Cryptography!
Lab 3

Prepared by Matt Bennett
for ELEC 3150
Professor Carpenter - Fall 2016

I - Introduction

The goal of this lab was to encrypt and decrypt a line of ASCII characters using two separate C++ programs that use cin and cout functions to read input from the console and return the corresponding resulting characters.

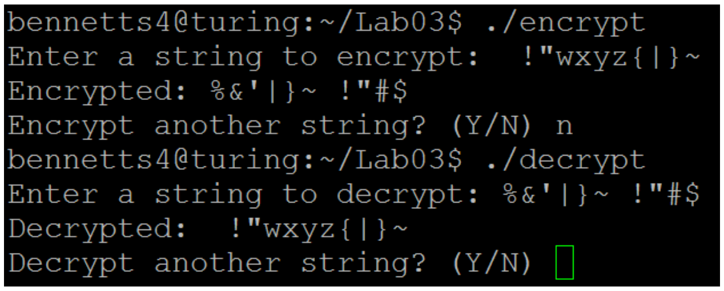
II - Procedure and Assumptions

The basic encryption algorithm implements a positive five bit mask. (Characters that overflow the 127 ASCII limit restart at ASCII value 32, the beginning of visible characters.) The decryption algorithm, the inverse of the encryption algorithm, implements a negative five bit mask. (Characters that overflow past the 32 visible character limit, restart at ASCII value 126.) The programs both assume that user input remains in visible ASCII characters only (32-126), as display or entry of non-visible, computational, characters do not provide useful meaning to text input or output.

III - Results

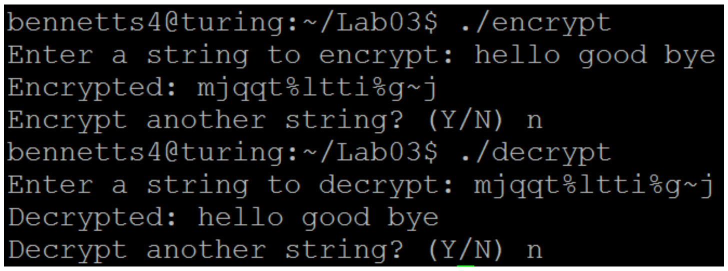
Decrypted Char	Decrypted ASCII	Encrypted ASCII	Encrypted Char
[SPACE]	32	37	%
!	33	38	&
"	34	39	'
...	...	...	...
w	119	124	|
x	120	125	}
y	121	126	~
z	122	32	[SPACE]
{	123	33	!
|	124	34	"
}	125	35	#
~	126	36	$

Table 1: Corresponding Encrypted and Decrypted Character Expected Behavior Boundary Cases



```
bennetts4@turing:~/Lab03$ ./encrypt
Enter a string to encrypt:  !"wxyz{|}~
Encrypted: %&'|}~ !"#$
Encrypt another string? (Y/N) n
bennetts4@turing:~/Lab03$ ./decrypt
Enter a string to decrypt: %&'|}~ !"#$
Decrypted:  !"wxyz{|}~
Decrypt another string? (Y/N)
```

Figure 1: Verification of Boundary Cases Encryption and Decryption as Shown in Table 1



```
bennetts4@turing:~/Lab03$ ./encrypt
Enter a string to encrypt: hello good bye
Encrypted: mjqqt%ltti%g~j
Encrypt another string? (Y/N) n
bennetts4@turing:~/Lab03$ ./decrypt
Enter a string to decrypt: mjqqt%ltti%g~j
Decrypted: hello good bye
Decrypt another string? (Y/N) n
```

Figure 2: Verification of Multiple Word Encryption Entry and Corresponding Decryption

IV - Analysis

The encryption and decryption behavior of the two programs function as expected and shown in brief in Table 1. The resulting encryption of input is correctly reversed when input into the decryption program. The encryption and decryption boundary cases, corresponding to the boundary visible characters located a decimal values 32 and 126, in addition to the five bit circular overlapping, were tested and verified to be functioning correctly in Figure 1. The desired encryption of full lines with space separation within the input text was tested and verified to be working correctly in Figure 2.

V - Conclusions

The encryption and decryption algorithm utilises a basic ASCII rotation to obscure and un-obscure a given line of text entry. Further development of the program could be altered to change the degree of rotation or, in the case of the decryption algorithm, to test a series of different rotations and compare the resulting strings to a dictionary to determine the most likely rotation used to encrypt the input text line.