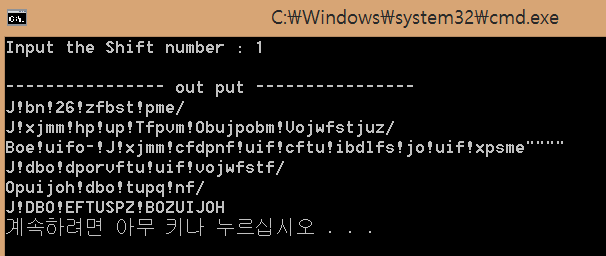
1. Project Description of 1

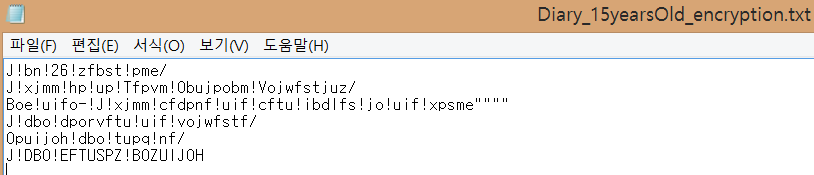
The first problem with the second task is to invoke, encrypt, and create and save other files with encrypted characters.

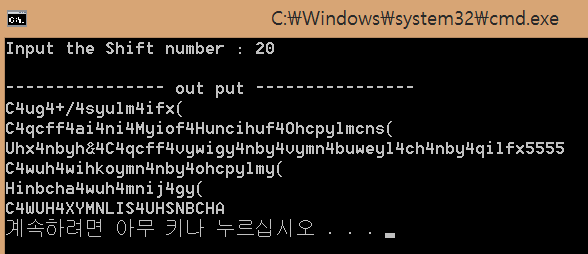
When you create a file to import information into the path you specify, use the File I/O function to store the information in a work environment variable, then insert the variable into a function to encrypt the information.

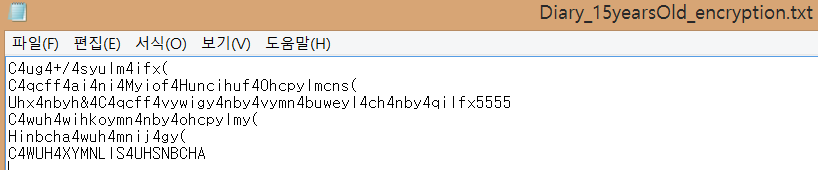
As an exception handling method, we have specified the range from 0 to 26 for encryption as there are 26 English alphabets, and if the numbers do not match, we can inform you that they are out of range and allow you to reprogram the integer programs.

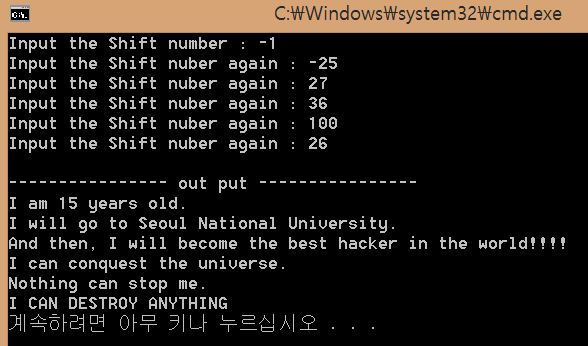
1. Action Results











1. Review

Designing the first task took a long time to understand how to read and save the file as it was normally written without the use of the file I/O function.

While designing the encryption function, we also found that if we exceeded the ASCII value, then the odd characters would be printed, and we could not exceed the error range for this issue. Finally, while learning about the File I/O function, it is important to learn that if a file fails to open normally, the exception handling for the one that did not open, and to close the open files before exiting the function.

1. Project Description of 2

The second issue is a program that sets the original time, checks the value for that time, adds the time you want to add, and gives a visible indication of the final time value.

