高精度计算

Lecture-4

介绍

- 高精度计算，也称为大数计算或任意精度计算，是指能够处理超出标准数据类型表示范围的数值计算方法。
- 实现：模拟竖式运算的过程来实现

  - Ps: 现在的大型比赛中不会出现高精度的题目了（除非蓝桥杯之类的。。）。因为python能够很容易的实现这个过程。

高精度加法

- 引例

题目描述

高精度加法，相当于 a+b problem，**不用考虑负数。**

输入格式

分两行输入。$a, b \leq 10^{500}$。

输出格式

输出只有一行，代表 $a + b$ 的值。

高精度加法

- 数据处理
  - 通过string读入大整数，并将它倒序存入数组中。
  - 倒序存放的原因：
    - 计算中结果可能会在最高位进位导致结果长度边长。
    - 数组能够很好的向后延长（len++），反而向前不太方便。

  - 例如： 2234 ---> [4, 3, 2, 2]
  - 122 ---> [2, 2, 1, 0]
  - plus = [6, 5, 3, 2] ---> 2356

高精度加法

- 算法流程
  - 初始化结果长度 l=max(n, m), 表示至少有 l个数需要相加。
  - 模拟竖式运算：
    - 按位相加: a[i] += b[i]
    - 进位，如果a[i] > 10就往后进位：
      - a[i] -= 10;
      - a[i + 1]++;
    - 判断长度是否变化

高精度加法

```
int l = max(n, m);
for (int i = 1; i <= l; i++) {
    a[i] += b[i];
    a[i + 1] += a[i] / 10;
    a[i] %= 10;
    if (a[l + 1] > 0) l++;
}
```

高精度减法

- 和加法同理
  - 只需要多注意一个正负
  - 例如 145 ---> [5, 4, 1, 0]
  - 2234 ---> [4, 3, 2, 2]
  - minus = [1,1,-1,-2] = -[-1,-1,1,2] (注意最高位不能借位，所以取负)
  - 处理借位:
  - -[-1,-1,1,2] = -[9,-2,1,2] = -[9,8,0,2] = -2089

高精度减法

```
int l = max(n, m);
    for (int i = 1; i <= l; i++) {
        a[i] -= b[i];
    }
    if (a[l] < 0) {
        sig = 1;
        for (int i = 1; i <= l; i++) {
            a[i] = -a[i];
        }
    }
    for (int i = 1; i <= l; i++) {
        if (a[i] < 0) {
            a[i] += 10;
            a[i + 1]--;
        }
    }
    while (l > 1 && a[l] == 0) l--;
```

高精度乘法（高精乘高精）

- 模拟竖式计算即可
- 例如：　　12 = [2, 1]　　56 = [6, 5]
- 　　[2, 1]
- 　　* [6, 5]
- 　----------
- 　　12　10
- 　　6　5
- 　----------
- 　= [12, 16, 5]　　= [2, 17, 5] = [2, 7, 6] = 672

高精度乘法（高精乘高精）

```
for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
        c[i + j - 1] += a[i] * b[j];
    }
}
int l = n + m;
for (int i = 1; i <= l; i++) {
    c[i + 1] += c[i] / 10;
    c[i] %= 10;
    if (c[l + 1] > 0) l++;
}
while (l > 1 && c[l] == 0) l--;
```

高精度除法（高精除单精，整除）

- 前三者的思想都是很类似的，但是除法的处理逻辑就更加的直接了。
- 例如　223 / 12　　223 = [2,2,3]
  - (1)　u = 0,　d = [2, 2, 3]　　ans = []
  - (2)　u = 2,　d = [2, 3]　　ans = [0]
  - (3)　u = 22,　d = [3]　　22 / 12 = 1 余 10　　ans = [0, 1]
  - (4)　u = 103 d = []　　103 / 12 = 8 余 7　　ans = [0, 1, 8] = 18

高精度除法（高精除单精，整除）

```
string ans;
int fl = 0;
for (int i = 1, u = 0; i <= n; i++) {
    u = u * 10 + a[i];
    int div = u / t;
    if (!div && !fl) continue;
    fl = 1;
    ans += div + '0';
    u %= t;
}
```

Python实现高精度

```python
a, b = map(int, input().split(' '))
print(a + b)
print(a - b)
print(a * b)
print(a // b)
```

如果你参加的比赛支持提交python，建议你直接使用python写高精度题目