

CS458/CS558: Introduction to Computer Security

Basic Terminology

Some Basic Terminology

- ◆ **Plaintext**: original message
- ◆ **Ciphertext**: coded message
- ◆ **Encryption**: converting plaintext to ciphertext
- ◆ **Decryption**: restoring plaintext from ciphertext
- ◆ **Cryptography**: the area of study of encryption principles/methods
- ◆ **Cipher**: an algorithm for performing encryption
- ◆ **Cryptanalysis**: the area of study of principles/ methods of deciphering ciphertext without knowing key - breaking the cipher

Some Basic Terminology

- ◆ **Cryptology**: areas of cryptography and cryptanalysis together
- ◆ **Secret key**: the input of encryption algorithm. The key is independent of the plaintext and the alg..

Cryptography

- ◆ Characterize cryptographic system by:
 - ❖ Type of encryption operations used
 - **Substitution**: each element (a bit or a letter) in the plaintext is mapped into another element
 - **Transposition**: elements in the plaintext are rearranged.
 - **product**: multiple stages of substitutions and transpositions
 - ❖ Number of keys used
 - **Symmetric, Single-key** encryption
 - **Asymmetric, Two-key** or **Public-key** encryption
 - ❖ Way in which plaintext is processed
 - **Block**: process one block of elements at a time, producing an output block for each input block
 - **Stream**: process the input elements continuously, producing one element at a time.

Cryptanalysis

- ◆ **Objective of attacking an encryption system**: recover key rather than simply to recover the plaintext of a single ciphertext.
- ◆ General approaches:
 - ❖ **Cryptanalytic attack**:
 - Rely on the nature of the algorithm + some knowledge of the plaintext (e.g. English or French) or some sample plaintext-ciphertext pairs.
 - ❖ **Brute-force attack**
 - Try every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained.

Chapter 2

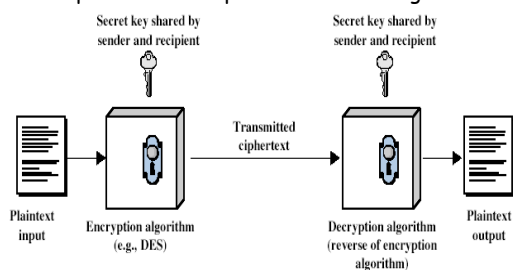
Symmetric Encryption

Symmetric Encryption

- A form of cryptosystem in which encryption and decryption are performed using the same key - **single-key encryption**
- Was only type prior to invention of public-key in 1970's, and by far most widely used.

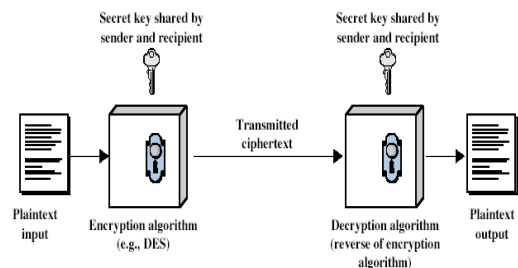
Symmetric Cipher Model

- **Encryption algorithm**: performs various substitutions and transformation on the plaintext.
- **Secret key**: the input of encryption algorithm. The key is independent of the plaintext and the alg..



Symmetric Cipher Model

- **Decryption algorithm**: the ciphertext and the secret key and produces the original plaintext.



Requirements

- Two requirements for secure use of symmetric encryption:
 - ❖ A **strong encryption algorithm**: the opponent should be unable to decrypt ciphertext or discover the key even if he/she has a number of ciphertexts and the plaintext that produced each ciphertext.
 - ❖ Assume encryption algorithm is known
 - ❖ Mathematically have:
 - X**: message, **K**: encryption key, **Y**: ciphertext
 - $Y = E(K, X)$
 - $X = D(K, Y)$
- An opponent can observe Y , but do not have access to K or X .

Requirements

- Two requirements for secure use of symmetric encryption:
 - ❖ A **secret key** known only to sender/receiver
 - Sender and receiver must have obtained copies of secret key in a secure fashion and must keep the key secure.
 - A **third party** could generate the key and securely deliver it to both source and destination.
 - If someone can discover the **key** and knows the **algorithm**, all communication using this key is readable.