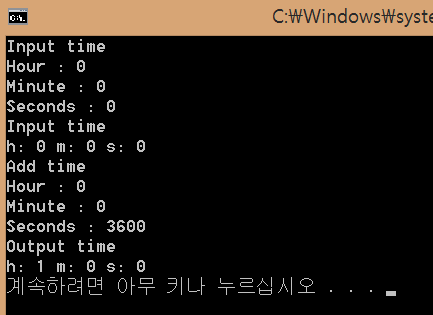
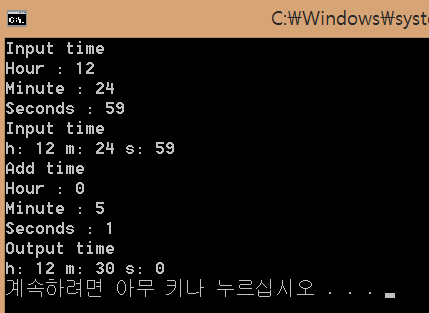
We designed the code using the function you specified, giving us the function to use in the problem, and the number and shape of variables that would fit in it.

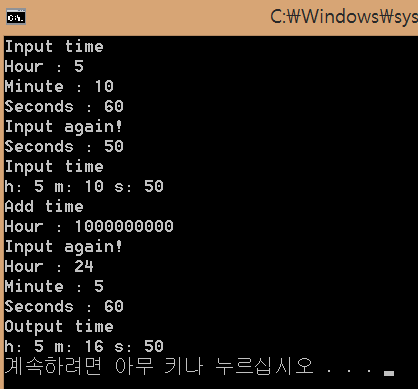
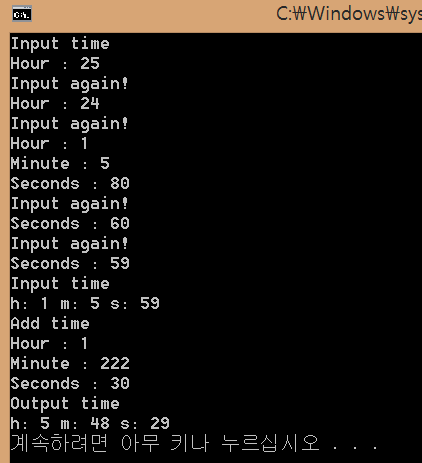
Using one function that adds and the main function are enough to solve the problem.

For exception handling, the first-time set is less than 24 hours, and the second is scoped less than 60 seconds.

The time, minute, and seconds available to add were later set to be smaller than six wins in 10, and when that was exceeded, they were told that the range was exceeded, so they could be redesignated.

1. Action Results





1. Review

I felt like I could solve this task most comfortably. It is designed to be received.

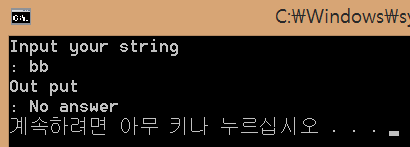
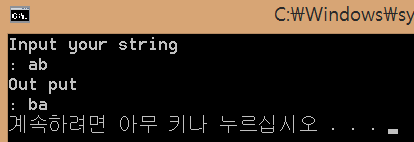
The reason is that the use and application of the pointer is seen as the purpose of the task, especially since the first grade, it has been very useful in conveying and receiving values between important and functions.

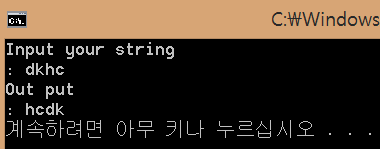
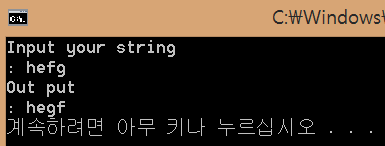
However, since we haven't used a lot of file I/O from question 1, for instance, we don't seem to be able to see the challenge with these differences.

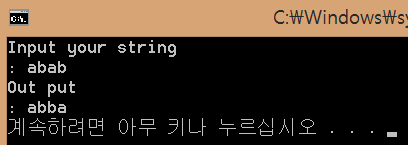
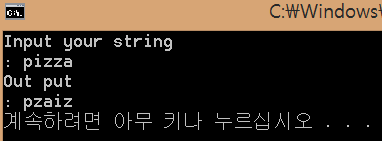
1. Project description of 3

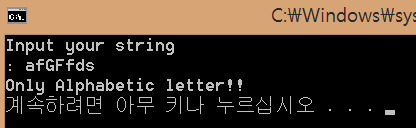
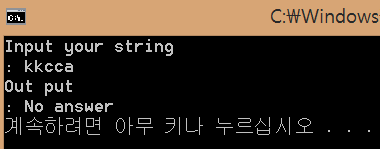
The third challenge is pre-alignment. In short, it's called a net column, and it makes sure you print the next character you type in alphabetical order. If the length of the character is set to 10, it will alert you to exceed the range when the length exceeds 10, and the function is designed to be terminated. It is also designed to terminate the function by displaying the phrase " No answer " when the arrangement is no longer required. Also, if you have non-alphabetic characters, “just put in lowercase letters!” With the phrase, and we designed it to end the function.

