CS 6324: Information Security **Spring 2021**

Homework #1

Due date & time: 11:59pm CST on February 12, 2021. Submit to eLearning by the due time.

Additional Instructions: The submitted homework must be typed. Using LaTeX is recommended, but not required.

Problem 1 (10 pts) Confidentiality, Integrity, Availability.

- (3 pts) State what is Confidentiality, Integrity, and Availability.
  - Confidentiality: The property, that information is not made available or disclosed to unauthorized individuals, entities, or processes.
  - Integrity: Maintaining and assuring the accuracy and completeness of data over its entire lifecycle.
  - Availability: The computing systems used to store and process the information, the security controls used to protect it, and the communication channels used to access it must be functioning correctly.
- (3 pts) For each, give two examples where they are violated.
  - Confidentiality:
    * 1. Anthem Data Leakage: personal data of 80 million people were stolen, accessible by unknown attackers.
    * 2. Confidential information sent to elsewhere after opening links attached in phishing email, which compromised the OS.
  - Integrity:
    * 1. Ransom-ware encrypt data on the computer and ask for money payments for decryption.
    * 2. Stuxnet brought down software systems of the industry control system of the power plant.
  - Availability:
    * 1. Denial of service attack, shutting down a particular server on the network by sending excessive amount of requests.
    * 2. Stuxnet disable some of the key functionalities in the power plant so that they are inaccessible, allowing nuclear related incidents to happen.
- (4 pts) Identify two computer security control measures on your computer(s). Which of the three properties Confidentiality, Integrity, and Availability do they aim at providing? What kinds of adversaries they **cannot** defend against?
  - 1. BitLocker function for hard drive
    * It is aim at protecting the confidentiality of data so it cannot be accessed by anyone unauthorised.

    * It cannot defend against integrity threats such as low-level formatting. Once the drive is wiped, information on the hard drive is no longer available.
  - 2. Tamper protection
    * It is aim at protecting the integrity of data, preventing any software from changing key files of the operating system.
    * It cannot defend against confidentiality threats as users' data may still be stolen without modifying key files of the OS. Also, availability is not guaranteed as denial of service attack arrives since blocking accessiblity does not require tampering key OS files.

Problem 2 (30 pts) Probability Review.

1. (10 pts) We roll two fair 6-sided dice.
   (a) Find the probability that doubles are rolled.
   -

	1	2	3	4	5	6
1	(1,1)	(1,2)	(1,3)	(1,4)	(1,5)	(1,6)
2	(2,1)	(2,2)	(2,3)	(2,4)	(2,5)	(2,6)
3	(3,1)	(3,2)	(3,3)	(3,4)	(3,5)	(3,6)
4	(4,1)	(4,2)	(4,3)	(4,4)	(4,5)	(4,6)
5	(5,1)	(5,2)	(5,3)	(5,4)	(5,5)	(5,6)
6	(6,1)	(6,2)	(6,3)	(6,4)	(6,5)	(6,6)

   - Rolling doubles: (1,1), (2,2), etc...
     Total possible rolls: $6 \times 6 = 36$
     Doubles Possible: 6
     $\mathbf{Pr}(doubles) = \frac{6}{36} = \frac{1}{6}$

   (b) Given that the roll results in a sum of 6 or less, find the conditional probability that doubles are rolled.
   - Rolls with sum of 6 or less: 5 + 4 + 3 + 2 + 1 = 15
     $\mathbf{Pr}$(sum of 6 or less) = $\frac{15}{36} = \frac{5}{12}$
     Rolls of double with sum of 6 or less: 3
     $\mathbf{Pr}$(doubles $\wedge$ 6 or less) = $\mathbf{Pr}$(6 or less) * $\mathbf{Pr}$(doubles | 6 or less)
     $\mathbf{Pr}$(doubles | 6 or less) = $\mathbf{Pr}$(doubles $\wedge$ 6 or less) / $\mathbf{Pr}$(6 or less)
     $\mathbf{Pr}$(doubles | 6 or less) = $\frac{3}{36}/\frac{5}{12} = \frac{1}{5}$

   (c) Find the probability that that larger of the two die's outcomes is at least 4.
   - Rolls that only 1 die is at least 4 : 9 + 9 = 18
     Rolls that 2 dies are at least 4: 9
     $\mathbf{Pr}$(larger of the 2 die's outcome is at least 4) = $\frac{27}{36} = \frac{3}{4}$

   (d) Given that the two dice land on different numbers, find the conditional probability that at least one die roll is a 1.
   - $\mathbf{Pr}$(2 dices land on differnt numbers) = 1 - $\mathbf{Pr}(doubles) = \frac{5}{6}$
     Rolls that at least one of them is 1: 11
     Since two dice land on differnet numbers, (1,1) is excluded.
     $\mathbf{Pr}$(at least one die roll is 1 $\wedge$ 2 dice land on different numbers) = $\frac{11-1}{36} = \frac{10}{36} = \frac{5}{18}$
     $\mathbf{Pr}$(at least one die roll is 1 | 2 dice land on different numbers)

= $\mathbf{Pr}$(at least one die roll is 1 $\wedge$ 2 dice land on different numbers) / $\mathbf{Pr}$(2 dices land on differnt numbers)
= $\frac{5}{18}/\frac{5}{6} = \frac{1}{3}$

(e) Let X be the event that the first die results in an odd number, and Y be the event that the rolled sum is an even number. Compute $\mathbf{Pr}[X], \mathbf{Pr}[Y], \mathbf{Pr}[X \wedge Y]$. Are X and Y independent?