Unconditional Security

Unconditional security

- ❖ No matter how much computer power or time is available, the cipher cannot be broken since the ciphertext provides insufficient information to determine the corresponding plaintext.

Computational Security

Computational security

- ❖ Given limited computing resources (eg time needed for calculations is greater than age of universe), the cipher cannot be broken
 - The cost of breaking the cipher exceeds the value of the encrypted information
 - The time required to break the cipher exceeds the useful lifetime of the information

Brute Force Search

- ❖ Simply try every key until an intelligible translation of the ciphertext into plaintext is obtained.
- ❖ On average, **half** of all possible keys must be tried to achieve success - proportional to key size
- ❖ DES: 56-bit, triple DES: 168-bit, AES: > 128 bits
- ❖ Time required for various key spaces:

Key Size (bits)	Number of Alternative Keys	Time required at 1 decryption/ μ s	Time required at 10^6 decryptions/ μ s
32	$2^{32} = 4.3 \cdot 10^9$	$2^{31} \mu\text{s} = 35.8 \text{ minutes}$	2.15 milliseconds
56	$2^{56} = 7.2 \cdot 10^{16}$	$2^{55} \mu\text{s} = 1142 \text{ years}$	10.01 hours
128	$2^{128} = 3.4 \cdot 10^{38}$	$2^{127} \mu\text{s} = 5.4 \cdot 10^{24} \text{ years}$	$5.4 \cdot 10^{18} \text{ years}$
168	$2^{168} = 3.7 \cdot 10^{50}$	$2^{167} \mu\text{s} = 5.9 \cdot 10^{36} \text{ years}$	$5.9 \cdot 10^{30} \text{ years}$

Substitution Cipher

Classical Substitution Ciphers

- ❖ Letters of plaintext are replaced by other letters or by numbers or symbols

Caesar Cipher

- ❖ The earliest known substitution cipher (by Julius Caesar)
- ❖ First attested use in military affairs
- ❖ Replaces each letter with the letter standing **K** places further down the alphabet.
- ❖ When **K = 3**, can define transformation as:
 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
 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
- ❖ Example:
 Plaintext: meet me after the toga party

Caesar Cipher

- ◆ The earliest known substitution cipher (by Julius Caesar)
- ◆ First attested use in military affairs
- ◆ Replaces each letter with the letter standing **K** places further down the alphabet.

- ◆ When **K=3**, can define transformation as:

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
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

- ◆ Example:

Plaintext: meet me after the toga party

Ciphertext: PHHW PH DIWHU WKH WRJD SDUWB

Cryptanalysis of Caesar Cipher

Cryptanalysis of Caesar Cipher

- ◆ A **brute force** search can be easily performed: simply try all the 25 possible keys - far from security.

- ◆ The **language** of the plaintext is known and easily recognizable.

- ◆ The input may be compressed, make recognition difficult, e.g., E.g. .zip file.

KEY	
1	oggv og chvgt vjg vqic zctva
2	nifu nf bquf uif uphb qbuis
3	meet me after the toga party
4	ldds ld zeadq agd sniz ozgsk
5	koer ko ydrop rfc rmey nyprv
6	jbbq jb xeqbo qeb qidk mxoqv
7	iaap ia wbpas pda pkov lwmpu
8	hzso hz vacem ocs ojvw kvnot
9	tyyn ty uzryl nby niau julns
10	faxm fx tymxk max mht itkmr
11	ewel ew xklw j lzw lqys hsljq
12	dvvk dv rkvvi kyv kfkr grikp
13	cuu j cu qvjuh jxu jewq fqhjo
14	btti bt puitg iwt idvp epgin
15	assh as othef hvs hcuo dofhm
16	zrrg zr nsgre gur gbtu cneql
17	yqef yq urqfd ftq fasm bndfk
18	xppe xp lqepc eap ezrl alcej
19	wood wo kpdob dro dyqk zkbdi
20	vnnc vn joona cqn expj yjach
21	ummb um inbmz bpm bwoi xizbg
22	tila ti hmaly aol avnh whyaf
23	skkz sk glakx znk zumg vxgze
24	rjly rj fkyjw ymj ytlf ufwyd
25	qilx qi ejxiv xli xake tewxc

Monoalphabetic Cipher

- ◆ Allow an **arbitrary** substitution rather than just shifting the alphabet
- ◆ Each plaintext letter maps to a **random** ciphertext letter

Plain: abcdefghIjklmnopqrstuvwxyz

Cipher:DKVQFIBJWPESCXHTMYAUOLRGZN

E.g.

