ICS 103, Term 103

Computer Programming in C

HW# 3 Solution
Due date: Saturday, July 30, 2011

Q.1. You are required to write a program to read a text file and encrypt it and store the encrypted text into another file. The same program can be used to decrypt the encrypted file. Encryption/decryption is performed according to a password entered by the user. To implement your program, do the following:

(i) Write a function that receives an integer value and returns a value between 0 and 31. The digit is computed by adding twice the digits in even positions and adding the digits in odd positions of the entered number and if the result is greater than 31, then the process is repeated until the result becomes within the range [0,31]. For example, if the integer is 918, then the program computes 2*9+1+2*8=35>31. Then, the process is repeated and 3+2*5=13. Thus, the function will return the value 13.

(ii) Ask the user to enter the file name to encrypt/decrypt and read it. Then, ask the user to enter the output file name and read it. You can define the input and output file names as char infname[40], outfname[40]. Open the input file for reading and the output file for writing handling file not found error.

(iii) Ask the user to enter a password as a string of characters and read it. Assume that the password can be a maximum of 100 characters. Write a function that receives a string of characters and returns an integer which is the sum of the ASCII code of these characters. Initialize the random number generator in the beginning of your program by the returned value, pwdvalue, of this function using the function srand(pwdvalue).

(iv) Read each character in the input file and encrypt/decrypt it if it is not a new line character. Save the encrypted/decrypted character in the output file. A new line character should be written as is. Encryption/decryption of each character is performed by generating a random number N and using the function developed in (i) with N as input to generate a number M in the range [0,31]. Then a character C is encrypted as C XOR M (XOR operation is ^). You can generate a random number using the function rand().

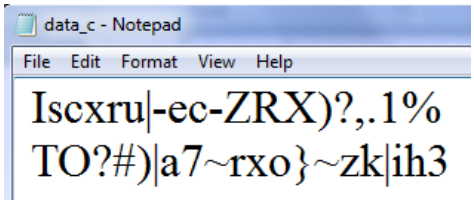
(v) Test the correctness of your program by encrypting and then decrypting the message given below using the password HelloICS103:
"Welcome to ICS 103!!
HW#3 is interesting."

Sample executions of the program are shown below:

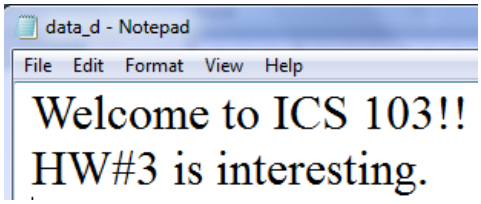
```
Enter the input file name to encrypt/decrypt: data.txt
Enter the output file name: data_c.txt
Enter your password: HelloICS103
Encryption/Decryption process completed ...
Press any key to continue . . .
```

```
Enter the input file name to encrypt/decrypt: data_c.txt
Enter the output file name: data_d.txt
Enter your password: HelloICS103
Encryption/Decryption process completed ...
Press any key to continue . . .
```

Encrypted file:



Decrypted file:




```
#include <stdio.h>
#include <stdlib.h>

int convert(int n);
int pwd_to_int(char pwd[]);

int main(void){

int pwdvalue, N, M;
FILE *inf, *outf;
char c;
char infname[40], outfname[40], pwd[100];


 printf("Enter the input file name to encrypt/decrypt: ");
 scanf("%s",infname);

 inf = fopen( infname, "r");
 if (inf == NULL){
   printf("Cannot open %s for reading \n", infname);
   exit(1);
 }
```

```
  printf("Enter the output file name: ");
  scanf("%s",outfname);
  outf = fopen( outfname, "w");

  printf("Enter your password: ");
  scanf("%s",pwd);
  pwdvalue=pwd_to_int(pwd);
  printf("pvalue=%d\n",pwdvalue);
  srand(pwdvalue);

  while (  fscanf(inf,"%c",&c) != EOF ) {
     if ( c  !=  '\n'  ){
        N = rand();
        M = convert(N);
        c ^= M;
     //   printf("ch=%c\tM=%d\tN=%d\n",c,M, N);
     }
     fprintf(outf,"%c",c);
  }

  printf("Encryption/Decryption process completed ...\n");

    system("pause");
    return 0;
}

int convert(int n){

    int i, r, sum;

    while (n>31){
        sum = 0; i=0;
        do {
            r = n%10;
            n = n/10;
            if (i%2) sum += r;
            else sum += 2*r;
            i++;
           } while (n !=0);
        n = sum;
    }

    return n;
}

int pwd_to_int(char pwd[]){

    int i=0, sum=0;

    while (pwd[i] != 0){
      sum += pwd[i];
      i++;
    }

    return sum;
}
```