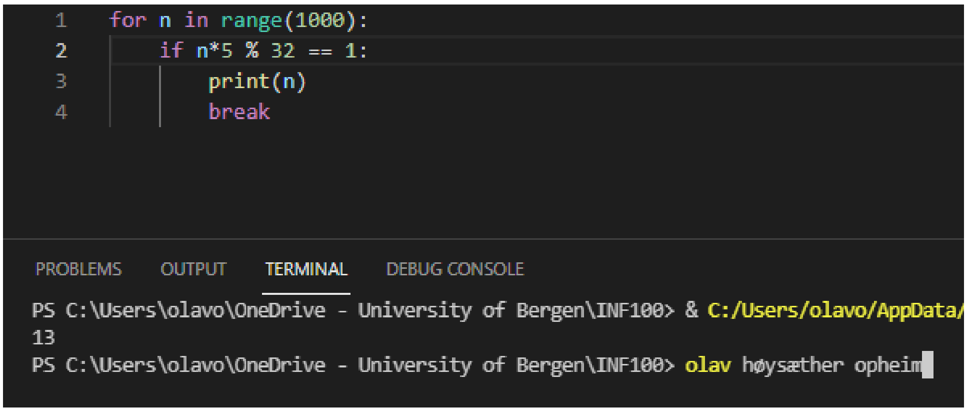
Figur 6:TTH message similarities (Opheim, 2021)

Question 6.

Part (1) Preforming encryption and decryption using RSA algorithms

1) p = 3; q = 17, e = 5; M = 6

1. P = 3 and q = 17
2. Calculating n = pq = 3 * 17 = 51
3. Calculating x(n) = (p – 1)(q – 1) = 2 * 16 = 32
4. e = 5, because it is relative to x(n) = 32 and less than x(n)
5. Now we determine d such as de mod 32 = 1 and d < 32. That means that d = 13, because 13 * 5 mod 32 = 1, we found this using python:



```
for n in range(1000):
    if n*5 % 32 == 1:
        print(n)
        break
```

```
PS C:\Users\olavo\OneDrive - University of Bergen\INF100> & C:/Users/olavo/AppData/
13
PS C:\Users\olavo\OneDrive - University of Bergen\INF100> olav høysæther opheim
```

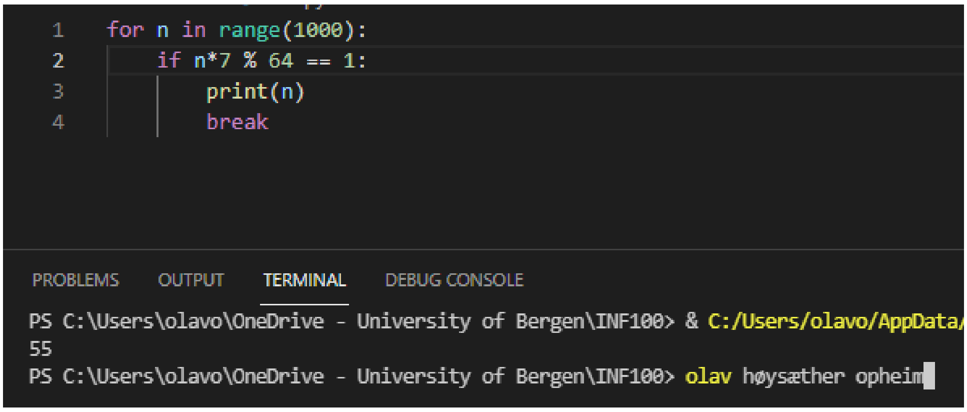
Figur 7:Python calculation (Opheim, 2021)

6. Encryption: M < n, C = M**e mod n = 6**5 mod 51 = 24
7. Decryption: M = C**d mod n = 24**13 mod 51 = 6

(Li, 2021)