If we wish to replace letters

Monoalphabetic Cipher

- ◆ Allow an **arbitrary** substitution rather than just shifting the alphabet
- ◆ Each plaintext letter maps to a **random** ciphertext letter

Plain: abcdefghIjklmnopqrstuvwxyz

Cipher:DKVQFIBJWPESCXHTMYAUOLRGZN

E.g.

If we wish to replace letters

WI RF RWAJ UH YFTSDVF SFUUFYA

Monoalphabetic Cipher Security

- ◆ Number of mappings

Monoalphabetic Cipher Security

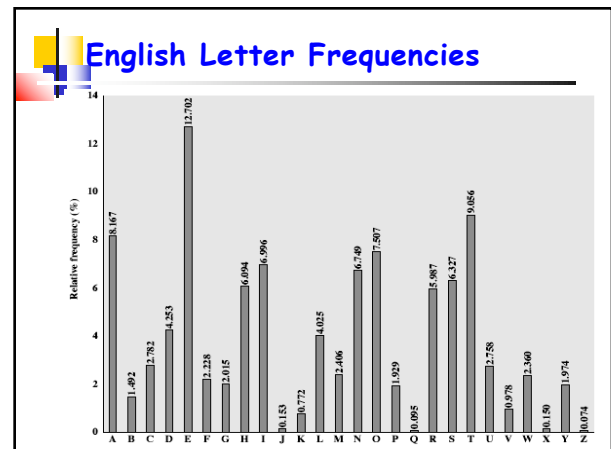
- The number of mappings
 $26! = 4 \times 10^{26}$ mappings
- With so many mappings, might think is secure

Monoalphabetic Cipher Security

- The cipher line can be any permutation of the 26 alphabetic characters
 $26! = 4 \times 10^{26}$ mappings
- With so many mappings, might think is secure
- But would be **WRONG** - if the cryptanalyst knows the nature of the plaintext (e.g. noncompressed English text), then the analyst can exploit the regularities (**frequency of letters**) of the language

Language Redundancy and Cryptanalysis

- Human languages are redundant. Letters are not equally commonly used.
- In English **E** is by far the most common letter, followed by **T, A, O, I, N, S, R**, other letters like **Z, J, K, Q, X** are fairly rare.
- If the msg. is **long** enough, this technique alone may be sufficient.



Use in Cryptanalysis

- Key concept - monoalphabetic substitution ciphers do not change relative letter frequencies
- Calculate letter frequencies for ciphertext
- Compare counts/plots against known values

Example

UZQSOVUOHXMOFPVGPOZPEVAGZWSZOPFPESXUDBMETSXAIZVUEPH
 ZHMDZSHZOWSFPAPPDTSVPQZWMXUZUHSX
 EPYEPDPDZSUFPOMBZWPFPZHMJDJDTMOGMQ

P 13.33 (16/120)	H 5.83	F 3.33	B 1.67	C 0.00
Z 11.67	D 5.00	W 3.33	G 1.67	K 0.00
S 8.33	E 5.00	Q 2.50	Y 1.67	L 0.00
U 8.33	V 4.17	Tt 2.50	J 0.83	N 0.00
O 7.50	X 4.17	A 1.67	I 0.83	R 0.00
M 6.67				

Example

UZQSOVUOHXMOPVGPOZPEVAGZWSZOPPFESXUDBMETSXAIZVUEPH
ZHMDZSHZOWSFPAPPDTSVPPQUZWMXUZHXS
EPYEPOPDZSUFFPOMBZPFPUPZHMDJUDTMOGMQ