- If $\mathbf{Pr}[X \wedge Y] = \mathbf{Pr}[X]$ * $\mathbf{Pr}[Y]$, X and Y are independent.
  Odd + Odd = Even; Even + Even = Even; Odd + Even = Odd.
  Rolls that first die results in an odd number: 6 + 6 + 6 = 18
  $\mathbf{Pr}[X] = \frac{18}{36} = \frac{1}{2}$
  Rolls that the rolled sum is even = 3 + 3 + 3 + 3 + 3 + 3 = 18
  $\mathbf{Pr}[Y] = \frac{18}{36} = \frac{1}{2}$
  Rolls that first die is odd and that sum is even: 3 + 3 + 3 = 9
  $\mathbf{Pr}[X \wedge Y] = \frac{9}{36} = \frac{1}{4}$
  Since $\mathbf{Pr}[X]$ * $\mathbf{Pr}[Y]$ = $\frac{1}{2} * \frac{1}{2} = \frac{1}{4}$, events X and Y are independent.

(f) Let X be the event that the first die results in a multiple of 2, and Y be the event that the rolled sum is a multiple of 2. Compute $\mathbf{Pr}[X], \mathbf{Pr}[Y], \mathbf{Pr}[X \wedge Y]$. Are X are Y independent?

- If $\mathbf{Pr}[X \wedge Y] = \mathbf{Pr}[X]$ * $\mathbf{Pr}[Y]$, X and Y are independent.
  Odd + Odd = Even; Even + Even = Even; Odd + Even = Odd.
  Rolls that the first die is a multiple of 2: 6 + 6 + 6 = 18
  $\mathbf{Pr}[X] = \frac{18}{36} = \frac{1}{2}$
  Rolls that the rolled sum is a multiple of 2: 3 + 3 + 3 + 3 + 3 + 3 = 18
  $\mathbf{Pr}[Y] = \frac{18}{36} = \frac{1}{2}$
  Rolls that first die is a multiple of 2 and that sum is a multiple of 2: 3 + 3 + 3 = 9
  $\mathbf{Pr}[X \wedge Y] = \frac{9}{36} = \frac{1}{4}$
  Since $\mathbf{Pr}[X]$ * $\mathbf{Pr}[Y]$ = $\frac{1}{2} * \frac{1}{2} = \frac{1}{4}$, events X and Y are independent.

(g) Let X be the event that the first die results in a multiple of 3, and Y be the event that the rolled sum is a multiple of 2. Compute $\mathbf{Pr}[X|Y]$ and $\mathbf{Pr}[Y|X]$. Are X are Y independent?

- If $\mathbf{Pr}[X \wedge Y] = \mathbf{Pr}[X]$ * $\mathbf{Pr}[Y]$, X and Y are independent.
  Odd + Odd = Even; Even + Even = Even; Odd + Even = Odd.
  Rolls that the first die is a multiple of 3: 6 + 6 = 12
  $\mathbf{Pr}[X] = \frac{12}{36} = \frac{1}{3}$
  Rolls that the rolled sum is a multiple of 2: 3 + 3 + 3 + 3 + 3 + 3 = 18
  $\mathbf{Pr}[Y] = \frac{18}{36} = \frac{1}{2}$
  Rolls that first die is a multiple of 3 and that sum is a multiple of 2: 3 + 3 = 6
  $\mathbf{Pr}[X \wedge Y] = \frac{6}{36} = \frac{1}{6}$
  Since $\mathbf{Pr}[X]$ * $\mathbf{Pr}[Y]$ = $\frac{1}{3} * \frac{1}{2} = \frac{1}{6}$, events X and Y are independent.

2. (5 pts) Let X denote the event that a student pass the midterm exam in a course. And Y denote the event that the student pass the final exam. We know that $\mathbf{Pr}[X] = 0.75$, $\mathbf{Pr}[Y] = 0.8$, and $\mathbf{Pr}[Y|X] = 0.9$. (The probability that a student will pass the final exam given that he or she has passed the midterm exam is 0.9.) Compute $\mathbf{Pr}[X \wedge Y]$, $\mathbf{Pr}[\neg X|Y]$, and $\mathbf{Pr}[Y|\neg X]$. Are X and Y independent?

- $\mathbf{Pr}[X \wedge Y] = \mathbf{Pr}[X] * \mathbf{Pr}[Y|X]$
  $\mathbf{Pr}[X \wedge Y] = 0.75 * 0.9 = 0.675$

- $\mathbf{Pr}[Y] = \mathbf{Pr}[X \wedge Y] + \mathbf{Pr}[\neg X \wedge Y]$
  $\mathbf{Pr}[Y] = \mathbf{Pr}[X \wedge Y] + \mathbf{Pr}[Y] * \mathbf{Pr}[\neg X|Y]$
  $1 - \frac{\mathbf{Pr}[X \wedge Y]}{\mathbf{Pr}[Y]} = \mathbf{Pr}[\neg X|Y]$
  $\mathbf{Pr}[\neg X|Y] = \frac{\mathbf{Pr}[Y] - \mathbf{Pr}[X \wedge Y]}{\mathbf{Pr}[Y]}$
  $\mathbf{Pr}[\neg X|Y] = \frac{0.8 - 0.675}{0.8}$
  $\mathbf{Pr}[\neg X|Y] = \frac{0.125}{0.8}$
  $\mathbf{Pr}[\neg X|Y] = \frac{5}{32}$