2) p = 5; q = 17, e = 7; M = 4

1. p = 5 and q = 17
2. Calculating n = pq = 5 * 17 = 85
3. Calculating x(n) = (p – 1)(q – 1) = 4 * 16 = 64
4. e = 7, because it is relative to x(n) = 64 and less than x(n)
5. Now we determine d such as de mod 64 = 1 and d < 64. That means that d = 55, because 55 * 7 mod 64 = 1, we found this using python:



```
for n in range(1000):
    if n*7 % 64 == 1:
        print(n)
        break
```

```
PS C:\Users\olavo\OneDrive - University of Bergen\INF100> & C:/Users/olavo/AppData/
55
PS C:\Users\olavo\OneDrive - University of Bergen\INF100> olav høysæther opheim
```

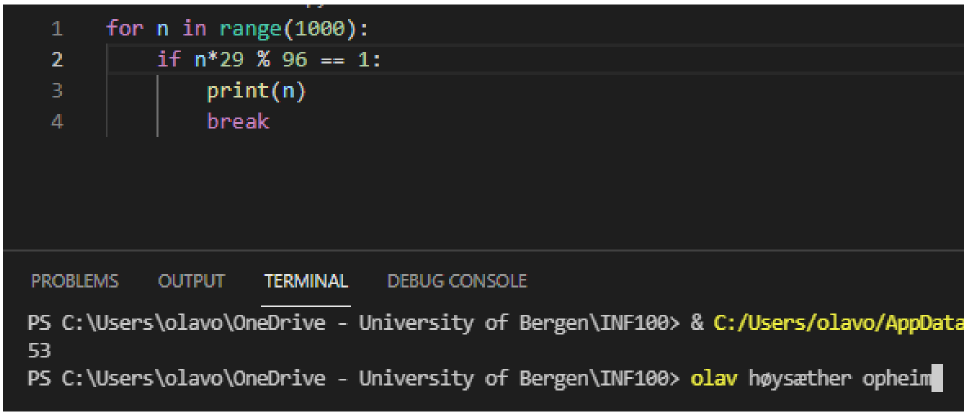
Figur 8:Python calculation (Opheim, 2021)

6. M < n, C = M**e mod n = 4**7 mod 85 = 64
7. Decryption: M = C**d mod n = 64**55 mod 85 = 4

(Li, 2021)

3) p = 7; q = 17, e = 29; M = 7

1. p = 7 and q = 17
2. Calculating n = pq = 7 * 17 = 119
3. Calculating x(n) = (p – 1)(q – 1) = 6 * 16 = 96
4. e = 29, because it is relative to x(n) = 96 and less than x(n)
5. Now we determine d such as de mod 96 = 1 and d < 96. That means that d = 53, because 53 * 29 mod 96 = 1, we found this using python:



```
for n in range(1000):
    if n*29 % 96 == 1:
        print(n)
        break
```

```
PS C:\Users\olavo\OneDrive - University of Bergen\INF100> & C:/Users/olavo/AppData
53
PS C:\Users\olavo\OneDrive - University of Bergen\INF100> olav høysæther opheim
```

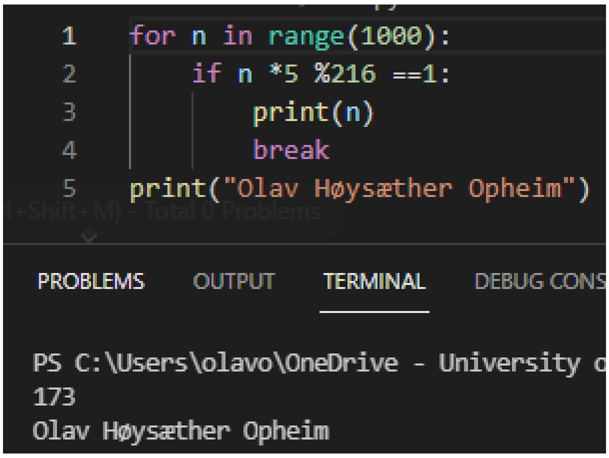
Figur 9:Python calculation (Opheim, 2021)