P 13.33	H 5.83	F 3.33	B 1.67	C 0.00
Z 11.67	D 5.00	W 3.33	G 1.67	K 0.00
S 8.33	E 5.00	Q 2.50	Y 1.67	L 0.00
U 8.33	V 4.17	Tt 2.50	J 0.83	N 0.00
O 7.50	X 4.17	A 1.67	I 0.83	R 0.00
M 6.67				

• $P \rightarrow e$

Example

UZQSOVUOHXMOPVGPOZPEVAGZWSZOPPFESXUDBMETSXAIZVUEPH
ZHMDZSHZOWSFPAPPDTSVPPQUZWMXUZHXS
EPYEPOPDZSUFFPOMBZPFPUPZHMDJUDTMOGMQ

P 13.33	H 5.83	F 3.33	B 1.67	C 0.00
Z 11.67	D 5.00	W 3.33	G 1.67	K 0.00
S 8.33	E 5.00	Q 2.50	Y 1.67	L 0.00
U 8.33	V 4.17	Tt 2.50	J 0.83	N 0.00
O 7.50	X 4.17	A 1.67	I 0.83	R 0.00
M 6.67				

• $P \rightarrow e, Z \rightarrow t$

Example

UZQSOVUOHXMOPVGPOZPEVAGZWSZOPPFESXUDBMETSXAIZVUEPH
ZHMDZSHZOWSFPAPPDTSVPPQUZWMXUZHXS
EPYEPOPDZSUFFPOMBZPFPUPZHMDJUDTMOGMQ

P 13.33	H 5.83	F 3.33	B 1.67	C 0.00
Z 11.67	D 5.00	W 3.33	G 1.67	K 0.00
S 8.33	E 5.00	Q 2.50	Y 1.67	L 0.00
U 8.33	V 4.17	Tt 2.50	J 0.83	N 0.00
O 7.50	X 4.17	A 1.67	I 0.83	R 0.00
M 6.67				

• $P \rightarrow e, Z \rightarrow t, \{S, U, O, M, H\} \rightarrow \{a, h, i, n, o, r, s\}$

Example

UZQSOVUOHXMOPVGPOZPEVAGZWSZOPPFESXUDBMETSXAIZVUEPH
ZHMDZSHZOWSFPAPPDTSVPPQUZWMXUZHXS
EPYEPOPDZSUFFPOMBZPFPUPZHMDJUDTMOGMQ

P 13.33	H 5.83	F 3.33	B 1.67	C 0.00
Z 11.67	D 5.00	W 3.33	G 1.67	K 0.00
S 8.33	E 5.00	Q 2.50	Y 1.67	L 0.00
U 8.33	V 4.17	Tt 2.50	J 0.83	N 0.00
O 7.50	X 4.17	A 1.67	I 0.83	R 0.00
M 6.67				

• $P \rightarrow e, Z \rightarrow t, \{S, U, O, M, H\} \rightarrow \{a, h, i, n, o, r, s\}$
• Most common pair: ZW \rightarrow and hence ZWP \rightarrow , ZWSZ \rightarrow

Example

UZQSOVUOHXMOPVGPOZPEVAGZWSZOPPFESXUDBMETSXAIZVUEPH
ZHMDZSHZOWSFPAPPDTSVPPQUZWMXUZHXS
EPYEPOPDZSUFFPOMBZPFPUPZHMDJUDTMOGMQ

P 13.33	H 5.83	F 3.33	B 1.67	C 0.00
Z 11.67	D 5.00	W 3.33	G 1.67	K 0.00
S 8.33	E 5.00	Q 2.50	Y 1.67	L 0.00
U 8.33	V 4.17	Tt 2.50	J 0.83	N 0.00
O 7.50	X 4.17	A 1.67	I 0.83	R 0.00
M 6.67				

• $P \rightarrow e, Z \rightarrow t, \{S, U, O, M, H\} \rightarrow \{a, h, i, n, o, r, s\}$
• Most common pair: ZW \rightarrow th and hence ZWP \rightarrow , ZWSZ \rightarrow

