

## 原创 大数加法+高精度加法+快速乘+龟速乘

2019-07-02 13:44:47 \_-Y--Y- 阅读数 33 更多

编辑

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。

本文链接：[https://blog.csdn.net/weixin\\_44410512/article/details/93487575](https://blog.csdn.net/weixin_44410512/article/details/93487575)

### 前提

因为在32位编译器下

int 4个字节

long 4个字节

long long 8个字节

\_int64 8个字节

double 8个字节

long double 12个字节

unsigned int 4个字节

unsigned long 8个字节

通常情况下一个字节 (bit) 等于八位

例如 int一共有32位 由于第一位是符号位，所以表示数字大小的就只有31位了

int 下最大的数的原码为 0111 1111 1111 1111 1111 1111 1111 1111 即  $2^{31} - 1$  (2147483647)

int 下最小的数的原码为 1000 0000 0000 0000 0000 0000 0000 0000 即  $-2^{31}$  (-2147483648)

同理 long long 最大为  $2^{63} - 1$  (9223372036854775807)

同理 long long 最小为  $-2^{63}$  (-9223372036854775808)

unsigned 表示无符号为 所以 unsigned int 表示数字大小的就有32位了

unsigned int 下最小的数的原码为 0000 0000 0000 0000 0000 0000 0000 0000 即 0

unsigned int 下最大的数的原码为 1111 1111 1111 1111 1111 1111 1111 1111 即  $2^{32} - 1$

同理 unsigned long 最大为  $2^{64} - 1$  (18446744073709551615)

同理 unsigned long 最小为 0

二进制下是怎么相加减的

类似与十进制加减

如图

<https://jingyan.baidu.com/article/86112f135745432736978776.html>

<https://jingyan.baidu.com/article/851fbc379ef4173e1e15ab71.html>

### 正文

如果两个int相乘可能会爆int，那么可以用更高级的long long

那要是两个long long 相乘爆long long了，那就要用  $O(1)$ 快速乘了（模数较小的时候也不准）或 龟速乘了

#### O1快速乘

O1快速乘原理： $a * b - \{(a/mod) * b\} * mod$

模运算实际上是： $a - (a/mod) * mod$

普通乘法取模： $a * b - (a * b/mod) * mod$

运用到乘法上就可以优化 可以将后面的  $(a * b/mod)$  可以用  $\{(longdouble)a/mod * b\} * mod$ 提高精度 但也有不准的可能性，要是没有不卡时间的情况下不建议用这个方法

```
1 | #define ll long long
2 | inline ll mulit(ll x,ll y,ll mod){//O1快速乘
3 |     return (x*y-(ll)((long double)x/mod*y)*mod+mod)%mod;
4 | }
```

#### 龟速乘(左云帆学长会给你们讲的，我就不讲了)

龟速乘其实就是将一个数转换成二进制，然后拆开

例如 23 二进制为 1 0 1 1 1

即  $23=16+4+2+1$

那么  $10^{23}$  就可以分解成

$$1 * 1 * 10 + 1 * 2 * 10 + 1 * 4 * 10 + 0 * 8 * 10 + 1 * 16 * 10$$

1,2,4,8,16都可以累计加起来(例如  $1 + 1 = 2$ ,  $2 + 2 = 4$ ,  $4 + 4 = 8$ )

所以只要看23的二进制是不是1或者是0

若是1就可以加进去, 若是0就不必在加进去

这种方法结果准确, 但时间复杂度较高  $O(\log n)$

## 模版

```

1  ll num_mulit(ll a,ll b,ll c){//龟速乘
2      ll ans=0;
3      ll res=a;
4      while(b){
5          if(b&1)
6              ans=(ans+res)%c;
7              res=(res+res)%c;
8              b>>=1;
9      }
10     return ans;
11 }
```

