Problem: Caesar Cipher

Julius Caesar protected his confidential information by encrypting it in a cipher. Caesar's cipher rotated every letter in a string by a fixed number, , making it unreadable by his enemies. Given a string, , and a number, , encrypt  and print the resulting string.

**Note:** The cipher *only* encrypts letters; symbols, such as -, remain unencrypted.

**Input Format**

The first line contains an integer, , which is the length of the unencrypted string.   
The second line contains the unencrypted string, .   
The third line contains the integer encryption key, , which is the number of letters to rotate.

**Constraints**   
   
   
 is a valid ASCII string and doesn't contain any spaces.

**Output Format**

For each test case, print the encoded string.

**Sample Input**

11

middle-Outz

2

**Sample Output**

okffng-Qwvb

**Explanation**

Each unencrypted letter is replaced with the letter occurring  spaces after it when listed alphabetically. Think of the alphabet as being both case-sensitive and circular; if  rotates past the end of the alphabet, it loops back to the beginning (i.e.: the letter after  is , and the letter after  is ).

**Selected Examples:**   
 (ASCII 109) becomes  (ASCII 111).   
 (ASCII 105) becomes  (ASCII 107).   
 remains the same, as symbols are not encoded.   
 (ASCII 79) becomes  (ASCII 81).   
 (ASCII 122) becomes  (ASCII 98); because  is the last letter of the alphabet,  (ASCII 97) is the next letter after it in lower-case rotation.

Solution

int main() {

string str;

int length, offset;

cin>>length >>str >>offset;

int temp=0;

for(int i=0; i<length; i++)

{

if( (int)str[i]>=65 && (int)str[i]<=90 )

{

temp=(int)str[i]+(offset%26);

temp=(temp>90 ? 64+(temp%90) : temp);

str[i]=(char)temp;

}

else if( (int)str[i]>=97 && (int)str[i]<=122 )

{

temp=(int)str[i]+ (offset%26);

temp=(temp>122 ? 96+(temp%122) : temp);

str[i]=(char)temp;

}

}

cout<<str;

return 0;

}

`’Anshul Aggarwal