Example

UZQSOVUOHXMOPVGPOZPEVAGZWSZOPPFESXUDBMETSXAIZVUEPH
ZHMDZSHZOWSFPAPPDTSVPPQUZWMXUZHXS
EPYEPOPDZSUFFPOMBZPFPUPZHMDJUDTMOGMQ

P 13.33	H 5.83	F 3.33	B 1.67	C 0.00
Z 11.67	D 5.00	W 3.33	G 1.67	K 0.00
S 8.33	E 5.00	Q 2.50	Y 1.67	L 0.00
U 8.33	V 4.17	Tt 2.50	J 0.83	N 0.00
O 7.50	X 4.17	A 1.67	I 0.83	R 0.00
M 6.67				

• $P \rightarrow e, Z \rightarrow t, \{S, U, O, M, H\} \rightarrow \{a, h, i, n, o, r, s\}$
• Most common pair: ZW \rightarrow th and hence ZWP \rightarrow the, ZWSZ \rightarrow

Example

UZQSOVUOHXMOVPVGPOZPEVAGZWSZOPPFESXUDBMETSXAIZVUEPH
ZHMDSHZOWSFPPAPDTSVPQZWMXZUHSX
EPYEPDPDZSUFFPOMBZWPFPZHMJDJDTMOGMQ

P 13.33	H 5.83	F 3.33	B 1.67	C 0.00
Z 11.67	D 5.00	W 3.33	G 1.67	K 0.00
S 8.33	E 5.00	Q 2.50	Y 1.67	L 0.00
U 8.33	V 4.17	Tt 2.50	J 0.83	N 0.00
O 7.50	X 4.17	A 1.67	I 0.83	R 0.00
M 6.67				

- P → e, Z → t, {S,U,O,M,H} → {a,h,i,n,o,r,s}
- Most common pair: ZW → th and hence ZWP → the, ZWSZ → that
- Finally: it was disclosed yesterday that several informal but direct contacts have been made with political representatives of the viet cong in moscow

Playfair Cipher

- Not even the large number of mappings (4×10^{26} mappings) in a monoalphabetic cipher provides security
- One approach to improving security is to **encrypt multiple letters** - e.g. **Playfair Cipher**.

Playfair Key Matrix

- A **5X5** matrix of letters based on a keyword
- Fill in **letters of keyword** (minus duplicates) from left to right and from top to bottom
- Fill rest of matrix with **other letters**
- Eg. using the keyword **MONARCHY**
- **I** and **j** count as one letter.

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

Playfair Cipher: Encryption

- Plaintext is encrypted **two letters** at a time
 - If a pair is a repeated letter, insert filler **x**, e.g. **balloon** → **bo lx lo on**
 - If both letters fall in the same row, replace each with **letter to right** with the first element of the row circularly following the last, e.g. **ar** → **rm**
 - If both letters fall in the same column, replace each with the letter below it (again wrapping to top from bottom), e.g. **mu** → **cm**

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

Playfair Cipher: Encryption

- Plaintext is encrypted two letters at a time
 - Otherwise, each letter is replaced by the letter in the same row and in the column of the other letter of the pair, e.g. **hs** → **bp**, **ea** → **im** (or **jm**)

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

Playfair Cipher: Decryption

- To decrypt, use the inverse of the encryption rules and drop any extra **x** that does not make sense in the final message.
 - Decrypts two letters at a time
 - If both letters fall in the same row, replace each with **letter to left**, e.g. **rm** → **ar**
 - If both letters fall in the same column, replace each with the letter **above** it, e.g. **cm** → **mu**

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

Playfair Cipher: Decryption

- Plaintext is encrypted two letters at a time
- Otherwise, each letter is replaced by the letter in the same row and in the column of the other letter of the pair
- E.g. bp → hs, im → ea

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

Example

- Use the playfair cipher and key word **common** to encrypt the message: **gdddcgs**

Security of Playfair Cipher

- Security much improved over monoalphabetic
- Would need a **676 (26*26)** entry frequency table to analyse (verses 26 for a monoalphabetic)
- Was widely used for many years
 - eg. by US & British military in WW1
- It is **relatively easy** to break because it still leaves much of the structure of the plaintext language intact.

