# DNA

One interesting use of computer is to analyze biological data such as DNA sequences. Biologically, a strand of DNA is a chain of nucleotides Adenine, Cytosine, Guanine, and Thymine. The four nucleotides are represented by characters `A`, `C`, `G`, and `T`, respectively. Thus, a strand of DNA can be represented by a string of these four characters. We call such a string a *DNA sequence*.

It is possible that the biologists cannot determine some nucleotides in a DNA strand. In such a case, the character `N` is used to represent an unknown nucleotides in the DNA sequence of the strand. In other words, `N` is a wildcard character for any one character among `A`, `C`, `G` or `T`. We call a DNA sequence with one or more character `N` an *incomplete sequence*; otherwise, it is called a *complete sequence*. A complete sequence is said to *agree with* an incomplete sequence if it is a result of substituting each `N` in the incomplete sequence with one of the four nucleotides. For example, `ACCCT` agrees with `ACNNT`, but `AGGAT` does not.

Researchers often order the four nucleotides the way we order the English alphabets: `A` comes before `C`, `C` comes before `G`, `G` comes before `T`. A DNA sequence is classified as *form*-1 if every nucleotide in it is the same as or comes before the nucleotides immediately to its right. For example, `AACCGT` is form-1, but `AACGTC` is not.

In general, a sequence is *form-$j$*, for $j > 1$, if it is a form-$(j-1)$ or it is a concatenation of a form-$(j-1)$ sequence and a form-1 sequence. For example, `AACCC`, `ACACC`, and `ACACA` are form-3, but `GCACAC` and `ACACACA` are not.

Again, researchers order DNA sequences lexicographically the way we order words in a dictionary. As such, the first form-3 sequence of length 5 is `AAAAA`, and the last is `TTTTT`. As another example, consider the incomplete sequence `ACANNCNNG`. The first seven form-3 sequences that agree with it are:

$$\begin{gathered}
\texttt{ACA}\underline{\texttt{AA}}\texttt{C}\underline{\texttt{AA}}\texttt{G}\\
\texttt{ACA}\underline{\texttt{AA}}\texttt{C}\underline{\texttt{AC}}\texttt{G}\\
\texttt{ACA}\underline{\texttt{AA}}\texttt{C}\underline{\texttt{AG}}\texttt{G}\\
\texttt{ACA}\underline{\texttt{AA}}\texttt{C}\underline{\texttt{CA}}\texttt{G}\\
\texttt{ACA}\underline{\texttt{AA}}\texttt{C}\underline{\texttt{CC}}\texttt{G}\\
\texttt{ACA}\underline{\texttt{AA}}\texttt{C}\underline{\texttt{CG}}\texttt{G}\\
\texttt{ACA}\underline{\texttt{AA}}\texttt{C}\underline{\texttt{CT}}\texttt{G}
\end{gathered}$$

## Task

Write a program to find the $R$th form-$K$ sequence that agrees with the given incomplete sequence of length $M$.

## Input

The first line contains three integers separated by one space: $M (1 \le M \le 50{,}000)$, $K (1 \le K \le 10)$, and $R$ $(1 \le R \le 2 \times 10^{12})$. The second line contains a string of length $M$, which is the incomplete sequence. It is guaranteed that the number of form-$K$ sequences that agrees with the incomplete sequence is not greater than $4 \times 10^{18}$, so it can be represented by a `long long` in C and C++ or an `Int64` in Pascal. Moreover, $R$ does not exceed the number of form-$K$ sequences that agree with the given incomplete sequence.

## Output

On the first line, print the $R$th form-$K$ sequence that agrees with the incomplete sequence in the input.

## Sample Input 1

```
9 3 5
ACANNCNNG
```

## Sample Output 1

```
ACAAACCCG
```

## Sample Input 2

```
5 4 10
ACANN
```

## Sample Output 2

```
ACAGC
```

## Programming Remark

In C and C++, you should use `long long` data type. The following piece of code show how to read and write a `long long` from and to standard input/output:

```
long long a;
scanf("%lld",&a);
printf("%lld\n",a);
```

In Pascal, you should use `Int64`. No special instructions are needed to manipulate data of this type.

## Time and Memory Limits

Your program must terminate in 1 second and use no more than 128 MB of memory.

## Scoring

The score for each input scenario will be 100% if the correct answer is outputed and 0% otherwise.

In test scenarios worthing 20 points, $M$ will be at most 10.