我要是想知道两个  $10^2$  位甚至  $10^8$  位的整数加起来到底等于多少该怎么办呢,

$10^2$  位的数字肯定不能用一般的int或者long long存了

大于18位的数字一般都是用字符串存

## 大数加法

用模拟的方法计算, 就相当于那手算一样

先将右边对齐, 然后加就可以了

例如:  $111111 + 22222222$

```

  1 1 1 1 1 1
2 2 2 2 2 2 2 2
2 2 2 3 3 3 3 3
```

例如:  $999999999 + 111111111111$

```

  9 9 9 9 9 9 9 9
1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 2 1 1 1 1 1 1 0
```

## 模版

```

1  void num_plus(char *a,char *b){//大数加法
2      char aa[maxn];
3      memset(aa,0,sizeof aa);
4      int lena=strlen(a);
5      int lenb=strlen(b);
6      strrev(a);
7      strrev(b);
8      int len=max(lena,lenb);
9      int yu=0;
10     int i=0;
11     while(1){
12         if(i<lena&&i<lenb){
13             yu=(a[i]-'0')+(b[i]-'0')+yu;
14             aa[i]=yu%10+'0';
15             yu=yu/10;
16         }else if(i<lena){
17             yu=(a[i]-'0')+yu;
18             aa[i]=yu%10+'0';
19             yu=yu/10;
20         }else if(i<lenb){
21             yu=(b[i]-'0')+yu;
22             aa[i]=yu%10+'0';
23             yu=yu/10;
24         }else if(yu!=0){
25             aa[i]=yu%10+'0';
26             yu=yu/10;
```

```
27         }else break;
28         ++i;
29     }
30     strrev(aa);
31     printf("%s\n", aa);
32     return ;
33 }
```

## 例题 1

<https://cn.vjudge.net/problem/HDU-1002>

### AC code

```
1  #include <iostream>
2  #include <cstdio>
3  #include <cstring>
4  #include <algorithm>
5  using namespace std;
6  #define maxn 1000050
7  void num_plus(char *a,char *b){
8      char aa[maxn];
9      memset(aa,0,sizeof aa);
10     int lena=strlen(a);
11     int lenb=strlen(b);
12     strrev(a);
13     strrev(b);
14     int len=max(lena,lenb);
15     int yu=0;
16     int i=0;
17     while(1){
18         if(i<lena&&i<lenb){
19             yu=(a[i]-'0')+(b[i]-'0')+yu;
20             aa[i]=yu%10+'0';
21             yu=yu/10;
22         }else if(i<lena){
23             yu=(a[i]-'0')+yu;
24             aa[i]=yu%10+'0';
25             yu=yu/10;
26         }else if(i<lenb){
27             yu=(b[i]-'0')+yu;
28             aa[i]=yu%10+'0';
29             yu=yu/10;
30         }else if(yu!=0){
31             aa[i]=yu%10+'0';
32             yu=yu/10;
33         }else break;
34         ++i;
35     }
36     strrev(aa);
37     printf("%s\n", aa);
38     return ;
39 }
40 char a[maxn],b[maxn];
41 int main(){
42     int t;
43     scanf("%d", &t);
44     for(int s=1;s<=t;s++){
45         scanf("%s", a);
46         scanf("%s", b);
47         printf("Case %d:\n", s);
48         printf("%s + %s = ", a, b);
49         num_plus(a,b);
50         if(s!=t) printf("\n");
51     }
52     return 0;
53 }
```