Polyalphabetic Ciphers

- Improve security using **multiple monoalphabetic substitutions**.
- Features
 - A set of related **monoalphabetic substitution rules** is used.
 - A **key** determines which particular rule is chosen for a given transformation
- Vigenère Cipher**: Simplest polyalphabetic substitution cipher

Modern Vigenère Tableau

Key	Plaintext																									
	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

The set of related monoalphabetic substitution rules consists of **26 Caesar ciphers**

Key	Plaintext																									
	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

- Each caesar cipher is denoted by a **key** letter
- A normal alphabet for the plaintext runs across the top

		Plaintext																									
		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
Key	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

Given a key letter **x** and a plaintext letter **y**, the ciphertext letter is at the intersection of row labeled **x** and the column labeled **y** → the ciphertext is **v**.

		Plaintext																									
		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
Key	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

Write the **plaintext** out, write the **keyword** repeated above it
 Eg using keyword **deceptive**
key: deceptive
plaintext: wearediscoveredsaveyourself

		Plaintext																									
		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
Key	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

Write the **plaintext** out, write the **keyword** repeated above it
 Eg using keyword **deceptive**
key: deceptive
plaintext: wearediscoveredsaveyourself
ciphertext: ZICVTWQNGRZGVTVAVZHCQYGLMGJ

Vigenère Cipher: Decryption

- Decryption:
 - The **key letter** identifies the **row**
 - The position of the **ciphertext letter** in that row determines the **column**
 - The plaintext letter is

Autokey Cipher: Decryption

- Eg. given
 - key **deceptive**
 - Ciphertext: **ZICVTWQNGKZEIIGASXSTSLVVWLA**

key: deceptive
 ciphertext: **ZICVTWQNGKZEIIGASXSTSLVVWLA**
 plaintext:

Autokey Cipher: Decryption

- Eg. given
 - key **deceptive**
 - Ciphertext: **ZICVTWQNGKZEIIGASXSTSLVVWLA**

key: deceptive
 ciphertext: **ZICVTWQNGKZEIIGASXSTSLVVWLA**
 plaintext: **w**

Autokey Cipher: Decryption

- Eg. given
 - key **deceptive**
 - Ciphertext: **ZICVTWQNGKZEIIGASXSTSLVVWLA**

key: deceptivew
 ciphertext: **ZICVTWQNGKZEIIGASXSTSLVVWLA**
 plaintext: **w**

Autokey Cipher: Decryption

- Eg. given
 - key **deceptive**
 - Ciphertext: **ZICVTWQNGKZEIIGASXSTSLVVWLA**

key: deceptivew
 ciphertext: **ZICVTWQNGKZEIIGASXSTSLVVWLA**
 plaintext: **we**

Autokey Cipher: Decryption

- Eg. given
 - key **deceptive**
 - Ciphertext: **ZICVTWQNGKZEIIGASXSTSLVVWLA**

key: deceptivewe
 ciphertext: **ZICVTWQNGKZEIIGASXSTSLVVWLA**
 plaintext: **we**

Autokey Cipher: Decryption

- Eg. given
 - key **deceptive**
 - Ciphertext: **ZICVTWQNGKZEIIGASXSTSLVVWLA**

key: deceptivewearediscoveredsaves
 ciphertext: **ZICVTWQNGKZEIIGASXSTSLVVWLA**
 plaintext: **wearediscoveredsavesyourself**

One-Time Pad

- Use a **random** key that is **as long as** the message.
- The key is used to encrypt and decrypt a **single** message, and then is discarded: one-time pad

One-Time Pad

- Use a **random** key that is **as long as** the message.
- The key is used to encrypt and decrypt a **single** message, and then is discarded: one-time pad
- **Problem:** generation & safe distribution of key
 - ❖ Is of limited utility

Flu, Fever, etc

- ❖ Please do **not** attend the class if you have flu, fever, bad cough, or any infectious diseases
- ❖ If you are not be able to attend the class due to sickness, please email me **before** the class