- $\mathbf{Pr}[Y] = \mathbf{Pr}[X \wedge Y] + \mathbf{Pr}[\neg X \wedge Y]$
  $\mathbf{Pr}[Y] = \mathbf{Pr}[X \wedge Y] + \mathbf{Pr}[\neg X] * \mathbf{Pr}[Y|\neg X]$
  $\frac{\mathbf{Pr}[Y]}{\mathbf{Pr}[\neg X]} = \frac{\mathbf{Pr}[X \wedge Y]}{\mathbf{Pr}[\neg X]} + \mathbf{Pr}[Y|\neg X]$
  $\mathbf{Pr}[Y|\neg X] = \frac{\mathbf{Pr}[Y] - \mathbf{Pr}[X \wedge Y]}{\mathbf{Pr}[\neg X]}$
  $\mathbf{Pr}[Y|\neg X] = \frac{\mathbf{Pr}[Y] - \mathbf{Pr}[X \wedge Y]}{1 - \mathbf{Pr}[X]}$
  $\mathbf{Pr}[Y|\neg X] = \frac{0.8 - 0.675}{1 - 0.75}$
  $\mathbf{Pr}[Y|\neg X] = \frac{0.125}{0.25}$
  $\mathbf{Pr}[Y|\neg X] = \frac{1}{2}$

- Since $\mathbf{Pr}[X] * \mathbf{Pr}[Y] = 0.75 * 0.8 = 0.6$, $\mathbf{Pr}[X \wedge Y] = 0.75 * 0.9 = 0.675$
  $\mathbf{Pr}[X \wedge Y] \neq \mathbf{Pr}[X] * \mathbf{Pr}[Y]$
  Events X and Y are not independent.

3. (6 pts) There are 78 qualified applicants for teaching positions in an elementary school, of which some have at least five years' teaching experience and some have not, some are married and some are single, with the exact breakdown being

	Married	Single
At least five years teaching experience	18	12
Less than five years teaching experience	30	18

   (a) The order in which the applicants are interviewed is random. M is the event that the first applicant interviewed is married and F is the event that the first applicant interviewed has at least five years teaching experience. Find the following probabilities: $\mathbf{Pr}[M], \mathbf{Pr}[F], \mathbf{Pr}[M \wedge F], \mathbf{Pr}[M|F], \mathbf{Pr}[F|M]$. Are M and F independent?

   - $\mathbf{Pr}[M] = \frac{18+30}{78}$
     $\mathbf{Pr}[M] = \frac{48}{78}$
     $\mathbf{Pr}[M] = \frac{8}{13}$

   - $\mathbf{Pr}[F] = \frac{18+12}{78}$
     $\mathbf{Pr}[F] = \frac{30}{78}$
     $\mathbf{Pr}[F] = \frac{5}{13}$

- $\mathbf{Pr}[M|F] = \mathbf{Pr}[M \wedge F]/\mathbf{Pr}[F]$
  Since $\mathbf{Pr}[M \wedge F] = \frac{18}{78} = \frac{3}{13}$
  $\mathbf{Pr}[M|F] = \frac{3}{13}/\frac{5}{13}$
  $\mathbf{Pr}[M|F] = \frac{3}{5}$

- $\mathbf{Pr}[F|M] = \mathbf{Pr}[M \wedge F]/\mathbf{Pr}[M]$
  $\mathbf{Pr}[F|M] = \frac{3}{13}/\frac{8}{13}$
  $\mathbf{Pr}[F|M] = \frac{3}{8}$

- Since $\mathbf{Pr}[M] * \mathbf{Pr}[F] = \frac{40}{169}$, and that $\mathbf{Pr}[M \wedge F] = \frac{3}{13}$,
  $\mathbf{Pr}[M \wedge F] \neq \mathbf{Pr}[M] * \mathbf{Pr}[F]$
  Events M and F are not independent.

(b) Suppose that there is only one opening in the third grade, and each applicant with at least five year experience has *twice* the chance of an applicant with less than five year experience. Let U denote the event that the job goes to one of the single applicants, and V the event that it goes to an applicant with less than five year experience. Find the following probabilities: $\mathbf{Pr}[U], \mathbf{Pr}[V], \mathbf{Pr}[U \wedge V], \mathbf{Pr}[U|V], \mathbf{Pr}[V|U]$. Are U and V independent?

- As those who have more than 5 years of experience has twice the chance, meaning 2 of the experienced can beat one inexperienced.
  Ratio applied to applicants count is 2:1

	Married	Single	Total
greater than 5 years	$18 * \frac{2}{3} = 12$	$12 * \frac{2}{3} = 8$	20
less than 5 years	$30 * \frac{1}{3} = 10$	$18 * \frac{1}{3} = 6$	16
Total	22	14	36

- $\mathbf{Pr}[U] = \frac{8+6}{22+14}$
  $\mathbf{Pr}[U] = \frac{14}{36}$
  $\mathbf{Pr}[U] = \frac{7}{18}$

- $\mathbf{Pr}[V] = \frac{16}{20+16}$
  $\mathbf{Pr}[V] = \frac{16}{36}$
  $\mathbf{Pr}[V] = \frac{4}{9}$

- $\mathbf{Pr}[U \wedge V] = \frac{6}{36} = \frac{1}{6}$

- $\mathbf{Pr}[U|V] = \mathbf{Pr}[U \wedge V]/\mathbf{Pr}[V]$
  Since $\mathbf{Pr}[U \wedge V] = \frac{1}{6}$
  $\mathbf{Pr}[U|V] = \frac{1}{6}/\frac{4}{9}$
  $\mathbf{Pr}[U|V] = \frac{3}{8}$

- $\mathbf{Pr}[V|U] = \mathbf{Pr}[U \wedge V]/\mathbf{Pr}[U]$
  $\mathbf{Pr}[V|U] = \frac{1}{6}/\frac{7}{18}$

$\mathbf{Pr}[V|U] = \frac{3}{7}$

- Since $\mathbf{Pr}[U] * \mathbf{Pr}[V] = \frac{9}{56}$, and that $\mathbf{Pr}[U \wedge V] = \frac{1}{6}$,
$\mathbf{Pr}[U \wedge V] \neq \mathbf{Pr}[M] * \mathbf{Pr}[F]$
Events M and F are not independent.