## 例题 2

<https://cn.vjudge.net/problem/HDU-1715>

## AC code

```

1  #include <cstdio>
2  #include <cstring>
3  #include <algorithm>
4  #include <iostream>
5  using namespace std;
6  #define maxn 1010
7  char f[maxn][maxn];
8  void solve(){
9      f[1][0]='1';
10     f[2][0]='1';
11     for(int i=3;i<=1000;i++){
12         int lena=strlen(f[i-2]);
13         int lenb=strlen(f[i-1]);
14         int j=-1;
15         int yu=0;
16         while(1){
17             j++;
18             if(j<lena&&j<lenb){
19                 yu+=(f[i-1][j]-'0')+(f[i-2][j]-'0');
20                 f[i][j]=yu%10+'0';
21                 yu=yu/10;
22             }else if(j<lena){
23                 yu+=(f[i-2][j]-'0');
24                 f[i][j]=yu%10+'0';
25                 yu=yu/10;
26             }else if(j<lenb){
27                 yu+=(f[i-1][j]-'0');
28                 f[i][j]=yu%10+'0';
29                 yu=yu/10;
30             }else if(yu!=0){
31                 f[i][j]=yu%10+'0';
32                 yu=yu/10;
33             }else break;
34         }
35     }
36 }
37 int main(){
38     solve();
39     int t,n;
40     scanf("%d", &t);
41     while(t--){
42         scanf("%d", &n);
43         strrev(f[n]);
44         printf("%s\n", f[n]);
45         strrev(f[n]);
46     }
47     return 0;
48 }
49

```

## 高精度加法

我要是计算  $1.1 + 1.9$  怎么办?

首先是要把小数点对齐, 然后从最低位开始加起

1.1

1.9

3.0

要是计算  $1.11 + 2$  怎么办?

因为2没有小数部分, 那就在最后添加小数点

并且为例方便计算, 在小数点后添加0, 是小数部分位数相同

1.11

2.00

3.11

这样, 就可以直接模拟高精度加法了

## 模板

```

1 void num_plus(char *a,char *b,int num){
2     char aa[maxn];
3     memset(aa,0,sizeof aa);
4     int lena=strlen(a);
5     int lenb=strlen(b);
6     strrev(a);
7     strrev(b);
8     int len=max(lena,lenb);
9     int yu=0;
10    int i=0;
11    while(1){
12        if(i<lena&&i<lenb){
13            yu=(a[i]-'0')+(b[i]-'0')+yu;
14            aa[i]=yu%10+'0';
15            yu=yu/10;
16        }else if(i<lena){
17            yu=(a[i]-'0')+yu;
18            aa[i]=yu%10+'0';
19            yu=yu/10;
20        }else if(i<lenb){
21            yu=(b[i]-'0')+yu;
22            aa[i]=yu%10+'0';
23            yu=yu/10;
24        }else if(yu!=0){
25            aa[i]=yu%10+'0';
26            yu=yu/10;
27        }else break;
28        ++i;
29    }
30    int s;
31    for(s=0;s<num;s++){
32        if(aa[s]!='0'){
33            break;
34        }
35    }
36    num-=s;
37    i-=s+1;
38    num=i-num;
39    strrev(aa);
40    for(int j=0;j<=i;j++){
41        printf("%c", aa[j]);
42        if(j==num&&num!=i) printf(".");
43    }
44    printf("\n");
45    return ;
46 }
47
48 void funum_plus(char *a, char *b){
49     char aa[maxn],bb[maxn];
50     memset(aa,0,sizeof aa);
51     memset(bb,0,sizeof bb);
52     int lena=strlen(a);
53     int lenb=strlen(b);
54     int fua=lena,fub=lenb;//行寻找小数点，要是找到记录位置，否则末尾添加小数点
55     for(int i=0;i<lena;i++){
56         if(a[i]=='.'){
57             fua=i;
58             break;
59         }
60     }
61     for(int i=0;i<lenb;i++){
62         if(b[i]=='.'){
63             fub=i;
64             break;
65         }
66     }
67     if(fua==lena) a[lena]='.',lena++;
68     if(fub==lenb) b[lenb]='.',lenb++;
69     int j=0;//将小数点后面的位数填0，使小数点后面位数相同

