# CS3626 Homework 02 Fall 2024

# Classical Encryption, Symmetric Encryption

**Total Points: 25**

Be as brief as possible and use your own words when describing concepts.   SHOW ALL WORK for Questions requiring calculations and algorithms.

**Q-1: There are two requirements for the secure use of a symmetric encryption algorithm? Describe them.**

1. Strong encryption algorithm
2. The sender and receiver must have a secure key.

**2 points**

**Q-2: How is a brute force attack conducted?**

A brute force basically goes through all possible combinations until a result it found.

**1 point**

**Q-3: How is a cryptanalytic attack performed? How does it differ from a brute force attack?**

A cryptanalysis attack is performed when an attacker has all necessary data before performing an attack, such as an encryption key. This is different from a brute force attack because an attacker doesn’t need data from the specific entities they are trying to attack.

**2 points**

**Q-4: Define Computationally Secure.**

Computationally secure means that something is impossible to make insecure by using a computer. Such as a brute forcing a 20-character long password that would take 304844284494 years to obtain.

**1 point**

**Q-5: Define Unconditionally Secure.**

Unconditional security means something can’t be obtained because an attacker doesn’t have the necessary information to attack it.

**Name an encryption scheme that is unconditionally secure: \_\_\_One time pad\_\_\_**

**2 points**

**Q-6: Describe the two concerns with using a One Time Pad.**

1. **Secret key must have an equal size of the message being sent**
2. **Key must be truly random.**

**2 points**

**Q-7: Construct the playfair Matrix using the keyword “CYPHER”:**

**MATRIX:**

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

**Use this matrix to encrypt this message: “HALLUX”**

**Rules and order of operations:**

1. **Double letter digraphs except “XX” get an “X” inserted between them.**
2. **Double X digraphs i.e. “XX” become “XYX”.**
3. **Unpaired digraphs except “X” get padded with “X”.**
4. **Unpaired letter “X” gets padded with “Y”**

**SHOW all DIGRAPH maps e.g. AR -> QC:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Cipher Text** | **HA** | **LX** | **LU** | **XY** |
| **Encryption** | **YD** | **SH** | **GX** | **VH** |

**Encrypted Message: \_\_\_YDSHGXVH\_\_\_\_\_\_\_\_**

**Decrypt the Encryption Message above, what is the result: \_\_\_\_HALXLUXY\_=> HALLUX\_\_\_\_\_\_\_\_\_\_\_\_**

**3 points**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Cipher Text** | **YD** | **SH** | **GX** | **VH** |
| **Decryption** | **HA** | **LX** | **LU** | **XY** |

**The following two questions are support functions for future assignment/project work and if well written can be recycled for use in the project and future assignments will build on these functions. Keep the source code.**

**Q-8: Part 1, implement a proper modulus function, one that satisfies a = q\*b + r | 0 <= r < b. It should accept any integer for a, b and return the values, also integers, for r and q. Test it and compare to the values given by your chosen languages native modulus operator (usually a%b in most languages).**

**(gnu c++ v11 % operator) (proper modulus)**

**a b q r q r**

**Example: -1011 11 -91 -10 -92 1**

**Example: -1011 -11 -91 -10 92 1**

**Example: -1011 0 Floating Point Crash( not *correct*) Prints Error (function is *correct*)**

**Results do not need to be provided, just need to verify it works and propery and is *correct* for 0 inputs as well. You decide what *correct* means (return 0 for q/r or print an error message or whatever else so long as it does not terminate unexpectedly or behave unpredictably etc… i.e. is not *correct*)**

**Part 2, use the function above to implement the Caesar algorithm to encrypt and decrypt messages and also to perform a brute force attack to determine the key. Use it for this encrypted text and provide the plaintext and key below for this message. The encryption scheme is known to be the Caesar monoalphabetic cypher using the alphabet below:**

Alphabet = abcdefghijklmnopqrstuvwxyz where a = 0, b = 1, c = 2 etc…

Cipher Text = wvmbqumxil Brute force attack to find the key.

Plain Text: \_\_\_\_\_\_\_\_\_\_\_\_\_ onetimepad \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ KEY: \_\_\_18\_\_\_\_\_\_\_\_\_\_

Provide source-code per instructions below. Keep this code it will be revisited in future assignments and in the class activity.

**5 points**

**Q-9: Using proper modulus above, implement the following:**

1. **Euclidean Algorithm**, determine d, the greatest common divisor gcd(a,b) where a and b are integers. It must accept positive and negative numbers and be *correct* for zero inputs and for values where |a| < |b|. Recall gcd(a,b) == to gcd( |a|, |b|).
2. **Extended Euclidean Algorithm,** given integers a, b: determine d, s and t where a\*t + b\*s = d = gcd(a,b). Your function should return the values of d, s, t (you decide how, call by reference or in a structure or some other way). Your function should be able to handle, b = 0 inputs, negative values, and also inputs where |a| < |b|. Recall that if |a| < |b| the modular inverse of a mod b is no longer found from t = 1,0,1 it will move to s = 0,1,-q. Ensure that the value t is always the modular inverse to **input** a mod (the a,b value prior to swapping if needed) and s is always the modular inverse to the **input** b mod a (these values prior to swapping). In practice this means the calculated value s will be returned by the function in t (and t will be in s) if the original inputs a, b satisfy |a| < |b|. This is so that the caller that provides a,b does not have to be aware that they must use s instead of t if |a| < |b| and the caller get the expected results a\*t mod b == 1 mod b and b\*s mod a == 1 even if they provided |a| < |b| in their inputs. It will help you later in the semester to return like this.

A b d a mod b Modular Inv. b mod a Modular Inv.

Example: 283 35 1 12 -97 a mod b from t(1,0,1)

Example 35 283 1 -97 (186) 12 b mod a from s(0,1,-q)

Provide source-code per instructions below. Keep this code, it will be revisited in future assignments and in the Feistel project.

**7 points**

**Submission Guidelines:**

* No handwritten submission is accepted, always submit answers as text within this or similar document file with any support images embedded in the file.
* **EXCEPTION**: If asked for source code implementation you can submit those individually and as separate files in ASCII format in their original file format .cpp, .java, .py, .cs etc. or even as a .txt file will be acceptable. Do not insert code into the submission document file. It ruins spacing which makes .python and some languages (perl, awk etc.) difficult to test build.
* Do not submit ZIP files… ever… for anything in D2L. The system is extremely unhelpful with regards to those filetypes and grading.
* You may include your freehand drawing/image and handwritten scans in the submission. However, the writing and images must be clearly legible. Though, it is best to present non-handwritten submissions, generally, as is done in the professional setting.
* If asked, show all work/calculations/graphs etc. in the determination of the problem.
* **Please complete your entire work in a single Word Document and Save the file as: yournetid\_CS3502\_Assignment01.docx (e.g. ogarcia5\_CS3502\_Assignment01.docx.) and upload your file in D2L.**
* Please observe the submission due date and time. After the due date there is a 50% penalty for the next 24 hours. Any submission after 24 hours of the due date will be graded at 0%.
* If you include a reference or an image taken from other sources, please cite them appropriately. APA is preferred but cite them so they can be found. **NOTE: verbatim copying or even paraphrasing is plagiarism so if the source used constitutes your answer rather than simply *supporting* the answer, it will be considered invalid. This is especially true of source code implementation answers.**
* If you resubmit, please make sure to attach the file again. Your latest submission before the due date will be the one graded.