63

Assignment 1

Assignment 1

Due: 11:59pm, 02/18/2017 (Sunday)

This assignment is done by a group of 2 students

1. Learn how to write and use **makefile**
<http://www.delorie.com/djgpp/doc/ug/larger/makefiles.html> (C)
<http://www.cs.swarthmore.edu/~newhall/unixhelp/javamakefiles.html> (Java)
2. Implement a telnet client and an **iterative** telnet server using TCP sockets.
3. Implement the monoalphabetic cipher

PART I: Telnet Client/Server

Implement a telnet client and an **iterative** telnet server using TCP sockets.

- ❖ Support commands: **ls**, **mkdir**, **rmdir**, **cd**, **pwd**, and **exit**.

telnet > ls

PART I: Telnet Client/Server

Implement a telnet client and an **iterative** telnet server using TCP sockets.

- ❖ Support commands: **ls**, **mkdir**, **rmdir**, **cd**, **pwd**, and **exit**.

```
telnet > ls //lists contents of the directory on the server side
telnet > mkdir <dir>
```

PART I: Telnet Client/Server

Implement a telnet client and an **iterative** telnet server using TCP sockets.

- ❖ Support commands: **ls**, **mkdir**, **rmdir**, **cd**, **pwd**, and **exit**.

```
telnet > ls //lists contents of the directory on the server side
telnet > mkdir <dir> //create a directory <dir> on the server
telnet > rmdir <dir>
```

PART I: Telnet Client/Server

Implement a telnet client and an **iterative** telnet server using TCP sockets.

- ❖ Support commands: **ls**, **mkdir**, **rmdir**, **cd**, **pwd**, and **exit**.

```
telnet > ls //lists contents of the directory on the server side
telnet > mkdir <dir> //create a directory <dir> on the server
telnet > rmdir <dir> //delete directory <dir>
telnet > cd <dir-relative-path>
```

PART I: Telnet Client/Server

Implement a telnet client and an **iterative** telnet server using TCP sockets.

- ❖ Support commands: **ls**, **mkdir**, **rmdir**, **cd**, **pwd**, and **exit**.

```
telnet > ls //lists contents of the directory on the server side
telnet > mkdir <dir> //create a directory <dir> on the server
telnet > rmdir <dir> //delete directory <dir>
telnet > cd <dir-relative-path> //change the current working
//directory to a path that is relative to
//the current directory, e.g. cd path
//<dir-relative-path> does not contain '/'
telnet > cd ..
```

PART I: Telnet Client/Server

Implement a telnet client and an **iterative** telnet server using TCP sockets.

- ❖ Support commands: **ls**, **mkdir**, **rmdir**, **cd**, **pwd**, and **exit**.

```
telnet > ls //lists contents of the directory on the server side
telnet > mkdir <dir> //create a directory <dir> on the server
telnet > rmdir <dir> //delete directory <dir>
telnet > cd <dir-relative-path> //change the current working
//directory to a path that is relative to
//the current directory, e.g. cd path
//<dir-relative-path> does not contain '/'
telnet > cd .. //move up one folder
telnet > pwd
```

PART I: Telnet Client/Server

Implement a telnet client and an **iterative** telnet server using TCP sockets.

- ❖ Support commands: **ls**, **mkdir**, **rmdir**, **cd**, **pwd**, and **exit**.

```
telnet > ls //lists contents of the directory on the server side
telnet > mkdir <dir> //create a directory <dir> on the server
telnet > rmdir <dir> //delete directory <dir>
telnet > cd <dir-relative-path> //change the current working
//directory to a path that is relative to
//the current directory, e.g. cd path
//<dir-relative-path> does not contain '/'
telnet > cd .. //move up one folder
telnet > pwd //print the working directory of the server
telnet > exit //ends the telnet session with the server
```

Telnet Server

The server is invoked as:

```
telnetserv <server_port> (C/C++)
java TelnetServ <server_port> (Java)
```

- ❖ **<server_port>**: the port at which the server accepts connection requests.
 - Use a unique port number such as last 4 digits of your ID.