```

```

70 while(1){
71     j++;
72     if(fua+j<lena&&fub+j<lenb){
73         continue;
74     }else if(fua+j<lena){
75         b[fub+j]='0';
76     }else if(fub+j<lenb){
77         a[fua+j]='0';
78     }else break;
79 }
80 lena+=j;
81 lenb+=j;
82 int la=0,lb=0;
83 for(int i=0;i<lena;i++){
84     if(i!=fua) aa[la++]=a[i];
85 }
86 for(int i=0;i<lenb;i++){
87     if(i!=fub) bb[lb++]=b[i];
88 }
89 num_plus(aa,bb,j-1);
90 }

```

### 例题 3

<https://cn.vjudge.net/contest/287022#problem/G>

### AC code

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  #define maxn 100000
4  void num_plus(char *a,char *b,int num){
5      char aa[maxn];
6      memset(aa,0,sizeof aa);
7      int lena=strlen(a);
8      int lenb=strlen(b);
9      strrev(a);
10     strrev(b);
11     int len=max(lena,lenb);
12     int yu=0;
13     int i=0;
14     while(1){
15         if(i<lena&&i<lenb){
16             yu=(a[i]-'0')+(b[i]-'0')+yu;
17             aa[i]=yu%10+'0';
18             yu=yu/10;
19         }else if(i<lena){
20             yu=(a[i]-'0')+yu;
21             aa[i]=yu%10+'0';
22             yu=yu/10;
23         }else if(i<lenb){
24             yu=(b[i]-'0')+yu;
25             aa[i]=yu%10+'0';
26             yu=yu/10;
27         }else if(yu!=0){
28             aa[i]=yu%10+'0';
29             yu=yu/10;
30         }else break;
31         ++i;
32     }
33     int s;
34     for(s=0;s<num;s++){
35         if(aa[s]!='0'){
36             break;
37         }
38     }
39     num-=s;
40     i-=s+1;
41     num=i-num;
42     strrev(aa);

```

```
43     for(int j=0;j<=i;j++){
44         printf("%c", aa[j]);
45         if(j==num&&num!=i) printf(".");
46     }
47     printf("\n");
48     return ;
49 }
50
51 void funum_plus(char *a, char *b){
52     char aa[maxn],bb[maxn];
53     memset(aa,0,sizeof aa);
54     memset(bb,0,sizeof bb);
55     int lena=strlen(a);
56     int lenb=strlen(b);
57     int fua=lena,fub=lenb;//行寻找小数点,要是找到记录位置,否则末尾添加小数点
58     for(int i=0;i<lena;i++){
59         if(a[i]=='.'){
60             fua=i;
61             break;
62         }
63     }
64     for(int i=0;i<lenb;i++){
65         if(b[i]=='.'){
66             fub=i;
67             break;
68         }
69     }
70     if(fua==lena) a[lena]='.',lena++;
71     if(fub==lenb) b[lenb]='.',lenb++;
72     int j=0;//将小数点后面的位数填0,使小数点后面位数相同
73     while(1){
74         j++;
75         if(fua+j<lena&&fub+j<lenb){
76             continue;
77         }else if(fua+j<lena){
78             b[fub+j]='0';
79         }else if(fub+j<lenb){
80             a[fua+j]='0';
81         }else break;
82     }
83     lena+=j;
84     lenb+=j;
85     int la=0,lb=0;
86     for(int i=0;i<lena;i++){
87         if(i!=fua) aa[la++]=a[i];
88     }
89     for(int i=0;i<lenb;i++){
90         if(i!=fub) bb[lb++]=b[i];
91     }
92     num_plus(aa,bb,j-1);
93 }
94
95 char a[maxn],b[maxn];
96 int main(){
97     while(~scanf("%s %s", a, b)){
98         funum_plus(a,b);
99         memset(a,0,sizeof a);
100        memset(b,0,sizeof b);
101    }
102    return 0;
103 }
104
```

有 0 个人打赏

文章最后发布于: 2019-07-02 13:44:47