4. (9 pts) A test for a certain rare disease is assumed to be correct 95% of the time. Let S denote the event that a person has the disease, and T denote the event that the test result is positive. We have $\mathbf{Pr}[T|S] = 0.95$ and $\mathbf{Pr}[\neg T|\neg S] = 0.95$. Assume that $\mathbf{Pr}[S] = 0.001$, i.e., the probability that a randomly drawn person has the disease is 0.001.

   (a) Compute $\mathbf{Pr}[S|T]$, the probability that one has the disease if one is tested positive.
   - $\mathbf{Pr}[T|S] = 0.95, \mathbf{Pr}[T|\neg S] = 1 - \mathbf{Pr}[\neg T|\neg S] = 1 - 0.95 = 0.05, \mathbf{Pr}[S] = 0.001, \mathbf{Pr}[\neg S] = 1 - 0.001 = 0.999$
   $\mathbf{Pr}[T] = \mathbf{Pr}[T \wedge S] + \mathbf{Pr}[T \wedge \neg S]$
   $\mathbf{Pr}[T] = \mathbf{Pr}[S] * \mathbf{Pr}[T|S] + \mathbf{Pr}[\neg S] * \mathbf{Pr}[T|\neg S]$
   $\mathbf{Pr}[T] = 0.001 * 0.95 + 0.999 * 0.05$
   $\mathbf{Pr}[T] = 0.00095 + 0.04995 = 0.0509$
   Now, $\mathbf{Pr}[S|T] = \frac{\mathbf{Pr}[T \wedge S]}{\mathbf{Pr}[T]}$
   $\mathbf{Pr}[S|T] = \frac{0.001 * 0.95}{0.0509} = \frac{19}{1018} \approx 0.1866$

   (b) Assume that one improves the test so that $\mathbf{Pr}[T|S] = 0.998$, i.e., if one has the disease, the test will come out positive with probability 0.998. Other things remain unchanged. Compute $\mathbf{Pr}[S|T]$.
   - $\mathbf{Pr}[T|S] = 0.998, \mathbf{Pr}[T|\neg S] = 1 - \mathbf{Pr}[\neg T|\neg S] = 1 - 0.95 = 0.05, \mathbf{Pr}[S] = 0.001, \mathbf{Pr}[\neg S] = 1 - 0.001 = 0.999$
   $\mathbf{Pr}[T] = \mathbf{Pr}[T \wedge S] + \mathbf{Pr}[T \wedge \neg S]$
   $\mathbf{Pr}[T] = \mathbf{Pr}[S] * \mathbf{Pr}[T|S] + \mathbf{Pr}[\neg S] * \mathbf{Pr}[T|\neg S]$
   $\mathbf{Pr}[T] = 0.001 * 0.998 + 0.999 * 0.05$
   $\mathbf{Pr}[T] = 0.000998 + 0.04995 = 0.050948$
   Now, $\mathbf{Pr}[S|T] = \frac{\mathbf{Pr}[T \wedge S]}{\mathbf{Pr}[T]}$
   $\mathbf{Pr}[S|T] = \frac{0.001 * 0.998}{0.050948} \approx 0.1959$

   (c) Assume that another improvement on the test improves $\mathbf{Pr}[\neg T|\neg S] = 0.998$, but have $\mathbf{Pr}[T|S]$ remain at 0.95. Compute $\mathbf{Pr}[S|T]$.
   - $\mathbf{Pr}[T|S] = 0.95, \mathbf{Pr}[T|\neg S] = 1 - \mathbf{Pr}[\neg T|\neg S] = 1 - 0.998 = 0.002, \mathbf{Pr}[S] = 0.001, \mathbf{Pr}[\neg S] = 1 - 0.001 = 0.999$
   $\mathbf{Pr}[T] = \mathbf{Pr}[T \wedge S] + \mathbf{Pr}[T \wedge \neg S]$
   $\mathbf{Pr}[T] = \mathbf{Pr}[S] * \mathbf{Pr}[T|S] + \mathbf{Pr}[\neg S] * \mathbf{Pr}[T|\neg S]$
   $\mathbf{Pr}[T] = 0.001 * 0.95 + 0.999 * 0.002$
   $\mathbf{Pr}[T] = 0.00095 + 0.001998 = 0.002948$
   Now, $\mathbf{Pr}[S|T] = \frac{\mathbf{Pr}[T \wedge S]}{\mathbf{Pr}[T]}$
   $\mathbf{Pr}[S|T] = \frac{0.001 * 0.95}{0.002948} \approx 0.32225$

Problem 3 (8 pts) Cryptanalysis Concepts.

