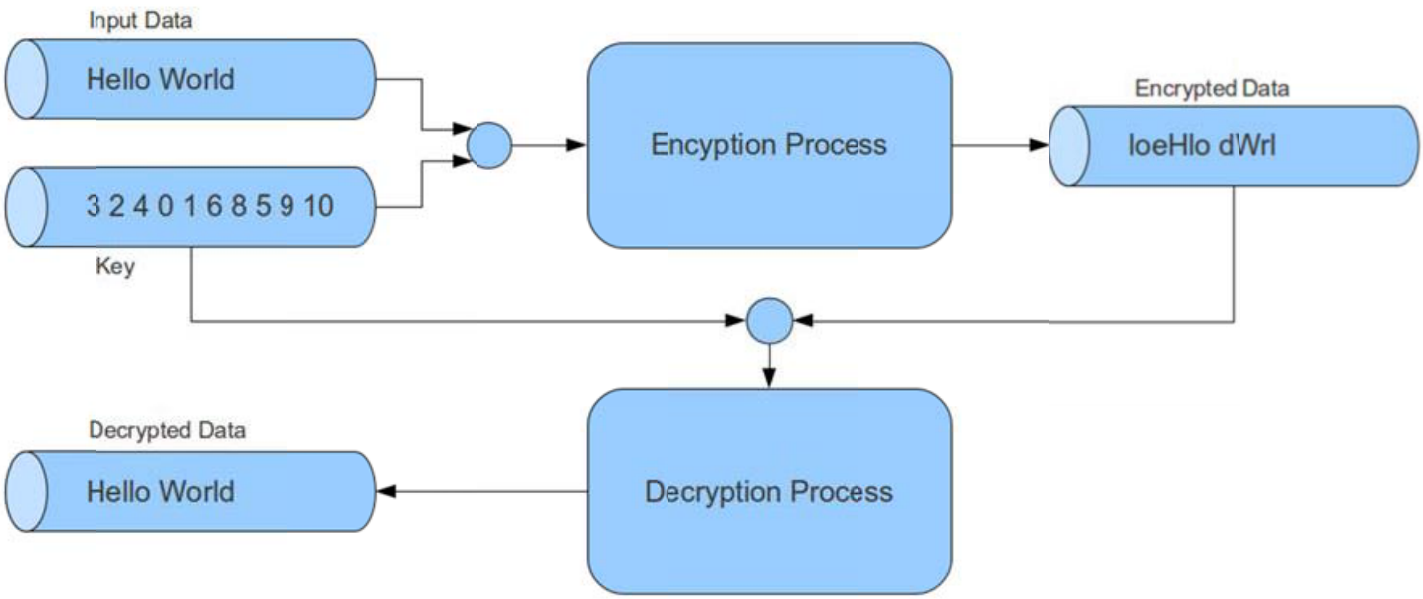
60-141 – Introduction to Programming II Winter, 2014
Assignment 2

Array

In this assignment you will implement an encryption and a corresponding decryption algorithm which involves only random shuffling of the input data. The process is explained in the following diagram.



Try to figure out the relationship between the Input data, the key and the output (encrypted) data. The following table may be handy. The decryption process just does the opposite of the encryption process, which is to restore the original content.

Index	Input String	Key	Encrypted String	Index
0	H	3	l	0
1	e	2	o	1
2	l	4	e	2
3	l	0	H	3
4	o	1	l	4

A skeleton program is provided for you with prototypes of the required functions and some function calls that you may not be familiar with, yet. Read the instructions in the program and build your solution around it (call it **Assign2.c**). You **must** implement all the **unimplemented functions without altering** their prototypes and some statements in the main, (look for the flag **TODO**). You are **NOT allowed to** include any additional header file(s).

A box below the skeleton program contains the output from a sample run.

Contact your Instructor or GA's for any additional explanation you might need.

REQUIREMENTS:
- Write and document a complete C program that is capable of satisfying the requirements of this assignment problem.
- UNDOCUMENTED OR IMPROPERLY DOCUMENTED code will automatically lose 50% marks.
- PLAGIARIZED work will not be graded and receive a mark of ZERO and reported according to the Senate bylaws.
- The question requires use of I/O redirection. Please review the textbook for an example on using I/O redirection from flat files.
- TO SUBMIT: No later than the submission deadline, your email must be received by: cs14101@uwindsor.ca, late submissions are not accepted and will receive a mark of ZERO.
- Email your work as an attachment, include both the source file (assign1.c) and the script file (assign1.txt) - see below how to create the script file.

