Useful functions

1. **Ignore k most significant digits in num n**

```
long ignore_first_k_digit(long n, long k)
{
  long power = 1;
  for (int i = 0; i < k; i += 1) {
    power *= 10;
  }
  return n % power;
}
```

2. **Ignore the k least significant digits in num n**

```
long ignore_last_k_digit(long n, long k)
{
  long power = 1;
  for (int i = 0; i < k; i += 1) {
    power *= 10;
  }
  return n / power * power;
}
```

3. **The recursive way to calculate the weight we need to put a digit at the front of the num n in decimal.**

```
long power(long num)
{
  if (num < 10)
  {
    return 1;
  }
  return 10 * power(num / 10);
}
```

4. **The greatest common divisor, which is also known as the greatest common factor.**

```
long gcd(long a, long b)
{
  if (b == 0)
  {
  return a;
  }
  return gcd(b, a % b);
}
```

5. **The least common multiple (lcm)**

```
long lcm(long a, long b, long gcd)
{
  return a * b / gcd;
}
```

6. **The correct way to ceil a number**

```
long num_of_segments(long distance) {
  long segments = distance / 1000;
  if (distance % 1000 != 0) {
    segments += 1;
  }
  return segments;
```

```
}
```

7. **The iterative way to get number of digits of a given number**

```
long num_digits(long num)
{
  long count = 0;
  while (num != 0)
  {
    count += 1;
    num /= 10;
  }
  return count;
}
```

8. **A recursive way to find the number of zero in a given number**

```
long find_zero(long num)
{
  if (num == 0)
  {
    return 0;
  }
  long ans = 0;
  if (num % 10 == 0)
  {
    ans = 1;
  }
  ans += find_zero(num / 10);
  return ans;
}
```

Tips

1. In **math.h** library in C, there is a function called **fabs(x)**, which will return the absolute value of a floating-point argument **x**.
2. We may need the idea of getting **ϵ** when we want to judge whether two double are equal. However, if we only want to know which double is bigger enough, we can just use the operators >,<,≤,≥,, that's enough.
3. Always regard **real numbers** as **double** type variable!
4. Marks won't be deducted if you use **break/continue** statement correctly in the loop.
5. The use of **return** statement in the loop is allowed in the loop. It will cause the current stack frame to the function to be destroyed.
6. When getting the simple fraction, you don't need to find the least common multiple (LCM) at first, you can just form the new denominator as the multiple of the two denominators (it may not be the LCM) and using fundamental numeric operations to form a new numerator. Then **divide both the new denominator and the numerator by the gcd() of them.** You will get the simplest version.
7. Using a negative number modulo a positive number, the result is a negative number. i.e. −1234 % 10=−4. We call the % **remainder operator** instead of **module operator**!!! The % operator in C is defined as follows: **x % n** is equivalent to **x - ((x / n) * n**) (where **/** is the integer division operator).

8. Three basic ideas of using wishful thinking to solve a problem recursively:
    a. Assume, by wishful thinking, that we can solve this problem for a smaller n;
    b. Use the solution for smaller n to solve for the original n; and
    c. Solve this problem for the simplest n
9. For recursion that involves **numbers**, which may be considered a variant of something else in the problem, like "path", always treat them as **a whole number**, this will make your life much easier.

Vim Settings

```
" Turn on syntax highlighting
syntax on
" Make vim behaves less like vi
set nocompatible
" Keep a backup copy of the file being edited
set backup
" The Location of the backup files.
set backupdir=~/.backup
" Automatically indent the new line
set autoindent
" Indent the new line according to C-like syntax
set smartindent
" Display lines longer than the current window on the next line(s)
set wrap
" Prevent breaking a word into multiple lines when wrapping
set linebreak
" Display line and column number on the lower-right corner.
set ruler
" Replace all tabs with spaces
set expandtab
" Use 2 spaces for each indentation level
set sw=2
" Load the relevant plugins and indentation rules based on file types
filetype plugin indent on
" Enable 24-bit colors
set termguicolors
" Use gruvbox (dark) as default color scheme
color gruvbox
set background=dark
" set line number
set number
" set tab length
set tabstop=2
set shiftwidth=2
" To get around problem with seemingly random characters appear in certain
" teminals.
set t_RV=
set t_u7=
```