6. M < n, C = M**e mod n = 7**29 mod 119 = 91
7. M = C**d mod n = 91**53 mod 119 = 7

(Li, 2021)

Part (2)

1. P = 13 and q = 19
2. Calculating n = pq = 13 * 19 = 247
3. Calculating x(n) = (p – 1)(q – 1) = 12 * 18 = 216
4. E =5, because it is relative to x(n) = 198 and less than x(n)
5. Now we determine d such as de mod 198 = 1 and d < 198. That means that d = 173, because 173 * 5 mod 216 = 1, we found this using python:

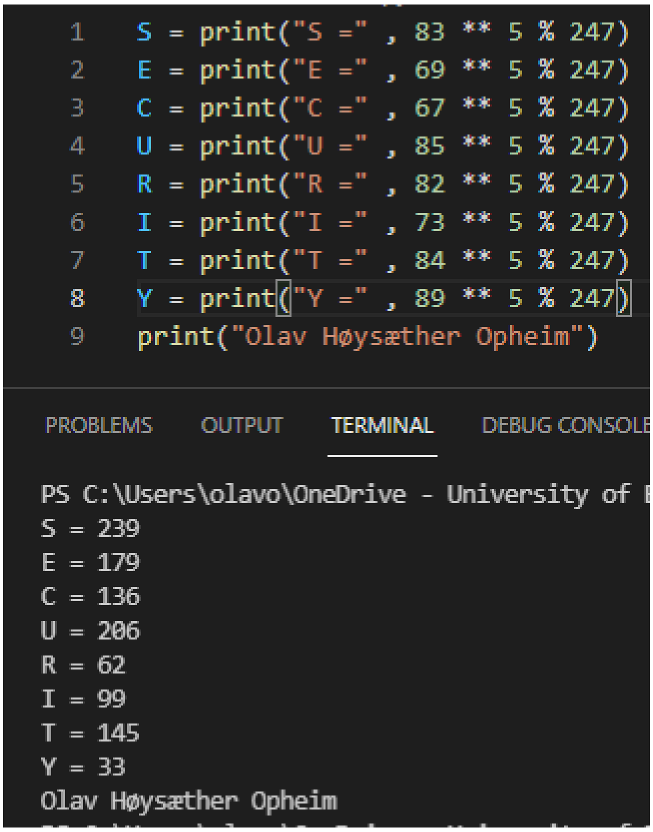


```
for n in range(1000):
    if n *5 %216 ==1:
        print(n)
        break
print("Olav Høysæther Opheim")
```

```
PS C:\Users\olavo\OneDrive - University o
173
Olav Høysæther Opheim
```

Figur 10:Python calculation (Opheim, 2021)

6. M1 = S, M2 = E, M3 = C, M4 = U, M5 = R, M6 = I, M7 = T, M8 = Y
7. Decimal: S = 83, E = 69, C = 67, U = 85, R = 82, I = 73, T = 84, Y = 89
8. Formula: C = M**e mod n, and we used python to calculate this:



```python
S = print("S =" , 83 ** 5 % 247)
E = print("E =" , 69 ** 5 % 247)
C = print("C =" , 67 ** 5 % 247)
U = print("U =" , 85 ** 5 % 247)
R = print("R =" , 82 ** 5 % 247)
I = print("I =" , 73 ** 5 % 247)
T = print("T =" , 84 ** 5 % 247)
Y = print("Y =" , 89 ** 5 % 247)
print("Olav Høysæther Opheim")
```

```
PS C:\Users\olavo\OneDrive - University of
S = 239
E = 179
C = 136
U = 206
R = 62
I = 99
T = 145
Y = 33
Olav Høysæther Opheim
```

Figur 11:Python calculation (Opheim, 2021)

Then we take the encrypted numbers:

239 179 136 206 62 99 145 33

Translate these into hex using an extended ASCII table (ASCII, Unknown) and we get the numbers:

EF B3 88 CE 3E 63 91 21

(Li, 2021)

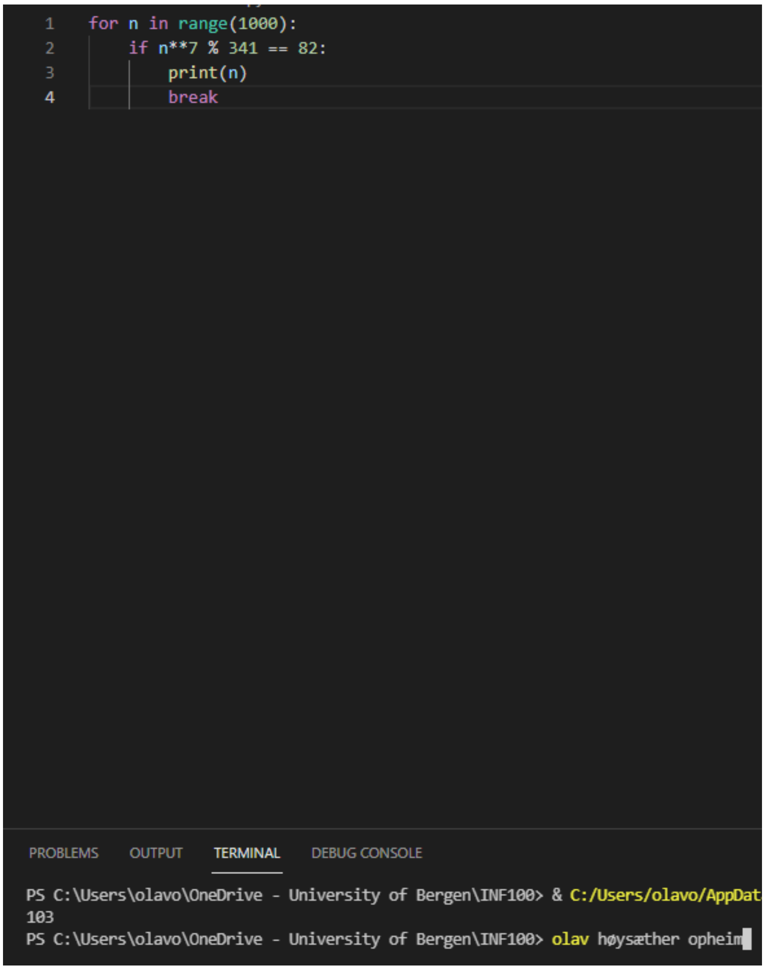
Question 7.

In this task, we are asked to find the Vigenère key used by Bob and find out what the original message from Bob is.

We know that n = 341, e = 7

First we start by decrypting 82 in the message to find the key to the Vigenère cipher used by Bob. To find this we need to use the RSA formula

1. Prime factoring 341 gives us the $p = 31$ and $q = 11$
2. Calculating $n = pq = 31 * 11 = 341$
3. Calculating $x(n) = (p – 1)(q – 1) = 30 * 10 = 300$
4. $E = 7$
5. Now we determine d such as de mod 300 = 1 and $d < 300$. That means that ed mod x(n) = 1 and gives us the equation $300 * 1 + 1 = 301/7 = 43$. $D = 43$
6. 82 = M**7 mod 341 and by inputting this into a for loop in python will we receive M = 103 and gives us the equation 82 =103**7 mod 341 that means M = 103 and then we can translate the numbers into letters and then we get M = BAD.