1. Action Results









1. Review

It was a difficult task for me personally.

First, it took a long time to implement a function that results in the next permutation output. We solved the problem by modifying the code carefully during debugging because the code was messy when handling exceptions.

While solving the task, we also discovered a header file called the algorithm library, and we also found that there is a function that outputs the next net column, which is most important for troubleshooting problems.

But I was told by my assistant that using that function to do the task would be out of line. We designed our own functions that give us the next permutation.

As you work through the task, you'll get more familiar with the arrangement and find that many header files have convenient functions.

Getting a job was a lot of work.

1. Project Description of 4

The fourth challenge is to print programs that print binary values that are set in alphabetical order.

As a major characteristic of this task, I wrote about the class I didn't write, and it is a starting point for me to learn about it for the first time.

The operation of the program is simple to learn.

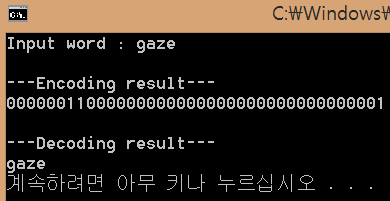
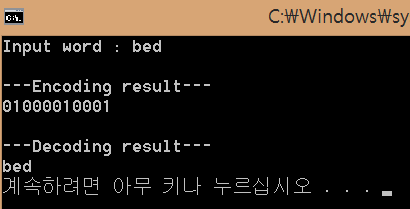
When you insert the alphabet you want, you print the binary value that corresponds to it, and then convert it back into the alphabet with the binary value you've printed.

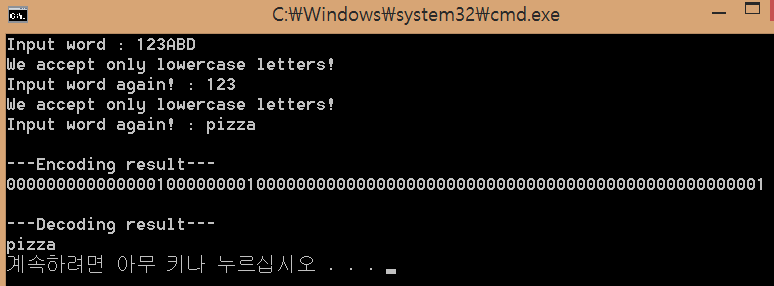
It is designed to design programs and to receive only small English characters as exceptions.

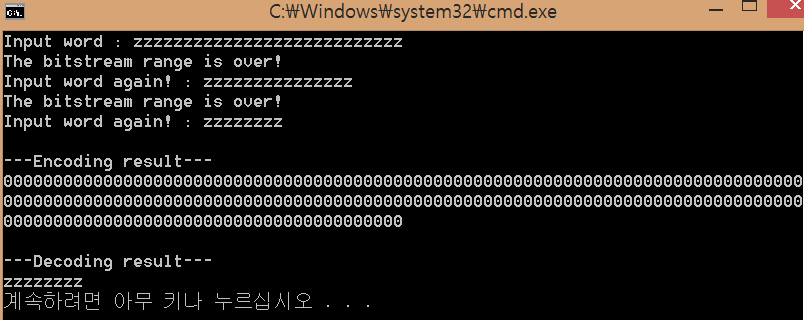
It is designed to receive text messages again if other characters or uppercase letters are included.

Because the buffer is 255 in size, it gives you a visible indication that an error occurred when you exceed the number during the binary translation process and is designed to receive characters again.

1. Action Results







1. Review

I wasn't used to writing classes, so I got lost.

At first, when the hint code was not raised, I completed the code by simply declaring the class to the working environment and using the structure as usual.

With the hint code, I learned how to make the class more efficient and clean the main function side.

We applied the code we originally designed to make the class more efficient and clean the main function part.

It was a task that allowed me to become familiar with the differences between structures and the use of the classes.

The bold exception to the design process was the problem with printing ' z ', which solved the problem by designing other actions when the character was ' z '.

1. Project Description of 5

The fifth task is related to the third matrix.

If you used the classes to design the program, as you would in the fourth task, you designed the program to fit the form.

