Trong chuyên đề này chúng ta xem xét việc biểu diễn số nguyên lớn ($\approx 10^{100000}$) bằng C++.

1. Biểu diễn số nguyên lớn

Chúng ta chọn COSO=100000000 (10^8) làm cơ số để biểu diễn. Như vậy mỗi chữ số sẽ tương ứng với 8 chữ số trong cơ số 10:

```
#define maxD 10000                    // Số lượng chữ số tối đa trong cơ số 10^8
#define COSO 100000000                // Định nghĩa hằng

struct big{int num;int d[maxD];};     // num - bậc cao nhất, d - mảng chữ số
```

Như vậy các chữ số bậc 0, bậc 1, ... sẽ là d[0], d[1],....

2. Nhập / xuất và so sánh số nguyên lớn

a) Đọc số nguyên lớn từ input

Nhập số nguyên lớn như xâu ký tự, sau đó đảo ngược các chữ số của xâu s để hàng đơn vị đứng đầu,.... Tiếp theo nhóm 8 chữ số của cơ số 10 để tạo ra 1 chữ số của cơ số mới:

```
char s[10*maxD];
void read_big(big &x) {
    scanf("%s",s);
    int n=strlen(s);
    reverse(s,s+n);
    x.num=-1;
    int u=0;
    while (u<n) {
        x.d[++x.num]=0;
        int lt=1;
        for(int i=0;i<8;i++) {
            if (u+i>=n) break;
            x.d[x.num]+=(s[u+i]-48)*lt;
            lt*=10;
        }
        u+=8;
    }
}
```

b) Ghi số nguyên ra output:

Chú ý, ngoại trừ chữ số bậc cao nhất, các chữ số còn lại phải thêm ký tự '0' ở phía trước để cho đủ 8 ký tự (format in "%08d"):

```
void write_big(big x) {
    printf("%d",x.d[x.num]);
    for(int i=x.num-1;i>=0;i--) printf("%08d",x.d[i]);
}
```

c) Gán một số nguyên kiểu int, long long thành số nguyên lớn

```
void Gan(int k,big &x) {
    x.num=-1;
    do {
        x.d[++x.num]=k % COSO;
        k/=COSO;
    } while (k);
}
```

d) So sánh hai số nguyên lớn

Ta viết hàm cmp(x,y) trả về -1,0,1 tùy thuộc vào x<y, x=y, x>y:

```
int cmp(big x,big y) {
    if (x.num>y.num) return 1;
    if (x.num<y.num) return -1;
    for(int i=x.num;i>=0;i--) {
        if (x.d[i]>y.d[i]) return 1;
        if (x.d[i]<y.d[i]) return -1;
    }
    return 0;
}
```

3. Các phép tính số học

a) Phép cộng hai số nguyên lớn:

```
big operator + (big x,big y) {
    big z; z.num=max(x.num, y.num);
    for(int i=x.num+1;i<=z.num;i++) x.d[i]=0;
    for(int i=y.num+1;i<=z.num;i++) y.d[i]=0;
    int tong=0;
    for(int i=0;i<=z.num;i++) {
        tong+=x.d[i]+y.d[i];
        z.d[i]=tong%COSO;
        tong/=COSO;
    }
    if (tong) z.d[++z.num]=1;
    return z;
}
```

b) Phép trừ hai số nguyên lớn

Chú ý rằng ta luôn thực hiện trừ một số lớn cho một số nhỏ hơn hoặc bằng:

```
big operator - (big x,big y) {
    big z; z.num=x.num;
    for(int i=y.num+1;i<=z.num;i++) y.d[i]=0;
    int nho=0;
    for(int i=0;i<=z.num;i++) {
        int hieu=x.d[i]-y.d[i]-nho;
        if (hieu<0) hieu+=COSO, nho=1; else nho=0;
        z.d[i]=hieu;
    }
    while (z.num && z.d[z.num]==0) z.num--;
    return z;
}
```

c) Phép nhân

Ta có hai phép nhân chia ra như sau:

c.1-Nhân một số int với một số nguyên lớn:

```
big operator * (int k,big x) {
    big y; y.num=x.num;
    long long tich=0;
    for(int i=0;i<=y.num;i++) {
        tich+=(long long)k*x.d[i];
        y.d[i]=tich % COSO;
        tich/=COSO;
    }
    while (tich) {y.d[++y.num]=tich % COSO; tich/=COSO;}
    while (y.num && y.d[y.num]==0) y.num--;
    return y;
}
```

c.2-Nhân hai số nguyên lớn

```
big operator * (big x,big y) {
    big z; z.num=x.num+y.num;
    for(int i=x.num+1;i<=z.num;i++) x.d[i]=0;
    for(int i=y.num+1;i<=z.num;i++) y.d[i]=0;
    long long tich=0;
    for(int k=0;k<=z.num;k++) {
        for(int i=0;i<=k;i++) tich+=(long long)x.d[i]*y.d[k-i];
        z.d[k]=tich % COSO;
        tich/=COSO;
    }
    while (tich) {z.d[++z.num]=tich % COSO; tich /=COSO;}
    while (z.num && z.d[z.num]==0) z.num--;
    return z;
}
```

d) Phép chia

d.1-Lấy số dư của một số nguyên lớn cho một số kiểu int:

```
int operator % (big x,int k) {
    int ret=0;
    for(int i=x.num;i>=0;i--) ret=((long long)ret*COSO+x.d[i]) % k;
    return ret;
}
```

d.2-Lấy thương (phần nguyên) của một số nguyên lớn cho một số kiểu int:

```
big operator / (big x,int k) {
    big y; y.num=-1;
    long long du=0;
    for(int i=x.num;i>=0;i--) {
        du=du * COSO + x.d[i];
        y.d[++y.num]=du / k;
        du %=k;
    }
    reverse(y.d,y.d+y.num+1);
    while (y.d && y.d[y.num]==0) y.num--;
    return y;
}
```

d.3-Lấy thương (phần nguyên) của hai số nguyên lớn:

Nhận xét rằng $\frac{a}{b} = x \leftrightarrow bx \leq a$ và ta có thể thực hiện việc tìm kiếm nhị phân kết hợp với các phép tính khác để thực hiện phép chia. Chú ý rằng kỹ thuật này có thể áp dụng để tìm các hàm dơn điệu khác (căn bậc hai, logarit,...):

```
big operator / (big x,big y) {
    big zero, one; Gan(0,zero); Gan(1,one);
    big dau=zero, cuoi=one;
    while (cmp(cuoi*y,x)!=1) {dau=cuoi; cuoi=2*cuoi;}
    while (cmp(cuoi-dau,one)==1) {
        big giua=(dau+cuoi) / 2;
        if (cmp(giua*y,x)!=1) dau=giua; else cuoi=giua;
    }
    return dau;
}
```