- (3 pts) Explain what do ciphertext-only attacks, known-plaintext attack, and chosen plaintext attack mean?
  - Ciphertext-only attack: The adversary knows only a number of ciphertexts.
  - Known-plaintext attack: The adversary knows some pairs of ciphertext and corresponding plaintext.
  - Chosen-plaintext attack: The adversary can choose a number of messages and obtain the ciphertexts.
- (2 pts) What attack can be used to break the substitution cipher under a ciphertext-only attack? Explain the simplest way to break it under a known-plaintext attack?
  - Known-plaintext attack can be used to break the substitution cipher under a ciphertext-only attack.
  - The simplest way to break substitution cipher under a known-plaintext attack is to perform a frequency analysis and find out key patterns that translate plaintext to ciphertext in future attacks.
- (3 pts) Explain how one may be able to carry out a known-plaintext attack against the wireless encryption between the laptop used by the target user and a wireless access point. Explain how one may be able to carry out a chosen-plaintext attack.
  - It can be done by mapping known plaintext-ciphertext pairs on hand to the new ciphertext captured between the laptop and the access point. By performing frequency test, one can find out more information about the key.
  - To carry out a chosen-plaintext attack, adversary can repeatedly send to the access point the chosen-plaintext and retrieve corresponding ciphertext. By finding the patterns between them, one can retrieve the key between the plain-text and cipher text.

Problem 4 (8 pts) Consider the following "Double Vigenère encryption". We choose two random keys K_1 and K_2 of lengths ℓ and $\ell + 1$. To encrypt a message, we first use key K_1 to encrypt in the Vigenère fashion, then use K_2 to encrypt.

- (4 pts) Describe how to break this encryption scheme under a ciphertext only attack. Does this double encryption offer increased level of security over Vigenère encryption against a ciphertext only attack?
  - To break the cipher, Kasisky's Test can be used to determine repeated patterns with a distance of common multiples of ℓ and $\ell + 1$, due to the fact that after a distance of common multiples of ℓ and $\ell + 1$, repeating plaintext strings will be encrypted by repeating sections of key strings in both keys. For example:
    PT: The sun met **the man** before **the man** saw the light
    Let $\ell = 2, K_1$: OX
    CT: HES PIK ABH **QVB AXB** YSCCOS **QVB AXB** POT HES IWDVQ
    $\ell + 1 = 3, K_2$: FAN
    CT: MEF UIX FBU **VVO FXO** DSPHOF **VVO FXO** UOG MEF NWQAQ
    Distance between bolded text: 12, a common multiple of 2 and 3.
  - Double encryption offers limited extra level of security when repeated no repeated text in a distance of any common multiples of the two key lengths. If a repetition is found, double encryption gives no extra level of security.

- (4 pts) Suppose that we know that $10 \leq \ell \leq 19$. Given a ciphertext of length 50 and its corresponding plaintext, describe how to recover the keys so that any other message encrypted under the same key pair can be decrypted.
  - Since two levels of encryption is CT = [(PT + K_1) MOD 26 + K_2] MOD 26, it is equivalent to CT = (PT + K_1 + K_2) MOD 26. With cipher text and its corresponding plaintext on hand, K_1 + K_2 for each letter space can be recovered. Since $10 \leq \ell \leq 19$ and lengts of K_1 and K_2 are ℓ and $\ell + 1$ correspondingly, by plotting all combinations of K_1 and K_2, we can bruteforce the K_1 and K_2 pair that convertes plaintext to ciphertext.
  - With K_1 + K_2 sum per letter space on hand, one can encrypt and decrypt under the same format of the original key pair.

Problem 5 (14 pts) Consider the following enhancement of the Vigenère cipher. We assume that the plaintext is a case-insensitive English text using only the 26 letters (without space or any other symbol). To encrypt a plaintext of length n, one first uniformly randomly generates a string over the alphabet $[A..Z]$ of length 17, and then inserts this string into the beginning of the plaintext. That is, we first construct a string $x = x_1 x_2 \ldots x_{n+17}$, such that $x_1 \cdots x_{17}$ is the string we have generated, and $x_{18} \cdots x_{n+17}$ is the original plaintext string. We then construct a string y as follows: $y_i = x_i$ for $1 \leq i \leq 17$, and for $i \in [18, n+17]$, y_i is the result of using y_{i-17} to encrypt x_i; that is, when the x_i's and y_i's are treated as numbers in $[0..25]$, we have $y_i \leftarrow ((x_i + y_{i-17}) \mod 26)$. We then apply the Vigenère cipher to the string y, while making sure that the key length is not a multiple of 17.

- Implement the encryption algorithm and decryption algorithm for this cipher in a programming language of your choice. Include the core part of your code.
  - core part at the end of document
- Choose a key, a plaintext, and run the encryption code multiple times, and ensure that the decryption results in the original plaintext. Include the key, the plaintext, and 3 ciphertexts.
  plaintext: FANCYPLAINTEXTTHATISLONG
  key: SOMEKINDOFKEYNOTAMULTIPLEOFSEVENTEEN
  - ciphertext 1: VONELVAZEYIAHQYRQUUZVHRJAMLJEBAQYUAVLZBED
  - ciphertext 2: XOFOUXMWBARAUNCYVWURFQTVXJNSEOXUFZCVDJKGP
  - ciphertext 3: OKATCHZUUXRLXKATZNQMKYDIVCKSPRUSADTRYOSQC
- Write the Pseudo-code to recover enough information from a known (plaintext,ciphertext) pair to decrypt other messages encrypted under the same key. That is, the pseudo-code takes three inputs (M_1, C_1, C_2), where C_1 is a ciphertext of M_1, and outputs M_2, the message encrypted in C_2.
  - recover(M1,C1,C2)
    - {
      n1 = length of M1;
      char M1FirstPart[17], C2FirstPart[17], C1FirstPart[17];
      char M1NextPart[n - 17], C2NextPart[n - 17], C1NextPart[n - 17];
      divideInto2Parts(M1,M1FirstPart,M1NextPart); //Divide M1 into 2 parts
      divideInto2Parts(C2,C2FirstPart,C2NextPart); //Divide C2 into 2 parts
      divideInto2Parts(C1,C1FirstPart,C1NextPart); //Divide C1 into 2 parts

