

Fernando Gonzalez

EGN 3211

HW # 1

Spring 2006  
(T,R 7:30 AM)

This program is used to encrypt files using an n-byte code sequence. The program receives n bytes from the command line plus a file name. This n-byte sequence code is used to perform an exclusive or (XOR) operation of the first n bytes of the file. It uses random access file technology to modify only the first n bytes of the file. The user may un-encrypt the file by repeating the process with the same code sequence. This program teaches how to use bit operations on C, how to use random access files, and how to receive information through the DOS command line.

The program was run using several PDF and BMP files and using a code of 100. The file was unable to be opened after the encryption but was restored to its original state after re-encrypting. The Word format files however were unable to return to their original state. I am not sure why.

```

////////////////////////////////////
//
// Title           Program 7.3
//
// Author:        Fernando Gonzalez
//
// Description:    This program is used to encrypt files using a code. The program receives an
//                N bytes from the command line. This n-byte sequence code is used to performe
//                an exclusive or (XOR) operation of the first N bytes of the file. It uses
//                random access file technology to modofy only the first N bytes. The user
//                may un-encrypt the file by repeating the process with the same code sequence.
//
//
// 5/15/1999      Created the program
// 5/24/1099     Changed the variable code to tyoe char so that the value is not signed exteded.
// 1/26/2006     Added comments to the code.
//
// Input:         The name of the file to encrypt.
//                The N byte code sequence
// Output:        The file.
//
// Assumptions:   The code sequence is less than 10 bytes (N < 10)
//                Each byte in the code need to be in the range (0 to 126)
//                The file is more then N bytes long.
//                The input is given through the command line.
//
////////////////////////////////////

```

```

#include "stdafx.h"

#include "stdio.h"
#include "stdlib.h"

```

```

#define FALSE 0
#define TRUE  !FALSE

```

```

char    buffer[10];
FILE    *fp;
int     i;
char    code;

```

```

void
main(   int    argc,
        char   *argv[])

```

```

{
  if (argc < 3)
  {
    printf("format: code bill.dat 100 \n");
    printf("Where bill.dat is the file to code and 100 is the code (1 to 126) \n");
    return;
  }

```

```

  code = (char)atoi(argv[2]);

```

```

  if ( (code > 126) || (code < 1))
  {
    printf("code is out of range\n");
    printf("format: code bill.dat 100 \n");
    printf("Where bill.dat is the file to code and 100 is the code (1 to 126) \n");
    return;
  }

```

```

  fp = fopen(argv[1],"r+");
  if (fp == NULL)
    printf("Error reading %s \n",argv[1]);
  else

```

```

  {
    fseek(fp,0,SEEK_SET);
    fread(buffer,sizeof(buffer),1,fp);

    for (i = 0; i < sizeof(buffer) - 1; i++)

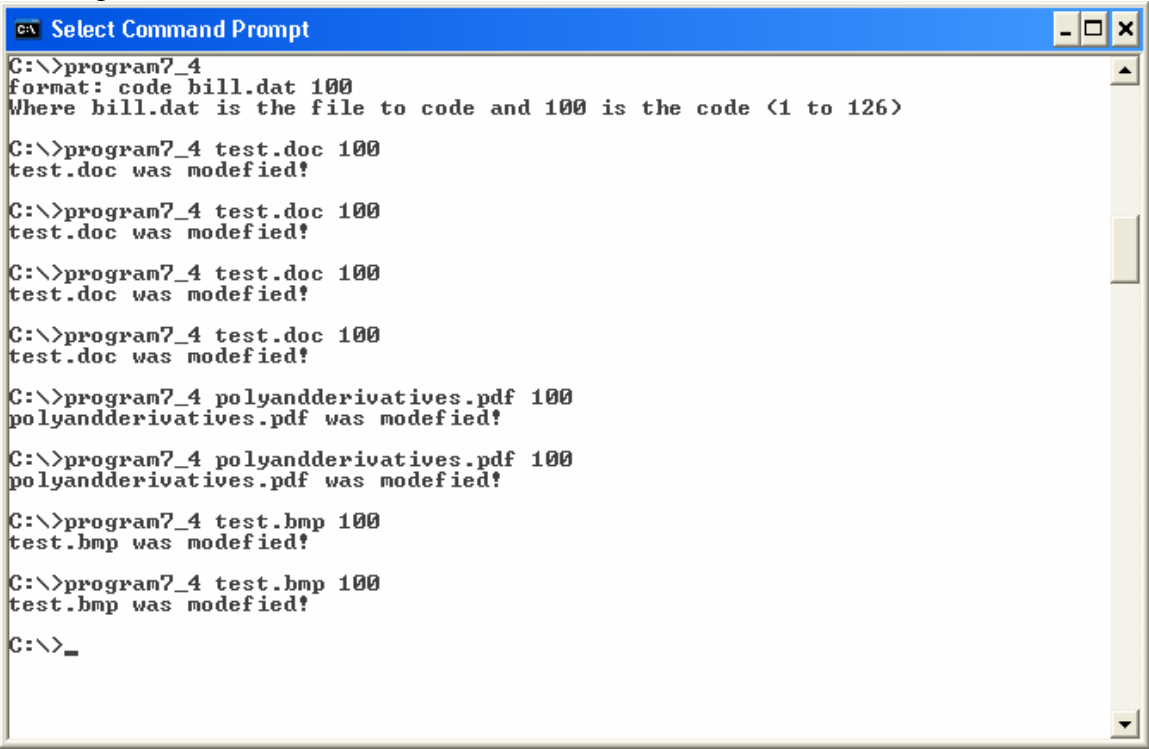
```

```
        buffer[i] = buffer[i] ^ code;

fseek(fp,0,SEEK_SET);
fwrite(buffer,sizeof(buffer),1,fp);
printf("%s was modified!\n",argv[1]);
}

fclose(fp);
}
```

The output shown with several runs:



```
C:\>program7_4
format: code bill.dat 100
Where bill.dat is the file to code and 100 is the code <1 to 126>

C:\>program7_4 test.doc 100
test.doc was modified!

C:\>program7_4 test.doc 100
test.doc was modified!

C:\>program7_4 test.doc 100
test.doc was modified!

C:\>program7_4 test.doc 100
test.doc was modified!

C:\>program7_4 polyandderivatives.pdf 100
polyandderivatives.pdf was modified!

C:\>program7_4 polyandderivatives.pdf 100
polyandderivatives.pdf was modified!

C:\>program7_4 test.bmp 100
test.bmp was modified!

C:\>program7_4 test.bmp 100
test.bmp was modified!

C:\>_
```