As a difference to task 4, it was different to declare the operator required for the computation between the classes.

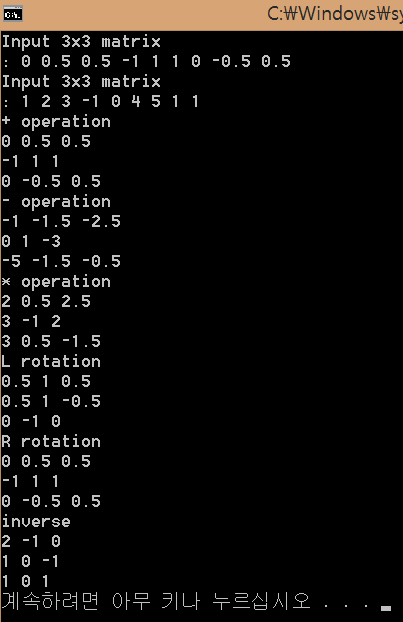
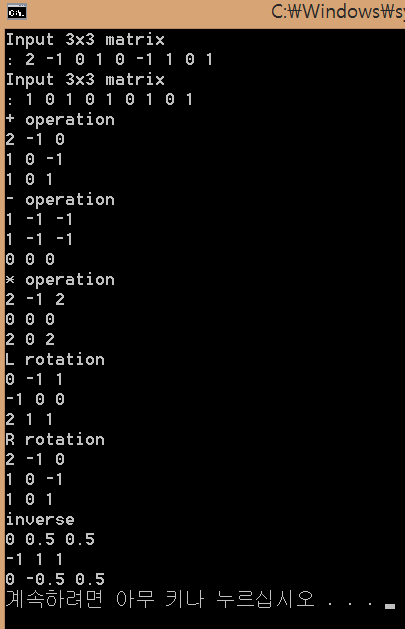
For the behavior of the program, if you enter the value of the desired third row, you automatically turn left for addition, subtraction, multiplication, and the first third row you entered, and then finally find the inverse matrix.

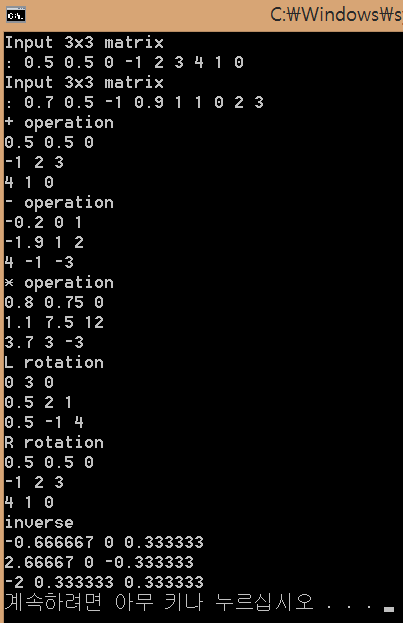
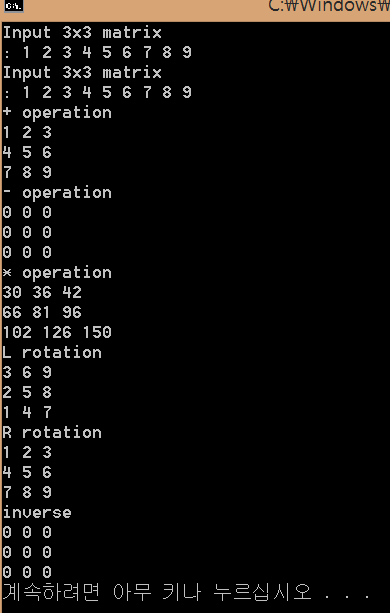
In case of exception handling, if the denominator is zero, then no significant exceptions are used when the task is submitted except for entering zeros in all inverse functions.

There was no noticeable exception handling when the task was submitted, so I paid much attention to it and focused on the use of the class.

복사하기

1. Action Results





1. Review

As with task number 4, it was a task to design a program using a class.

It was not that difficult to design a functional function that produced the relevant additions, subtraction, multiplication, and right turns of the matrix while designing the reverse trip curves. However, in the operation between the class and \* this refers to when the class is rewritten, the designer must declare the operator directly and define the operator, that is reason that I felt difficult In particular, we had too much time for understood where the variable used in the function to define the operator came from, and it took some time to define the operator when it was reset. As with question 4, it was a task to become familiar with the class and to learn about defining the operator.