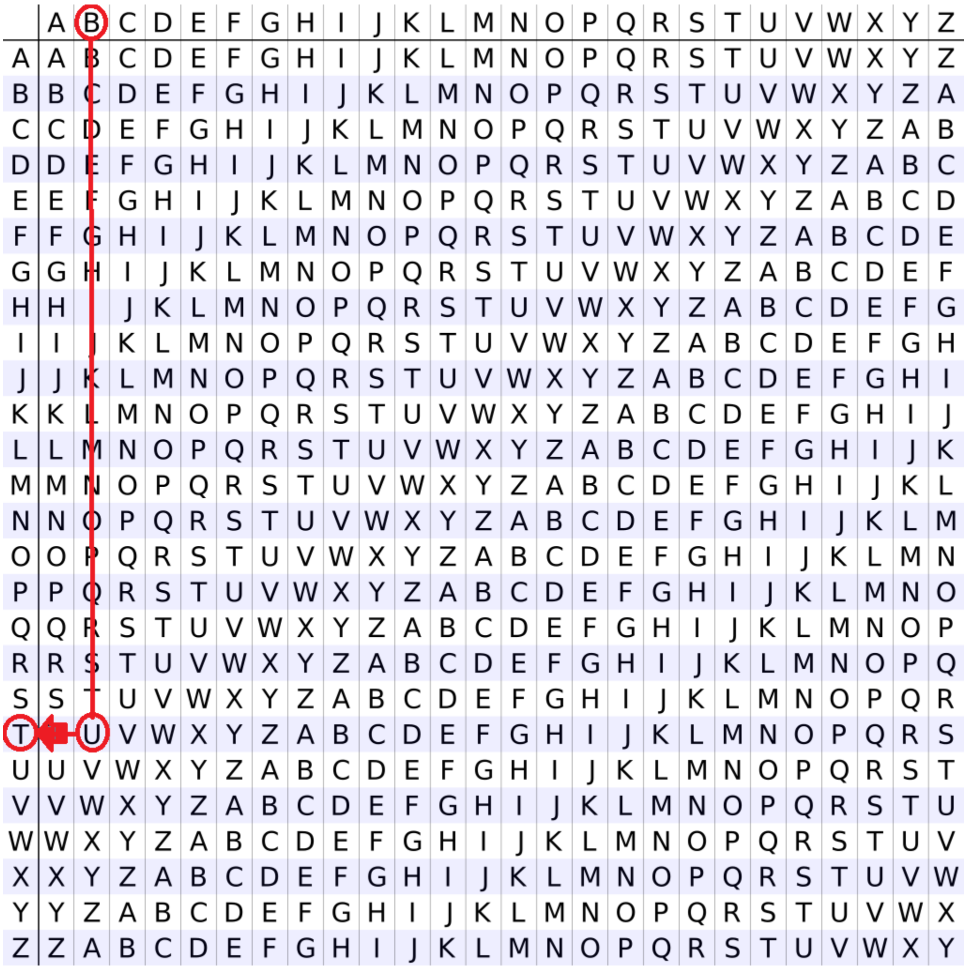


```
for n in range(1000):
    if n**7 % 341 == 82:
        print(n)
        break
```

```
PS C:\Users\olavo\OneDrive - University of Bergen\INF100> & C:/Users/olavo/AppDat
103
PS C:\Users\olavo\OneDrive - University of Bergen\INF100> olav høysæther opheim
```

Figur 12:Python calculation (Opheim, 2021)

Then by doing what we did in question 4 when we were decrypting the Vigenère cipher we do the same here. So, if we start by finding the letter “B” at the top row and once “B” is found we then follow that column down until we find the first letter of the ciphertext “U” and then follow that row to the left until we reach the end, and the last letter “T” is the first letter of the original message. Then we do this process for every letter until we receive the original message.



