Second Assignment

Mariano D'Angelo

4th April 2022

Task 1 — Big O Notation

Prove that $O(n^2) = an^2 + bn - c$, where a - is first 2 digits of your student code, b - is 3rd and 4th digit of your student code, c - last two digits of your student code.

Code 201752IVSB, then $a = 20$, $b = 17$, $c = 52$,
equation is $O(n^2) = 20n^2 + 17n - 52$

From theory we know that $f(n) \leq c \cdot g(n)$, so we get:

$20n^2 + 17n - 52 \leq c \cdot n^2$

To simplify our equation we can choose that $c = 21$. Now let's solve this:

$20n^2 + 17n - 52 \leq 21n^2$
$20n^2 - 21n^2 + 17n - 52 \leq 0$
$-n^2 + 17n - 52 \leq 0 \mid \cdot -1$
$n^2 - 17n + 52 \geq 0$

By taking the derivative of the quadratic formula we get:

$2n - 17 \geq 0$
$2n \geq 17$
$n = 8.5$

Meaning that the function starts to grow again after the value 8.5 is encountered.

By solving this quadratic formula equation we get the following values:

$n_1 = 13, n_2 = 4$, so $n < 4 \;\; V \;\; n > 13$

Meaning that the function is positive only when bigger than 4 or 13.

As a result we get — $c = 21, n = 14$

Task 2 — Complexity theory

Give an example of a *search* problem and corresponding *decision* problem, which was not discussed in the lectures.

Is $x \in \mathbb{Z}$ positive or negative?

Consider the corresponding verification function $V(x, y)$:

$$\begin{cases} 1 \text{ where } x = y + z \text{ and } (y \geq 0 \text{ and } z \geq 0) \text{ or } (+y > -|z| \text{ and viceversa}) \\ 0 \text{ if otherwise} \end{cases}$$

Search Problem (Summing): Given a positive x, find y such that $V(x, y) = 1$.

Decision Problem (Positiveness/Negativeness): Given x, decide if there is y such that $V(x, y) = 1$.

Task 3 — Block ciphers

Assume you are the agent of Mission Impossible. Top-level agents have decided that your agency will use AES-128 block cipher for its missions. You are given a task to choose a suitable encryption mode for the following mission scenarios:

1. Encryption of the agent's 6 digit identification number stored in the Super Secret Database (SSD)
2. Encryption of a document that will be sent via email.

Please, motivate your answer.

1. In order to encrypt a 6 digit identification number I would use ECB. Even though it is not that safe when it comes to the encryption of multiple blocks (lack of diffusion), it can do well when it comes to the encryption of a single block. If we think of the 6 digit number as an integer it only occupies 4 bytes and we are encrypting with an 128 bit AES. Besides the lack of diffusion it has many positive sides, such as parallelizable encryption and decryption, and the possibility of random read.
2. In order to encrypt a document I would use CTR. This mode of encryption has many benefits, both encryption and decryption are parallelizable, random read is possible, faulty blocks affect only their current block. CTR uses a counter that is encrypted with the cipher text and increases with each block, so repeating ciphertext will not appear (good diffusion). No padding is used in the CTR mode (ciphertext is the same length as the plaintext) avoiding the padding oracle attack.

Task 4 — Modes of encryption

Alice wants to send a message m, encrypted with block cipher, to Bob. The message m is split into 4 blocks of equal length $m = m_0||m_1||m_2||m_3$ and encrypted using 3DES. However, a transmission error occurs (or malicious Carol got access to the channel) and one bit of the ciphertext c_1 changes its value. Bob receives the ciphertexts, decrypts them and gets the following message $m' = m'_0||m'_1||m'_2||m'_3$. Please, explain to Bob how many bits (approximately) are expected to be wrong in each block m'_i if Alice used ECB and CBC modes.