Telnet Client

The client is invoked as:

```
telnetscli <server_domain> <server_port> (C/C++)
java TelnetCli <server_domain><server_port> (Java)
```

- ❖ **<server_domain>**: the domain name of the machine hosting the server. If you use C/C++, you will need to convert the domain to the 32-bit IP address using `gethostbyname(...)` etc.
 - <http://retran.com/beej/gethostbyname.html>
- ❖ **<server_port>**: the port number on which the server is listening.
- ❖ Upon connecting to the server, the client prints out **telnet>**, which allows the user to type commands.

Error Handling

- ❖ CS458: no error handling is required
- ❖ CS558: **Error handling**
 - ❖ **Invalid commands**, i.e., commands other than `ls`, `cd`, `pwd`, `mkdir`, `rmdir`, and `exit` specified above.
 - You should not implement commands that are supported in Linux/Unix, but are not specified in the description.
 - ❖ `telnet > mkdir <dir>`: **<dir> exists.**
 - ❖ `telnet > rmdir <dir>`: **<dir> does not exist**
 - ❖ `telnet > cd <dir>`: **<dir> does not exist.**
 - Your implementation should NOT support absolute path in `cd` command.

PART II: Monoalphabetic cipher

- Implement **monoalphabetic cipher** to encrypt/decrypt files
 - ❖ The file contains **only a-z** and does not contain space, tabs, and newlines.
 - ❖ Assume that the file contains less than 10000 characters.
- For each plaintext letter, your program should **randomly** generate the corresponding ciphertext letter based on a **seed**.
- Your code should result in an executable of the following form: `./mono <inputfile> <outputfile> <seed> 1/0`
 - **inputfile**: input file name
 - **outputfile**: output file name
 - **seed**: the seed used to generate the mapping
 - **1/0**: encryption/decryption
- Your program prints **the mappings** generated using the following format: `a-b, b-d, c-h,`

Grading Guideline (CS558)

- ❖ Readme, correct format of execution: **4'**
- ❖ Makefile: **8'**
- ❖ Part I: **50'**
 - ❖ `ls, exit`: **12'**
 - ❖ `cd, mkdir, rmdir, pwd`: **26'**
 - ❖ Error handling: **12'**
- ❖ Part II: **38'**

Grading Guideline (CS458)

- ❖ Readme, correct format of execution: **4'**
- ❖ Makefile: **8'**
- ❖ Part I: **46'**
 - ❖ `ls, exit`: **16'**
 - ❖ `cd, mkdir, rmdir, pwd`: **30'**
- ❖ Part II: **42'**

Submission Guideline

- ◆ Only **one** of the group members should submit the program
- ◆ Hand in your **source code**, two **makefiles** (**one for Part I and one for Part II**) and a **Readme** electronically.
- ◆ Part I
 - ❖ If you use C/C++, the Makefile must give the executable server code the name **telnet serv** and the executable client code the name **telnet cli**.
 - ❖ If you use Java, generate **TelnetServ.class** and **TelnetCli.class**.
- ◆ Part II: The name of the executable must be **mono**.
- ◆ Your code should compile and run correctly on **binguns.binghamton.edu**.

Submission Guideline

- ◆ Write a **README** file (text file, do not submit a .doc file):
 - Names and email addresses of all group members.
 - Whether your code has been tested on bingsuns.
 - The language you are using (C/C++/Java)
 - How to execute your program.
 - (Optional) Anything special about your submission that the TA should take note of.

Submission Guideline (Cont.)

- ◆ Create two folders: **part1** (source code of PART I and a Makefile) and **part2** (source code of PART II and a Makefile).
- ◆ Place **both folders** and the **README** file under one directory with a unique name (e.g. p1-[userid] for assignment 1, e.g. p1-pyang).
- ◆ Tar the contents of this directory:


```
tar -cvf p1-<userid>.tar p1-<userid>
```

 E.g. `tar -cvf p1-pyang.tar p1-pyang/`
- ◆ Use the mycourses to upload the **tared** file you created above.

Academy Integrity

- ◆ We will use **moss** to detect to plagiarism in this assignment.