To create a script file (one that logs your compilation steps and your output in a text file):
1. script assign2.txt
2. cat assign2.c
3. cat input.txt
4. cc assign2.c
5. a.out < input.txt
6. ls -l
7. exit (DO NOT FORGET THIS STEP!!)

Email both files with the mail Subject field format " **Ass #2 section 51**" (replacing 51 by your actual registered lab section number) to:

cs141@courses.cs.uwindsor.ca

NOTE: Submissions that are not received correctly by the deadline will automatically receive a ZERO mark. In the event that more than one email submission is sent, only the last one (according to the date and time stamp) will be marked.

It is your responsibility to ensure the email attachment is sent correctly (readable) and to the right mailbox by the deadline. If you omit the email subject header or fail to follow the format provided above for the attachment file, your assignment may not be graded.

Late assignment submissions are not accepted!

NOTES:
1. Your assignment must be RECEIVED by the due date and time. Late assignment submissions are NOT accepted. Keep your script file, and all your code unmodified as proof of its completion in case it is not received.

2. It is your responsibility to get an early start on the assignment, research and ask questions ahead of time from the due date.

3. You must use your uwindsor email account to submit your work. Please do not use other email accounts (hotmail, yahoo etc...) for the purpose of assignment submissions.

4. Marks will be deducted for unclear code. (improper spacing and alignment, hard to read programs and missing outputs).

5. Make sure you turn in a complete script file that clearly shows: your code, your compilation process, a listing of the directory showing your source file(s) and the a.out with the date/time stamps, and the output.

6. **PLAGIARISM:** CHEATING IS NOT TOLERATED. You must submit your own work. Students who are suspected of copying someone else's work will be reported to the department's chair and the Dean of Science and be dealt with in accordance with the University policies. You should not share your code with others. Codes that are similar to each other will BOTH be reported as potential evidence of copying. It is imperative that you write your own code.

7. Authorized/limited help on this assignment may be provided directly from your Lecture or Lab instructors and Teaching Assistants.

/** Partially completed data encryption program “Assign2.c” **/

```
/** TODO: Documentations **/
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<time.h>

#define SIZE 256

/** Function Prototypes **/
void generate_key( int array[], int size );
void encrypt_data( char out[], const char in[], const int key[], int size );
void decrypt_data( char out[], const char in[], const int key[], int size );

int main() {

        char inputStr[SIZE]; char
        encrypt[SIZE]; char
        decrypt[SIZE]; int
        key[SIZE];
        int i, size;

        printf( "Enter a string to encrypt: " );
        scanf( " %[^\n]s", inputStr ); // to scan a sentence from the keyboard

        size = strlen( inputStr ); // strlen is a library function defined in
                                                // string.h to find the length of a string

        printf( "Generating key...\n" );
        generate_key( key, size ); // should randomly generate unique keys

        /** TODO: Print keys **/



        printf( "Encrypting string...\n" );
        encrypt_data( encrypt, inputStr, key, size ); // encrypt a string

          /** TODO: Print encrypted string **/



        printf( "Decrypting string...\n" );
        decrypt_data( decrypt, encrypt, key, size ); // decrypt a string

          /** TODO: Print decrypted string **/

          return 0;
}
```

```
/** TODO: Documentations **/
void generate_key( int array[], int size ){

        srand( time( NULL ) );

        /** TODO: Define function generate_key.
              REMEMBER: Each key (integer) MUST be unique! **/
}



/** TODO: Documentations **/
void encrypt_data( char out[], const char in[], const int key[], int size ){

        /** TODO: Define function encrypt_data **/
}



/** TODO: Documentations **/
void decrypt_data( char out[], const char in[], const int key[], int size ){

        /** TODO: Define function decrypt_data **/
}

/* END OF PROGRAM LISTING */
```

A Sample run of the program:

```
$ gcc Assign2.c
$ a.out

Enter a string to encrypt: one ring to rule them all
Generating key...
Key = 18 6 17 19 23 21 3 0 9 12 7 8 11 22 4 14 16 1 15 2 24 10 13 5 20
Encrypting string...
Encrypted string is "hntel otrg  arl neelouim"
Decrypting string...
Decrypted string is "one ring to rule them all"
```