链滴

PAT 甲级刷题实录——1024

作者：aopstudio
原文链接：https://ld246.com/article/1581221056064
来源网站：链滴

原题链接

https://pintia.cn/problem-sets/994805342720868352/problems/994805476473028608

思路

这题考的又是大数加法操作。一开始我看 N 最多到 10 的 10 次方，还以为可以用 long long，但还有两个测试点过不去，后来发现 K 小于 100，也就是说最多可以加 100 次，那就超出任何整数类型表示的范围了，只能把每一位分别存起来计算了。一开始想用 vector，后来参考了网上的一个方法，stack，因为这样更方便求它的回文数。一个需要注意的点是求每一位加法的时候不要太急，正确的法应该是这样的：

```
int x = result.top() + num.top() + r.top();
result.top() = x % 10;
result.push(x / 10);
```

我一开始仿照 1023 里的加法，写的是

```
result.top() += (num.top() + r.top()) % 10;
result.push((num.top() + r.top()) / 10);
```

结果错了。原因是假设前一位计算产生的进位使得 result.top()当前值 1，则当 num.top()与 r.top()和为 9 的时候，实际上会产生进位，而我的算法却会遗漏这种进位。那就又有一个问题，为什么 1023 里面用类似的方法，不会产生这个问题，1023 中的代码如下：

```
dbNum[index] += digit * 2 % 10; //1023用的是vector实现
dbNum[index + 1] += digit * 2 / 10;
```

相信聪明的读者已经想到了，因为 1023 里面算的是 digit*2，这个结果不可能为 9，也就不会出现 124 里面的这种错误。

完整代码

```
#include <iostream>
#include <stack>
#include <string>
using namespace std;
stack<int> reverse(stack<int> num);
stack<int> addReverse(stack<int> num);
bool equal(stack<int> num, stack<int> rev);
int main()
{
    string N;    //题中要求N小于等于10的10次方，int值已经无法满足
    int K;
    stack<int> num;    //题中要求K小于等于100，也就是最多加它的倒置100次，long long也不够只能逐位处理。使用stack方便倒置
    stack<int> rev;    //倒置后的数
    stack<int> addRev;    //原数和倒置数相加的结果
    cin >> N >> K;
    for (int i = 0; i < N.size(); i++)
    {
        num.push(N[i]-'0');    //逐位读入
```

```
    }
    for (int i = 0; i < K; i++)
    {
        rev = reverse(num);
        if (equal(num, rev))
        {
            while (!num.empty())
            {
                cout << num.top();
                num.pop();
            }
            cout << endl << i;
            return 0;
        }
        else
        {
            num = addReverse(num);
        }
    }
    while (!num.empty())
    {
        cout << num.top();
        num.pop();
    }
    cout << endl << K;
    return 0;
}
stack<int> reverse(stack<int> num)
{
    stack<int> r;
    while(!num.empty())
    {
        r.push(num.top());
        num.pop();
    }
    return r;
}
stack<int> addReverse(stack<int> num)
{
    stack<int> r = reverse(num);
    stack<int> result;
    result.push(0);
    while (!num.empty())
    {
        int x = result.top() + num.top() + r.top();
        result.top() = x % 10;
        result.push(x / 10);
        num.pop();
        r.pop();
    }
    if (result.top() == 0)
        result.pop();
    return result;
}
```

```
bool equal(stack<int> num, stack<int> rev)
{
    bool flag = true;
    while (!num.empty())
    {
        if (num.top() != rev.top())
        {
            flag = false;
            break;
        }
        num.pop();
        rev.pop();
    }
    return flag;
}
```