	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
A	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
B	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	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	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
I	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
J	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	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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

(C = Ciphertext, K = Key, M = Original Message)

C:

uhhgiyfmrtthgfldihotfzbhsdhgeqeeutaufaquifjpduiroegvcduirodhuefuirorhbcwjoqbngsevjllfnff drsepfmefrfzbhsshduujtbjspvcknouftkbndoiwuosjc

K:

BADBADBADBADBADBADBADBADBADBADBADBADBADBADBADBADBADBADBA DBADBADBADBADBADBADBADBADBADBADBADBADBADBADBADBADBADBADB ADBADBADBADBADBADBADBADBADBA

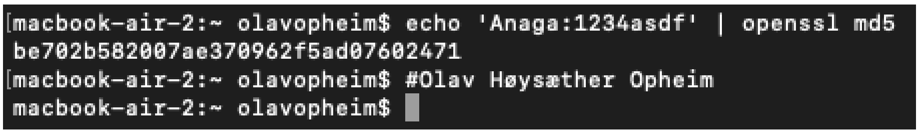
M:

THEFIVEMOSTEFFICIENTCYBERDEFENDERSAREANTICIPATIONEDUCATIONDE TECTIONREACTIONANDRESILIENCEDOREMEMBERCYBERSECURITYISMUCHM ORETHANANITTOPIC

User Authentication

Question 8.

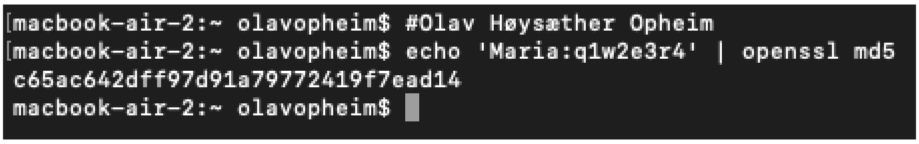
1. We are asked to use OpenSSL to calculate the md5 of these three usernames and passwords, to do this we write echo “username:password” and then openssl md5. This then gives us the md5 for these usernames and passwords:
    a. Anaga:1234asdf



```
[macbook-air-2:~ olavopheim$ echo 'Anaga:1234asdf' | openssl md5
be702b582007ae370962f5ad07602471
[macbook-air-2:~ olavopheim$ #Olav Høysæther Opheim
macbook-air-2:~ olavopheim$ █
```

Line two gives us the answer

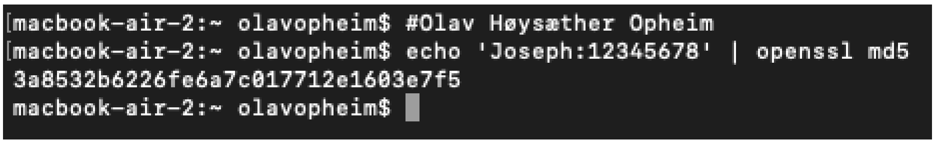
    b. Maria:q1w2e3r4



```
[macbook-air-2:~ olavopheim$ #Olav Høysæther Opheim
[macbook-air-2:~ olavopheim$ echo 'Maria:q1w2e3r4' | openssl md5
c65ac642dff97d91a79772419f7ead14
macbook-air-2:~ olavopheim$ █
```

Line three gives us the answer

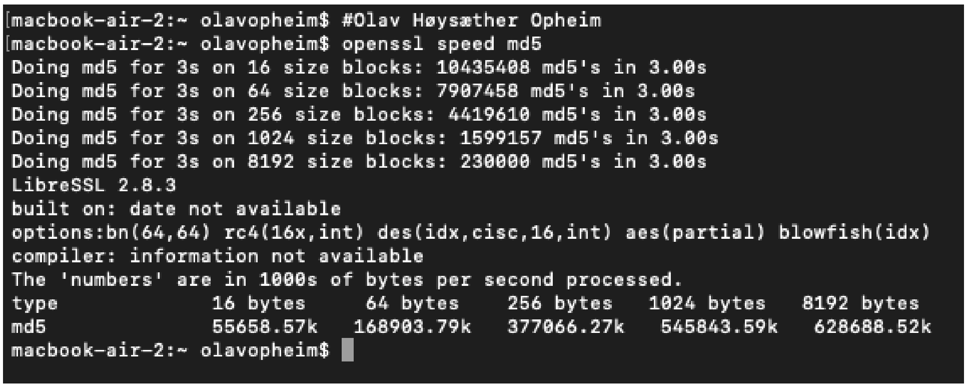
    c. Joseph:12345678



```
[macbook-air-2:~ olavopheim$ #Olav Høysæther Opheim
[macbook-air-2:~ olavopheim$ echo 'Joseph:12345678' | openssl md5
3a8532b6226fe6a7c017712e1603e7f5
macbook-air-2:~ olavopheim$ █
```

