SOLUTIONS FOR 21ST OF JULY

ABSURD TRIANGLE

```
for(int i = 1; i <= n; i++) {
  for(int j = 1; j <= i; j++) {
    printf("\\");
  }
  printf("\n");
}
```

Note that in C++, to print the backslash, we have to write like ‘\\’ or 92 (ASCII code) because ‘\’ is an escape character.

BROKEN KEYPADS

First, let’s consider “012345” instead of “134579”. The sequence is like:

0, 1, 2, 3, 4, 5, 00, 01, 02, 03, 04, 05, 10, 11, 12, 13, 14, 15, ..., 54, 55,
000, 001, 002, ..., 553, 554, 555, 0000, 0001, 0002, ...

Separately look elements by their length:

- 0, 1, 2, 3, 4, 5 (6^1 numbers)
- 00, 01, 02, 03, 04, 05, 10, 11, 12, 13, 14, 15, 20, 21, 22, 23, 24, 25, ..., 50, 51, 52, 53, 54, 55 (6^2 numbers)
- 000, 001, 002, 003, 004, 005, 010, 011, 012, 013, 014, 015, ..., 549, 550, 551, 552, 553, 554, 555 (6^3 numbers)
- ...

As we can see, the elements are sorted by their value for each length. If we consider these numbers are written in base 6, we can see the numbers are consecutive.

So the answer is: $6^1 + 6^2 + \cdots 6^{length-1} + (\text{the value of the input if we read it in base } 6)$. We can just use the definition of bases to calculate the value of the input.

However, lots of students used 64-bit integer type to read n. Unfortunately, it caused your solution to be wrong. The statement says, “n is given so that the answer never exceeds 10^{18}”. It **DOESN’T** mean that $n \le 10^{18}$. For example, there was an input “39455715939991374579445” whose answer is “408767112702437025”. The length of the input could be larger than 18 because $6^{23} < 10^{18} < 6^{24}$.

So, you had to read n using a character array or a string, and then apply this algorithm. Read the constraints carefully!

Converting Coordinates

$\rho = \sqrt{x^2 + y^2 + z^2}$. We can calculate this value using the function `sqrt` and (maybe) function `pow`.

$\tan\varphi = \sqrt{x^2 + y^2}/z$. We can calculate φ using the function `atan`. However, the range of `atan` is $\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$, so we should add π if the returned value is negative. Also, don't forget to convert into degrees.

$\tan\theta = y/x$. We can also try to calculate θ using the function `atan`. However, the range of `atan` is $\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$, and the range of θ is $[0, 2\pi)$. The range is really different, so we cannot directly use the value of `atan`. The easiest approach is to consider four cases $x > 0, y > 0$ / $x > 0, y < 0$ / $x < 0, y > 0$ / $x < 0, y < 0$.

Also, there is a function called `atan2` that do all this job for you. We excluded from the lecture because our initial explanation of this function was wrong. Try to look at this reference and learn it correctly.

Problem in reading inputs

In this problem (and lots of other problems I've given to you), many students read the input like this:

```
while (true) {
  int x;
  cin >> x;
  if(x == -1) {
    break;
  }

  int y, z;
  cin >> y >> z;

  // ...
}
```

However, this is generally **NOT A GOOD IDEA**. We've seen at least one team that solved the problem correctly but got wrong answer because of this.

This code is assuming that if "-1" is given in the line, the input *must* terminate. However, this is not the case in the problem, because $-100 \le x, y, z \le 100$, inputs like "-1 5 4" could be given. An alternative is doing like this:

```
while (cin >> x >> y >> z) {
  ...
}
```

or

```
while (scanf("%d%d%d", &x, &y, &z) == 3) {
  ...
}
```

Dome

?

EASY CONNECTION

What do we want to compute for each plan? Let's write into a formula:

$$\sum_{i=a}^{b}\left(\sum_{j=c}^{d}|x_i - x_j|\right)$$

As $a \le b < c \le d$ and x is increasing, $x_j > x_i$ always holds. So let's get rid of the absolute value:

$$\sum_{i=a}^{b}\left(\sum_{j=c}^{d}(x_j - x_i)\right)$$

By the properties of $\sum$,

$$\sum_{i=a}^{b}\left(\sum_{j=c}^{d}x_j - \sum_{j=c}^{d}x_i\right)$$

And x_i is independent with j, so we can consider as a constant, so

$$\sum_{i=a}^{b}\left(\sum_{j=c}^{d}x_j - x_i \times (d-c+1)\right)$$

Again, by the properties of $\sum$,

$$\sum_{i=a}^{b}\left(\sum_{j=c}^{d}x_j\right) - \sum_{i=a}^{b}\left(x_i \times (d-c+1)\right)$$

$\sum_{j=c}^{d}x_j$ is independent with i, and $(d-c+1)$ is a constant. So finally,

$$\left\{(b-a+1) \times \sum_{j=c}^{d}x_j\right\} - \left\{(d-c+1) \times \sum_{i=a}^{b}x_i\right\}$$

This is the value that we want to calculate. To calculate this formula, we need to know the values of the form

$$\sum_{i=p}^{q}x_i$$

There is an easy way. First calculate the prefix sum $S_i = x_1 + x_2 + \cdots + x_i$ for all i. We can do this by calculating the recursive formula $S_i = S_{i-1} + x_i$ in increasing order of i. Then, using the formula below

$$\sum_{i=p}^{q}x_i = (x_1 + x_2 + \cdots + x_q) - (x_1 + x_2 + \cdots + x_{p-1}) = S_q - S_{p-1}$$

you are able to calculate this sum in $O(1)$ time.

Fastest Walking Strategy

?

Game Inversing

?