Subtask1:

In DNA strings, symbols 'A' and 'T' are complements of each other, as are 'C' and 'G'. Given a nucleotide *p*, we denote its complementary nucleotide as *c*. The reverse complement of a DNA string *Pattern* = $p_1 \ldots p_n$ is the string *Pattern* = $c_n \ldots c_1$ formed by taking the complement of each nucleotide in *Pattern*, then reversing the resulting string.

For example, the reverse complement of *Pattern* = "GTCA" is *Pattern* = "TGAC".

Reverse Complement Problem

Find the reverse complement of a DNA string.

Given: A DNA string *Pattern*.

Return: *Pattern*, the reverse complement of *Pattern*.

Sample Dataset

```
AAAACCCGGT
```

Sample Output

```
ACCGGGTTTT
```

Subtask2:

We say that position *i* in *string* $p_1 \ldots p_k$ and $q_1 \ldots q_k$ is a **mismatch** if $p_i \neq q_i$. For example, CG**A**A**T** and CG**G**A**C** have two mismatches. The number of mismatches between strings *p* and *q* is called the Hamming distance between these strings and is denoted *HammingDistance*(*p*, *q*).

Hamming Distance Problem

Compute the Hamming distance between two DNA strings.

Given: Two DNA strings.

Return: An integer value representing the Hamming distance.

Sample Dataset

```
GGGCCGTTGGT

GGACCGTTGAC
```

Sample Output

```
3
```

Subtask3:

We say that a *Pattern* appears as a substring of *Text* with at most *d* mismatches if there is some substring *Pattern'* of *Text* having *d* or fewer mismatches with *Pattern*,

Approximate Pattern Matching Problem

Find all approximate occurrences of a pattern in a string.

Given: Strings *Pattern* and *Text* along with an integer *d*.

Return: All starting positions where *Pattern* appears as a substring of *Text* with at most *d* mismatches.

Sample Dataset

```
ATTCTGGA

CGCCCGAATCCAGAACGCATTCCCATATTTCGGGACCACTGGCCTCCACGGTACGGACGTCAATCAAATGCCTAGCGGCTT
GTGGTTTCTCCTACGCTCC

3
```

Sample Output

```
6 7 26 27 78
```

Subtask4:

Frequent Words with Mismatches Problem

Consider pattern P, it could be consist of ***('A', 'C', 'G', 'T')*** and the length would be k.

Find all the possible most frequent P with d mismatches in a the Text.

Given: A string *Text* as well as integers *k* and *d*.

Return: All possible most frequent *P* with up to *d* mismatches in *Text*.

Sample Dataset

```
ACGTTGCATGTCGCATGATGCATGAGAGCT

4 1
```

Sample Output

```
GATG ATGC ATGT
```

Subtask5:

Frequent Words with Mismatches and Reverse Complements Problem

*Consider pattern P, where it could be consist of **('A', 'C', 'G', 'T')*** and the length would be k.

Find all the possible most frequent P with d mismatches and reverse complements in the Text.

Given: A string *Text* as well as integers *k* and *d*.

Return: All possible most frequent *P with d mismatches and reverse complements* in *Text*.

Sample Dataset

```
ACGTTGCATGTCGCATGATGCATGAGAGCT
4 1
```

Sample Output

```
ATGT ACAT
```

Subtasks6:

Implement PatternToNumber

Convert a DNA string to a number.

Given: A DNA string *Pattern*.

Return: *PatternToNumber*(*Pattern*).

Sample Dataset

```
AGT
```

Sample Output

```
11
```

Subtask7:

Implement NumberToPattern

Convert an integer to its corresponding DNA string.

Given: Integers *index* and *k*.

Return: *NumberToPattern*(*index*, *k*).

Sample Dataset

```
45
4
```

Sample Output

```
AGTC
```