Line three gives us the awsner

2. In this task I am asked to use OpenSSL to test the speed of MD5 on my computer, and here is the result:



```
[macbook-air-2:~ olavopheim$ #Olav Høysæther Opheim
[macbook-air-2:~ olavopheim$ openssl speed md5
 Doing md5 for 3s on 16 size blocks: 10435408 md5's in 3.00s
 Doing md5 for 3s on 64 size blocks: 7907458 md5's in 3.00s
 Doing md5 for 3s on 256 size blocks: 4419610 md5's in 3.00s
 Doing md5 for 3s on 1024 size blocks: 1599157 md5's in 3.00s
 Doing md5 for 3s on 8192 size blocks: 230000 md5's in 3.00s
 LibreSSL 2.8.3
 built on: date not available
 options:bn(64,64) rc4(16x,int) des(idx,cisc,16,int) aes(partial) blowfish(idx)
 compiler: information not available
 The 'numbers' are in 1000s of bytes per second processed.
 type             16 bytes     64 bytes    256 bytes   1024 bytes   8192 bytes
 md5              55658.57k   168903.79k   377066.27k   545843.59k   628688.52k
 macbook-air-2:~ olavopheim$
```

By looking at this image we can see that my computer can manage to do 10435408 MD5 in 3 seconds. Let's take 10435408 and divide it by 3 to get 3478469,33 MD65 in 1 second. Now that we know what md5 speed of the computer we can procced answer a, b and c to give an estimate on how long time I need to crack Nikolay's passwords.

   a. Nikolay chooses a password of length 4 with all digits

      The equation for the estimation for how long it would take to crack a password with the length 4 with all digits is (We take 10 because its 10 digits: 0, 1, 2…9 and we take ^4 because its 4 in length): 10^4/3478469,3 = 0.0029 seconds to crack the password.

   b. Nikolay chooses a password of length 8 with all digits

      The equation for the estimation for how long it would take to crack a password with the length 8 with all digits is (we take 10 because its ten digits: 0, 1, 2…9 and we take ^8 because the length is 8): 10^8/3478469,3 = 29 seconds to crack the password.

   c. Nikolay chooses a password of length 8, where each position either a digit or a lower-case letter

      The equation for the estimation for how long it would take to crack a password with the length 8 where each position is either a digit or lower-case letter (we take 36 because its 10 digets: 0, 1, 2 … 9 and 26 letters in the alphabet so 10 + 26 = 36, we take ^8 because the length is 8):

36^8/3478469,3 = 811020,5 seconds to crack the password, and

that is 225,3 hours

Question 9.

Why is it considered that the additional salt in the UNIX password scheme increases security?

To be able to answer this question we first need to understand how creating a password works and the process. So, let’s say a user goes on a website and decides to create an account. The user then needs to input the required information, but also create a password. Once the password is created it gets saved as a hash in a storage system. If the user wants to, he needs to input the required username and password. Then the system checks if the password matches any other password hashes on the storage system, and if it does the user is granted access, and if not, the user is not granted access.

The problem with these hash tables is that they are often designed to be fast, and that often means security isn't the best. An example is LinkedIn who was a victim of a massive data breach in 2012. The reason was that they stored their passwords with a hash algorithm, but without salt or any other security measures. This meant that their plain hashes were able to be cracked within minutes which lead to the massive data breach.

