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

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

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

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

```
ll num_mulit(ll a,ll b,ll c){//龟速乘
    ll ans=0;
    ll res=a;
    while(b){
      if(b&1)
        ans=(ans+res)%c;
      res=(res+res)%c;
      b>>=1;
    }
    return ans;
}
```

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：$111111+222222222$

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

例如：999999999 + 1111111111111

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

模版

```
void num_plus(char *a,char *b){//大数加法
    char aa[maxn];
    memset(aa,0,sizeof aa);
    int lena=strlen(a);
    int lenb=strlen(b);
    strrev(a);
    strrev(b);
    int len=max(lena,lenb);
    int yu=0;
    int i=0;
    while(1){
        if(i<lena&&i<lenb){
            yu=(a[i]-'0')+(b[i]-'0')+yu;
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else if(i<lena){
            yu=(a[i]-'0')+yu;
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else if(i<lenb){
            yu=(b[i]-'0')+yu;
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else if(yu!=0){
            aa[i]=yu%10+'0';
            yu=yu/10;
```

```
        }else break;
        ++i;
    }
    strrev(aa);
    printf("%s\n", aa);
    return ;
}
```

例题 1

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

AC code

```
#include <iostream>
#include <cstdio>
#include <cstring>
#include <algorithm>
using namespace std;
#define maxn 1000050
void num_plus(char *a,char *b){
    char aa[maxn];
    memset(aa,0,sizeof aa);
    int lena=strlen(a);
    int lenb=strlen(b);
    strrev(a);
    strrev(b);
    int len=max(lena,lenb);
    int yu=0;
    int i=0;
    while(1){
        if(i<lena&&i<lenb){
            yu=(a[i]-'0')+(b[i]-'0')+yu;
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else if(i<lena){
            yu=(a[i]-'0')+yu;
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else if(i<lenb){
            yu=(b[i]-'0')+yu;
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else if(yu!=0){
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else break;
        ++i;
    }
    strrev(aa);
    printf("%s\n", aa);
    return ;
}
char a[maxn],b[maxn];
int main(){
    int t;
    scanf("%d", &t);
    for(int s=1;s<=t;s++){
        scanf("%s", a);
        scanf("%s", b);
        printf("Case %d:\n", s);
        printf("%s + %s = ", a, b);
        num_plus(a,b);
        if(s!=t) printf("\n");
    }
    return 0;
}
```

例题 2

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

AC code

```
#include <cstdio>
#include <cstring>
#include <algorithm>
#include <iostream>
using namespace std;
#define maxn 1010
char f[maxn][maxn];
void solve(){
  f[1][0]='1';
  f[2][0]='1';
  for(int i=3;i<=1000;i++){
    int lena=strlen(f[i-2]);
    int lenb=strlen(f[i-1]);
    int j=-1;
    int yu=0;
    while(1){
      j++;
      if(j<lena&&j<lenb){
        yu+=(f[i-1][j]-'0')+(f[i-2][j]-'0');
        f[i][j]=yu%10+'0';
        yu=yu/10;
      }else if(j<lena){
        yu+=(f[i-2][j]-'0');
        f[i][j]=yu%10+'0';
        yu=yu/10;
      }else if(j<lenb){
        yu+=(f[i-1][j]-'0');
        f[i][j]=yu%10+'0';
        yu=yu/10;
      }else if(yu!=0){
        f[i][j]=yu%10+'0';
        yu=yu/10;
      }else break;
    }
  }
}
int main(){
  solve();
  int t,n;
  scanf("%d", &t);
  while(t--){
    scanf("%d", &n);
    strrev(f[n]);
    printf("%s\n", f[n]);
    strrev(f[n]);
  }
  return 0;
}

```

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 . 1 1
2 . 0 0
3 . 1 1
这样，就可以直接模拟高精度加法了

