

Description

Write a program that will encrypt and decrypt integer data.

Requirements

- Part of the project's purpose is to practice using / and *. Thus, you must get the number as an integer (not a string) and break the entered number apart with / and % and piece back together again with * and +
For example,
 $7234 / 1000 = 7$ and $7234 \% 1000 = 234$ (the remainder to work with)
Also, for example,
 $8 * 100 + 5 * 10 + 3 * 1 = 853$ (note that multiplying by 1 would not be needed)
- Follow all coding guidelines (all comments required) and break the code within Main into paragraphs of code each preceded with a blank line and a comment. Remember named constants for numbers other than 0, 1, or 2; however, you are not required to create named constants for powers of 10 in this program (e.g. 10, 100, 1000)
- Write code according to the directions given in problem 5.41 on p.188 in the textbook. However, rather than writing two separate programs, you will combine them into one.
 - First, encrypt by prompting for and retrieving a non-negative integer from the user.
 - Then create and display the new encrypted **integer**. **Note the D4 format specifier stated in the problem. You may not just display digits, but create a whole new integer to display.**
e.g. `Console.WriteLine("Encrypted Integer: {0:D4}", encryptedData);`
 - Second, decrypt by prompting for and retrieving a non-negative integer from the user.
 - Then create and display the new decrypted integer. **Note the D4 format specifier stated in the problem. You may not just display digits, but create a whole new integer to display.**
e.g. `Console.WriteLine("Decrypted Integer: {0:D4}", decryptedData);`
- In addition to what the textbook problem states, you must account for a negative integer entered for either the encryption or decryption phases. Do this by continually prompting the user for a non-negative integer until a valid integer is entered. A Do-While loop is helpful here.
- You may assume that the user always enters at least a 1 digit number but no more than 4.
- Match the input prompts and output shown in the sample runs.
- If you are already familiar with methods, you may optionally break your code into static methods such as Main, Decrypt, Encrypt.
- For projects, I don't give as much help as labs, but I do help clarify the problem as needed. You may work with one other person in the class provided that you both work on it equally. If doing this, turn in only one project for your team – submit it to one of your drop boxes and hand in a printed version with both names after @author.

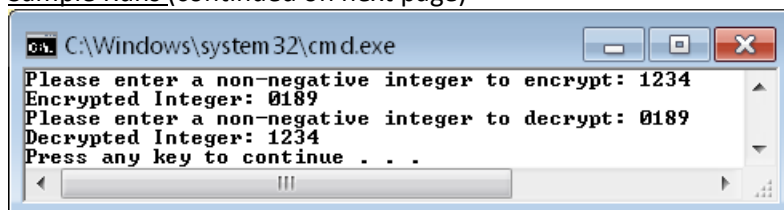
Submission

Before class:

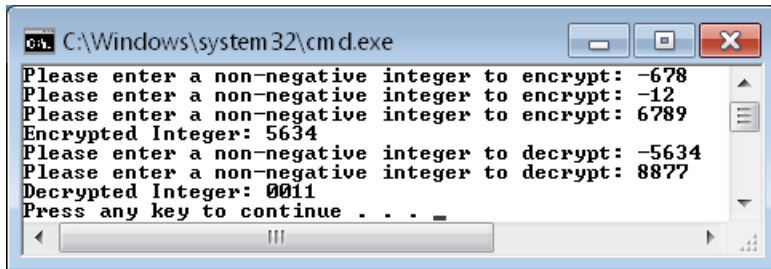
- Print the source code (.cs file)
- Submit the source code to the designated drop box in D2L

Beginning of class:

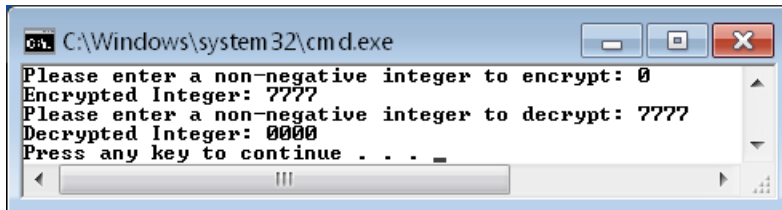
- Turn in the source code

Sample Runs (continued on next page)

```
C:\Windows\system32\cmd.exe
Please enter a non-negative integer to encrypt: 1234
Encrypted Integer: 0189
Please enter a non-negative integer to decrypt: 0189
Decrypted Integer: 1234
Press any key to continue . . .
```



```
C:\Windows\system32\cmd.exe
Please enter a non-negative integer to encrypt: -678
Please enter a non-negative integer to encrypt: -12
Please enter a non-negative integer to encrypt: 6789
Encrypted Integer: 5634
Please enter a non-negative integer to decrypt: -5634
Please enter a non-negative integer to decrypt: 8877
Decrypted Integer: 0011
Press any key to continue . . .
```



```
C:\Windows\system32\cmd.exe
Please enter a non-negative integer to encrypt: 0
Encrypted Integer: 7777
Please enter a non-negative integer to decrypt: 7777
Decrypted Integer: 0000
Press any key to continue . . .
```

A company that wants to send data over the Internet has asked you to write a program that will encrypt it so that it may be transmitted more securely. All the data is transmitted as four-digit integers. Your app should read a four-digit integer entered by the user and encrypt it as follows: Replace each digit with the result of adding 7 to the digit and getting the remainder after dividing the new value by 10. Then swap the first digit with the third, and swap the second digit with the fourth. Then display the encrypted integer. Write a separate app that inputs an encrypted four-digit integer and decrypts it (by reversing the encryption scheme) to form the original number. Use the format specifier D4 to display the encrypted value in case the number starts with a 0.