A way to fix this is to implement salt or other security measures, but we will look at salt and how that provides increased security to the UNIX password scheme. So, the problem not using salt is that a password like “dog” used by person a will give the same hash value as person b who also uses the password “dog”. The security risk with this is that an attacker can use a pre-computed table to cross r stolen hashed passwords and manage to determine the password by using a reverse lookup method.

Salt on the other hand will increase the security to a UNIX password scheme because it adds random letters and numbers to the original password before it gets hashed. Let's go back to the example where person a uses “dog” as a password and before it gets hashed a random letter and number combination gets added “A3M5F” so that gives us the text “dogA3M5F” before it gets hashed. Person b also uses “dog” as a password, and again a random combination of letters and numbers gets added “M5D67”, which gives us the text “dogM5D67”. Once these passwords are hashed, they will receive different hash values and makes it impossible for attackers to use the reverse lookup method to figure out the original password. This means that matching passwords are store with different hash values, and that makes it harder for attackers to figure it out. (McAfee Cloud BU, 2016)

Question 10.

In this task we were given a password, and told to explain each segment of the password:

**root:$6$Q8uKtWWm/dptau2a$E184j/HJuiuw2lsUT7yuBvTh3FioWj5KKUvPQT
/1OJT4rtBACAm4NlEFV4n4x6ndTN3wD9A5uHOjEQQ/JJqN./:18142:0:99999:7:::**

Segments

First: root

**Second: $6$Q8uKtWWm/dptau2a$E184j/HJuiuw2lsUT7yuBvTh3FioWj5KKUvPQT
/1OJT4rtBACAm4NlEFV4n4x6ndTN3wD9A5uHOjEQQ/JJqN./**

Third: 18142

Fourth: 0

Fifth: 99999

Sixth: 7

Seventh: ::: (three blank fields)

Explanation

First segment shows us the username of the account which here is root. **Second** segment is the encrypted password. And the segment displayed between the first and second segment **"$6$"** shows us the algorithm used on Linux, and here we can see that SHA-512 is used. **Third** segment shows days since last time the password was changed which is 18 142 days. **Fourth** segment displays how many days a person needs to wait to change password. In this case we can see that the required days to wait is 0 days. **Fifth** segment shows how many days a password is valid before it needs to be changed. In this case we see that the password is valid in 99 999 days. **Sixth** segment shows how many days before the password expires that they will be warned, and in this case 7 days. **Seventh** segment shows the blank fields. The first blank field shows days before the account becomes inactive, second blank field shows days before the account password expires and the third blank field is often not used and just left blank. (Gite, 2021)

Sources:

- ASCII. (Unknow) ASCII. Avalible at: https://www.ascii-code.com/ (Accessed: 26. September 2021)
- Fields, B.T. (2011) The Vigenère square. Obtained from: https://en.wikipedia.org/wiki/Vigen%C3%A8re_cipher#/media/File:Vigen%C3%A8re_square_shading.svg (Downloaded 20 September 2021)
- Gite, V. (2021) Understanding /etc/shadow file format on Linux. Available at: https://www.cyberciti.biz/faq/understanding-etcshadow-file/ (Accessed: 21. September 2021)
- Li, C. (2021) Cryptographic Tools and Algorithms. Available at: https://mitt.uib.no/courses/29694/files/folder/LectureSlides?preview=3578405 (Accessed: 21. September 2021)
- McAfee Cloud BU (2016) What is Salt and How does It Make Password Hashing More Secure? Available at: https://www.mcafee.com/blogs/enterprise/cloud-security/what-is-a-salt-and-how-does-it-make-password-hashing-more-secure/ (Accessed: 21. September 2021)
- Opheim, O. (2021) Figure 7-11. Available at: Zip File (Accessed: 21 September 2021)
- Opheim, O. (2021) Float chart. Obtained from: screenshot (Downloaded 21 September 2021)
- Stallings, W. and Brown, L. (2017) *Computer Security: Principles and Practices.* 4th edn. New York: Pearson