模板

```
void num_plus(char *a,char *b,int num){
    char aa[maxn];
    memset(aa,0,sizeof aa);
  int lena=strlen(a);
  int lenb=strlen(b);
  strrev(a);
  strrev(b);
    int len=max(lena,lenb);
    int yu=0;
    int i=0;
    while(1){
        if(i<lena&&i<lenb){
            yu=(a[i]-'0')+(b[i]-'0')+yu;
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else if(i<lena){
            yu=(a[i]-'0')+yu;
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else if(i<lenb){
            yu=(b[i]-'0')+yu;
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else if(yu!=0){
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else break;
        ++i;
    }
  int s;
  for(s=0;s<num;s++){
    if(aa[s]!='0'){
      break;
    }
  }
  num-=s;
  i-=s+1;
  num=i-num;
    strrev(aa);
  for(int j=0;j<=i;j++){
    printf("%c", aa[j]);
    if(j==num&&num!=i) printf(".");
  }
    printf("\n");
    return ;
}

void funum_plus(char *a, char *b){
  char aa[maxn],bb[maxn];
  memset(aa,0,sizeof aa);
  memset(bb,0,sizeof bb);
  int lena=strlen(a);
  int lenb=strlen(b);
  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
  for(int i=0;i<lena;i++){
    if(a[i]=='.'){
      fua=i;
      break;
    }
  }
  for(int i=0;i<lenb;i++){
    if(b[i]=='.'){
      fub=i;
      break;
    }
  }
  if(fua==lena) a[lena]='.',lena++;
  if(fub==lenb) b[lenb]='.',lenb++;
  int j=0;//将小数点后面的位数填0，使小数点后面位数相同
```

```
  while(1){
    j++;
    if(fua+j<lena&&fub+j<lenb){
      continue;
    }else if(fua+j<lena){
      b[fub+j]='0';
    }else if(fub+j<lenb){
      a[fua+j]='0';
    }else break;
  }
  lena+=j;
  lenb+=j;
  int la=0,lb=0;
  for(int i=0;i<lena;i++){
    if(i!=fua) aa[la++]=a[i];
  }
  for(int i=0;i<lenb;i++){
    if(i!=fub) bb[lb++]=b[i];
  }
  num_plus(aa,bb,j-1);
}
```

例题 3

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

AC code

```
#include <bits/stdc++.h>
using namespace std;
#define maxn 100000
void num_plus(char *a,char *b,int num){
    char aa[maxn];
    memset(aa,0,sizeof aa);
  int lena=strlen(a);
  int lenb=strlen(b);
  strrev(a);
  strrev(b);
    int len=max(lena,lenb);
    int yu=0;
    int i=0;
    while(1){
        if(i<lena&&i<lenb){
            yu=(a[i]-'0')+(b[i]-'0')+yu;
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else if(i<lena){
            yu=(a[i]-'0')+yu;
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else if(i<lenb){
            yu=(b[i]-'0')+yu;
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else if(yu!=0){
            aa[i]=yu%10+'0';
            yu=yu/10;
        }else break;
        ++i;
    }
  int s;
  for(s=0;s<num;s++){
    if(aa[s]!='0'){
      break;
    }
  }
  num-=s;
  i-=s+1;
  num=i-num;
    strrev(aa);
```

```
  for(int j=0;j<=i;j++){
    printf("%c", aa[j]);
    if(j==num&&num!=i) printf(".");
  }
    printf("\n");
    return ;
}

void funum_plus(char *a, char *b){
  char aa[maxn],bb[maxn];
  memset(aa,0,sizeof aa);
  memset(bb,0,sizeof bb);
  int lena=strlen(a);
  int lenb=strlen(b);
  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
  for(int i=0;i<lena;i++){
    if(a[i]=='.'){
      fua=i;
      break;
    }
  }
  for(int i=0;i<lenb;i++){
    if(b[i]=='.'){
      fub=i;
      break;
    }
  }
  if(fua==lena) a[lena]='.',lena++;
  if(fub==lenb) b[lenb]='.',lenb++;
  int j=0;//将小数点后面的位数填0，使小数点后面位数相同
  while(1){
    j++;
    if(fua+j<lena&&fub+j<lenb){
      continue;
    }else if(fua+j<lena){
      b[fub+j]='0';
    }else if(fub+j<lenb){
      a[fua+j]='0';
    }else break;
  }
  lena+=j;
  lenb+=j;
  int la=0,lb=0;
  for(int i=0;i<lena;i++){
    if(i!=fua) aa[la++]=a[i];
  }
  for(int i=0;i<lenb;i++){
    if(i!=fub) bb[lb++]=b[i];
  }
  num_plus(aa,bb,j-1);
}

char a[maxn],b[maxn];
int main(){
    while(~scanf("%s %s", a, b)){
      funum_plus(a,b);
      memset(a,0,sizeof a);
      memset(b,0,sizeof b);
    }
  return 0;
}
```


文章最后发布于: 2019-07-02 13:44:47