char keyForFirstPart[17] = kasiskyTest(M1FirstPart,C1FirstPart); //Find key used to encrypt the first 17 elements.
char M2FirstPart[17] = decrypt(C2FirstPart, keyForFirstPart); // Use the key found to decrypt the first 17 elements of C2.
if(keyForFirstPart has repetition)
· {
char key[] = trim(keyForFirstPart); //Find the key in its actual length
char keyPadded[n - 17] = pad(keyForFirstPart, n-17); //Pad the key to length of second part.
char M2NextPart[n - 17] = decrypt(C2NextPart, keyPadded);//Decrypt C2 using the key.
}
· else{
int value;
while(tries $<$ n AND NOTfound) // Loop to try values
{
char key[] = kasiskyTest(M1NextPart,C1NextPart, 17, value); //Use Kasisky Test to find patterns in distance of common multiples of 17 and the value specified until a key is found.
do{ //Skipping values multiple of 17
value++;
}while(value MOD 17 == 0); //end do-while loop
} //end while loop
char keyPadded[n - 17] = pad(key, n-17); //Pad the key to length of second part.
char M2NextPart[n - 17] = decrypt(C2NextPart, keyPadded);//Decrypt C2 using the key.
} //End if-else
Combine2Parts(M1FirstPart, M1NextPart); //Combine the 2 parts.
} //End Recover function

- How to effectively attack this cipher in a ciphertext only attack?
  - Since the first 17 elements are only encrypted once in Vigenere fashion, groups of letters in ciphertext will always match to a certain plaintext-key pair according to Kasisky test, even if the plaintext are randomly generated. Thus the first 17 elements can be obtained. Then, after the 18th element till the end, attack the cipher by finding whether a pattern exsists in a distance of common multiples of 17 (length of first portion) and the non-17-multiple key length.

Hint: You may want to do this question last.

Problem 6 (10 pts) Consider an example of encrypting the result of a 6-side dice (i.e., $M \in [1..6]$), as follows. Uniformly randomly chooses $K \in [1..6]$, ciphertext is $C = (M * K) \mod 13$. The ciphertext space is thus $[1..12]$. We have $\mathbf{Pr}[\mathsf{PT} = 1] = \mathbf{Pr}[\mathsf{PT} = 2] = \mathbf{Pr}[\mathsf{PT} = 3] = \cdots = \mathbf{Pr}[\mathsf{PT} = 6] = 1/6$, and we use a vector notation $\mathbf{Pr}[\mathsf{PT}] = \langle 1/6, 1/6, \ldots, 1/6 \rangle$ to denote this.

- Assume that you stole a glance at the dice value and saw that there are many dots on it, and hence are quite certain that M is either a 5 or a 6. You then learned the ciphertext of the encrypted dice value. Under which ciphtertext value(s) can you learn the value M.

  Hint: You may want to start by writing out the 6 by 6 table of the ciphertext for each possible combination of plaintext and key.

	M = 1	M = 2	M = 3	M = 4	M = 5	M = 6
K = 1	(1,1); C = 1	(1,2); C = 2	(1,3); C = 3	(1,4); C = 4	(1,5); C = 5	(1,6); C = 6
K = 2	(2,1); C = 2	(2,2); C = 4	(2,3); C = 6	(2,4); C = 8	(2,5); C = 10	(2,6); C = 12
K = 3	(3,1); C = 3	(3,2); C = 6	(3,3); C = 9	(3,4); C = 12	(3,5); C = 2	(3,6); C = 5
K = 4	(4,1); C = 4	(4,2); C = 8	(4,3); C = 12	(4,4); C = 3	(4,5); C = 7	(4,6); C = 11
K = 5	(5,1); C = 5	(5,2); C = 10	(5,3); C = 2	(5,4); C = 7	(5,5); C = 12	(5,6); C = 4
K = 6	(6,1); C = 6	(6,2); C = 12	(6,3); C = 5	(6,4); C = 11	(6,5); C = 4	(6,6); C = 10
  - According to the chart, if M is either 5 or 6, with ciphertexts equivalent to 2 or 7, K must be 3 and 4, correspondingly, so M must be 5. I ciphertexts is 6 or 11, K must be 1 and 4, correspondingly, so M must be 6.