3DES encrypts in blocks of 64 bits. So let's suppose each block contains about 64 bits of information.
If Alice used ECB, the whole block in which the error resuted (m_1), would become faulty, so we would lose approximately 64 bits of data. ECB divides the plaintext into blocks and every block is encrypted with the same key and algorithm. Because no parts of the faulty block are used in the other blocks, the error is isolated.
If Alice used CBC, the error would be a bit worse than the one for ECB. This is because the resulting ciphertext of the current block is used to encrypt the next block (resulting ciphertext is XORed with next plaintext). If we get the error in m_1 there will be one fully faulty block (current) and one with a wrong bit (the next one). So we will have approximately 64 + 1 bits that are expected to be wrong.

Task 5 — Diffie-Hellman

Consider the Diffie-Hellman protocol with $\alpha = 3$ and $p = 673$. Alice chooses $A = 95$ and Bob chooses $B = 240$. Compute the messages that Alice and Bob send to each other and the final shared key.

The Diffie-Hellman key establishment can be divided in 5 steps:

1. Alice and Bob choose a private key, $\omega_A \leftarrow \{1, ..., p-1\}$ for Alice and $\omega_B \leftarrow \{1, ..., p-1\}$ for Bob
2. Alice computes its public key and sends it to Bob, computes $y_B = \alpha^{\omega_A} mod\ p$ and sends $m_A^1 = y_A$
3. Bob computes its public key and sends it to Alice, computes $y_B = \alpha^{\omega_B} mod\ p$ and sends $m_B^1 = y_B$
4. Alice computes the shared key, $k_A = y_B^{\omega_A} mod\ p$
5. Bob recomputes the shared key, $k_B = y_A^{\omega_B} mod\ p$ and checks with Alice if the shared keys are the same

In order to accomplish this task, I wrote a simple script located in APPENDIX A.

In the beginning I declared 4 variables, which are: Alice's and Bob's shared secret or α (3), Alice's private key or ω_A(95), Bob's private key or ω_B (240), and a large prime or p (673). Once the variables are declared I input them in the function *establish_key*, which will calculate Alice's and Bob's shared key. Firstly, it calculates Alice's public key or y_A using the formula in point number 2. Secondly, it caluclates Bob's public key or y_B using the formula in point number 3. Finally, it calculates the shared key, first for Alice, using the formula in point number 4, and thne for Bob, using the formula in point number 5. In order to check if the keys are the same, the program compares the two calculated shared keys and if are the same print the message "The key has been successfully established! The key is ...", but if the keys are not equal, then the program exits with the following message "An error has occured!".

Task 6 — Group theory

Do the following sets with defined operations form a group? Provide explanation.

- $\mathbb{Z}_{10}^{+}$ (*Integers modulo 10 under addition*) — This is group, because it has all of the three attributes required: it associative, e.g. 3 + (1 + 2) = (3 + 1) + 2, has a unit which is 0, e.g. 7 + 0 = 0 + 7 = 7, and is invertible, we can get by summing a pair of numbers 10, e.g. 8 + 2, 9 + 1, 6 + 4, taken mod 10 will be 0.
- $\mathbb{Q}$ (*Rational numbers under multiplication*) — This is not a group, since it is for all rational numbers, 0 too and 0 has not an inverse.
- $\mathbb{Z}_{37}^{-}$ (*Intergers under subtractions*) — This is not a group, because it is not associative, e.g. 10 - (5 - 3) $\neq$ 10 - 5 - 3
- $(\mathbb{C} - 0, *)$ (*Complex numbers under multiplication*) — This is a group, because it is associative (multiplication is associative), it has a unit, which is 1 (every number multiplied by 1 will remain the same), and is inverse (only without 0).

APPENDIX A - task 5 program

```
def establish_key(shared_secret, a_priv, b_priv, p):
    a_pub = shared_secret**a_priv%p
    b_pub = shared_secret**b_priv%p
    a_key = b_pub**a_priv%p
    b_key = a_pub**b_priv%p
    if a_key == b_key:
        print("The key has been successfully established!")
        print(f"The key is: {a_key}")
    else:
        exit("An error has occured!\n")


shared_secret, alice_priv, bob_priv, p = 3, 95, 240, 673
establish_key(shared_secret, alice_priv, bob_priv, p)
```