- Compute $\mathbf{Pr}[\mathsf{PT}|\mathsf{CT}=i]$, for each $i \in [1..12]$, similar to the one for $\mathbf{Pr}[\mathsf{PT}]$ given above.
  - $\mathbf{Pr}[\mathsf{PT}|\mathsf{CT}=1] = \langle \frac{1}{36}, 0, 0, 0, 0, 0\rangle$
  - $\mathbf{Pr}[\mathsf{PT}|\mathsf{CT}=2] = \langle \frac{1}{36}, \frac{1}{36}, \frac{1}{36}, 0, \frac{1}{36}, 0\rangle$
  - $\mathbf{Pr}[\mathsf{PT}|\mathsf{CT}=3] = \langle \frac{1}{36}, 0, \frac{1}{36}, \frac{1}{36}, 0, 0\rangle$
  - $\mathbf{Pr}[\mathsf{PT}|\mathsf{CT}=4] = \langle \frac{1}{36}, \frac{1}{36}, 0, \frac{1}{36}, \frac{1}{36}, \frac{1}{36}\rangle$
  - $\mathbf{Pr}[\mathsf{PT}|\mathsf{CT}=5] = \langle \frac{1}{36}, 0, \frac{1}{36}, 0, \frac{1}{36}, \frac{1}{36}\rangle$
  - $\mathbf{Pr}[\mathsf{PT}|\mathsf{CT}=6] = \langle \frac{1}{36}, \frac{1}{36}, \frac{1}{36}, 0, 0, \frac{1}{36}\rangle$
  - $\mathbf{Pr}[\mathsf{PT}|\mathsf{CT}=7] = \langle 0, 0, 0, \frac{1}{36}, \frac{1}{36}, 0\rangle$
  - $\mathbf{Pr}[\mathsf{PT}|\mathsf{CT}=8] = \langle 0, \frac{1}{36}, 0, \frac{1}{36}, 0, 0\rangle$
  - $\mathbf{Pr}[\mathsf{PT}|\mathsf{CT}=9] = \langle 0, 0, \frac{1}{36}, 0, 0, 0\rangle$
  - $\mathbf{Pr}[\mathsf{PT}|\mathsf{CT}=10] = \langle 0, \frac{1}{36}, 0, 0, \frac{1}{36}, \frac{1}{36}\rangle$
  - $\mathbf{Pr}[\mathsf{PT}|\mathsf{CT}=11] = \langle 0, 0, 0, \frac{1}{36}, 0, \frac{1}{36}\rangle$
  - $\mathbf{Pr}[\mathsf{PT}|\mathsf{CT}=12] = \langle 0, \frac{1}{36}, \frac{1}{36}, \frac{1}{36}, \frac{1}{36}, \frac{1}{36}\rangle$
- Show that if K is uniformly randomly chosen from $[1..12]$, then this cipher provides perfect secrecy.

	M = 1	M = 2	M = 3	M = 4	M = 5	M = 6
K = 1	(1,1); C = 1	(1,2); C = 2	(1,3); C = 3	(1,4); C = 4	(1,5); C = 5	(1,6); C = 6
K = 2	(2,1); C = 2	(2,2); C = 4	(2,3); C = 6	(2,4); C = 8	(2,5); C = 10	(2,6); C = 12
K = 3	(3,1); C = 3	(3,2); C = 6	(3,3); C = 9	(3,4); C = 12	(3,5); C = 2	(3,6); C = 5
K = 4	(4,1); C = 4	(4,2); C = 8	(4,3); C = 12	(4,4); C = 3	(4,5); C = 7	(4,6); C = 11
K = 5	(5,1); C = 5	(5,2); C = 10	(5,3); C = 2	(5,4); C = 7	(5,5); C = 12	(5,6); C = 4
K = 6	(6,1); C = 6	(6,2); C = 12	(6,3); C = 5	(6,4); C = 11	(6,5); C = 4	(6,6); C = 10
K = 7	(7,1); C = 7	(7,2); C = 1	(7,3); C = 8	(7,4); C = 2	(7,5); C = 9	(7,6); C = 3
K = 8	(8,1); C = 8	(8,2); C = 3	(8,3); C = 11	(8,4); C = 6	(8,5); C = 1	(8,6); C = 9
K = 9	(9,1); C = 9	(9,2); C = 5	(9,3); C = 1	(9,4); C = 10	(9,5); C = 6	(9,6); C = 2
K = 10	(10,1); C = 10	(10,2); C = 7	(10,3); C = 4	(10,4); C = 1	(10,5); C = 11	(10,6); C = 8
K = 11	(11,1); C = 11	(11,2); C = 9	(11,3); C = 7	(11,4); C = 5	(11,5); C = 3	(11,6); C = 1
K = 12	(12,1); C = 12	(12,2); C = 11	(12,3); C = 10	(12,4); C = 9	(12,5); C = 8	(12,6); C = 7

- As can be seen from the chart above, C can be used to represent any value of M with corresponding K. For this reason, $\mathbf{Pr}[\mathsf{CT} = C \mid \mathsf{PT} = M_1] = \mathbf{Pr}[\mathsf{CT} = C \mid \mathsf{PT} = M_2]$ holds.
  $\therefore$ If K is uniformly randomly chosen from $[1..12]$, then this cipher provides perfect secrecy.

Problem 7 (5 pts) Consider the following way of using the Vigenere cipher to send one encrypted message. The possible plaintexts are English texts of length 100. The key is a random string of length 50. Show that this does not satisfy perfect secrecy by finding two plaintext messages M_1, M_2 and a ciphertext message C such that:

$$\mathbf{Pr}[\mathsf{CT} = C_0 \mid \mathsf{PT} = M_1] \neq \mathbf{Pr}[\mathsf{CT} = C_0 \mid \mathsf{PT} = M_2]$$

- M_1 = somelongtext... (length of 100)
  M_2 = C = someothertext... (length of 100)

- $\mathbf{Pr}[\mathsf{CT} = C_0 \mid \mathsf{PT} = M_1] = \mathbf{Pr}$[key matching M_1 to C_0 is chosen] = $\frac{1}{(26)^{50}}$
  Let M_2 = C_0. The only key that possibly match them is when key = all A's.
  $\mathbf{Pr}[\mathsf{CT} = C_0 \mid \mathsf{PT} = M_2] = \mathbf{Pr}$[key matching M_2 to C_0 is chosen] = 1
  $\therefore \mathbf{Pr}[\mathsf{CT} = C_0 \mid \mathsf{PT} = M_1] \neq \mathbf{Pr}[\mathsf{CT} = C_0 \mid \mathsf{PT} = M_2]$

Problem 8 (5 pts) Prove that for any cipher that offers perfect secrecy, the number of the possible keys must be at least as large as the number of plaintexts.

- To reach perfect secrecy, probability of n messages map to the same ciphertext is the same. That means to differentiate n messages out of a single ciphertext requries n different keys, as the same key applied on the ciphertext produces same result. On the other hand, if applying the same key to a ciphertext yields different result, one cannot uniquely distinguish different plaintexts from a single ciphertext.
  $\therefore$ The number of the possible keys must be at least as large as the number of plaintexts.

Problem 9 (10 pts) Implement the following encryption/decryption function, which uses the RC4 stream cipher.

- byte[] encrypt(byte[] pt, byte[] key)
- byte[] decrypt(byte[] ct, byte[] key)

You should implement RC4 algorithm yourself. Google to find out the algorithm. You can assume that the key is an array of length between 16 and 32 bytes. You need to use a 256-bit (32-byte) initial vector so that when one invokes the encrypt function with the same pair of plaintext and key twice, with high probability the resulting ciphertexts are different.

You can choose any programming language to do this problem. You need to figure out what library function to call to generate a random IV. Note that the random IV is required to be unpredictable. This call to generate IV should be the only library call. You should drop the first 3072 bytes of RC4's output as recommended. You should run your code to verify that decrypt(encrypt(pt, key), key) = pt.

Include your code in the HW submission, and provide information regarding the library function you use to generate the random IV.

- Code and screenshots are submitted separately in HW1Q9. Core part at the end of the document.
- The library function called and used in this problem was time(0) to fetch the computer time as seed, srand() to set a seed, and rand() to generate the corresponding sets of random numbers.

Core code part of HW1Q5

HW1Q5.c

```c
void convertToASCII(char* string, int stringLength){
    for(int i = 0; i < stringLength; i++){
        string[i] = string[i] + 65; //Convert to ASCII numbers
    }

}

int getMyKey(char* myKey){
    //Getting the specified key, counting actual key length
    int counter;
    do{
        printf("Enter your key which has a length of non-17 multiple with no space: \n");
        scanf("%s", myKey);
        printf("\n");
        printf( "You entered: %s \n", myKey);
        counter = 0;
        while(counter < plaintextLength && myKey[counter] != '\0'){
            myKey[counter] = toupper(myKey[counter]);
            counter++;
        }
        if(counter % 17 == 0){//Reject keys that does not satisfy the requirement.
            printf("Your key has a length of %d . It is a multiple of 17 and does not fit the length requirement. \n", counter);
        }
    }while(counter % 17 == 0);

    return counter;

}

void keyArrayGen(char* keyPadded, char* myKey, int keySize, int expectedLength){
    for(int i = 0; i < expectedLength; i++){
        keyPadded[i] = myKey[i % keySize];//pad the input key to 32 byte, same as size IV.
    }
    keyPadded[expectedLength] = '\0';
}

void vigenereEncrypt(char* plaintext, char* ciphertext, char* key, int inputLength){
    for(int i = 0; i < inputLength; i++){
        ciphertext[i] = (plaintext[i] + key[i]) % 26;
    }
    ciphertext[inputLength] = '\0';

}
void vigenereDecrypt(char* plaintext, char* ciphertext, char* key, int inputLength){
    for(int i = 0; i < inputLength; i++){
        plaintext[i] = ((ciphertext[i] - key[i]) % 26 + 26) % 26; //Formal modulo operation implemented in case of a negative value modding a positive value
    }
    plaintext[inputLength] = '\0';

}


int main()
{
    char plaintext[plaintextLength];
```

Core code part of HW1Q9

HW1Q9.c

```c

}

void keyArrayGen(char* keyPadded, char* myKey, int keySize){
    for(int i = 0; i < defaultLength; i++){
        keyPadded[i] = myKey[i % keySize];//pad the input key to 32 byte, same as size IV.
    }
    keyPadded[defaultLength] = '\0';
}

void KSA(char* initVector, char* keyPadded){ //Key Scheduling Algorithm
    int j = 0; //Swap index
    for (int i = 0; i < defaultLength; i++){
        j = (j + initVector[i] + keyPadded[i]) % defaultLength;
        /*Swap S[i] and S[j]*/
        char temp;
        temp = initVector[i];
        initVector[i] = initVector[j];
        initVector[j] = temp;
    }
}
void PRGA(char* initVector, char* keyStream){ //Pseudo-random generation algorithm
    int i = 0, j = 0;//Swap index
    int t; //index which the key stream will use
    unsigned int counter = 0; //counter of the keyStream
    for(i = i + 1; counter < defaultTextLength + defaultKeyStreamBytesDropped; counter++){//generate keyStream long enough to drop the first 3072 bytes then XOR with plaintext
        j = (j + initVector[i]) % defaultLength;
        char temp;
        temp = initVector[i];
        initVector[i] = initVector[j];
        initVector[j] = temp;

        t = (initVector[i] + initVector[j]) % defaultLength;
        keyStream[counter] = initVector[t];
        i = (i + 1) % defaultLength;
    }
    keyStream[defaultTextLength + defaultKeyStreamBytesDropped] = '\0';

}
void encrypt(char* encryptResult,char* pt, char* keyStream){
    for(int i = 0; i < defaultTextLength; i++){
        encryptResult[i] = pt[i] ^ keyStream[defaultKeyStreamBytesDropped + 1 + i];//Encrypt plaintext by XOR with key stream that has its first 3072 bytes dropped.
    }
    encryptResult[defaultTextLength] = '\0';
    return;
}
void decrypt(char* decryptResult, char* ct, char* keyStream){
    for(int i = 0; i < defaultTextLength; i++){
        decryptResult[i] = ct[i] ^ keyStream[defaultKeyStreamBytesDropped + 1 + i];//Decrypt ciphertext by XOR with key stream
    }
    decryptResult[defaultTextLength] = '\0';
    return;
}

int main()
```