进制转换、高精度计算

进制原理：

为了计算数量，产生了数字，数字的进制很多，例如 8 进制，64 进制。以糖葫芦为例讲解，十进制是 10 个葫芦一串，10 串糖葫芦放 1 板，10 板糖葫芦放一盒子；16 进制是 16 个葫芦一串，16 串糖葫芦放 1 板，16 板放 1 盒子。

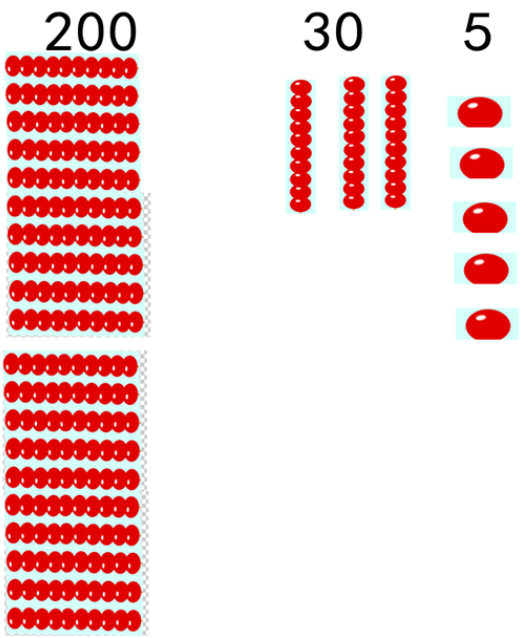
十进制是满 10 进位，16 进制是满 16 进位。

十进制用 0 1 2 3 4 5 6 7 8 9 来表示，
16 进制用 0 1 2 3 4 5 6 7 8 9 A B C D E F 来表示（或者用 0 1 2 3 4 5 6 7 8 9 阿猫 阿狗 鸡 鸭 猪 牛来表示也是可以的，总之数量满了 16 就进位）。

十进制 235，是 10x10x2+10x3+5，
235 转换成 16 进制是 EB，是 16x14+11，E 表示 14 个 16（14 串糖葫芦，1 串 16 个），B 表示 11 个糖葫芦。

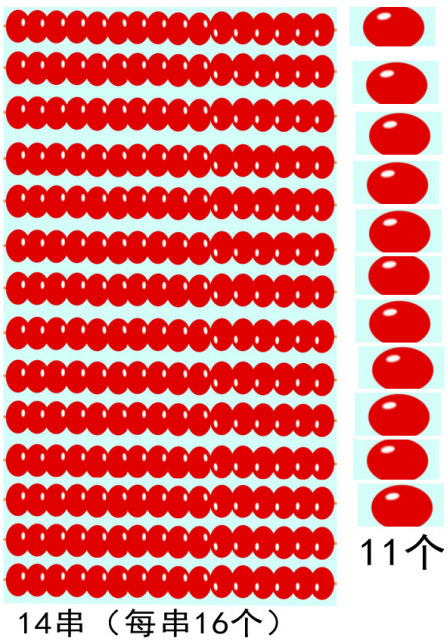
十进制　235

百位 2 是 10x10x2，十位是 10x3，个位是 5



16 进制　EB

E 是 16x14，B 是 11



进制转换

5.2-1

```
#include <iostream>//p 进制转换为十进制
#include <cmath>
using namespace std;
int main()
{
	string pp;//p 进制字符串
	int p,j=0;//p 为本串的进制 j 存第几位
	long long shi=0;//shi 是十进制，使用 long long 是防止 int 存不下
	cin>>pp>>p;
	for(int i=pp.size()-1;i>=0;i--)// 倒序，从个位开始判断
	{
		int num=0;
		if(pp[i]>='0'&&pp[i]<='9') num=pp[i]-'0';// 如字符是数字，转化成数字
		else num=pp[i]-55;// 以 A 为例，A-55=10，恰好对应进制的位数
		shi+=num*pow(p,j);// 数位上的数字乘以该进制的几次方
		j++;
	}
	cout<<shi;
	return 0;
}
```

shi+=num*pow(p,j);

这句代码，以 64 16 为例讲解：shi=4+6*16

以 1E 16 为例讲解：shi=14+16*1

以 ABF 16 为例讲解：shi=15+11*16+10*16*16

F=15,

B=11*16,

A=10*16*16

```
64 16
100

ABF 16
2751

1E 16
30

10000000000     2
1024
```

5.2-2

```
#include <iostream>// 十进制转 16 进制之内
using namespace std;
string pp;//p 进制的字符
string pz="0123456789ABCDEF";
int main()
{
	int i,r,shi,p;//shi, 十进制数值; p,p 进制
	cin>>shi>>p;
	while(shi!=0)
	{
		r=shi%p;// 记录余数
		pp+=pz[r];// 存入数组中
		shi/=p;
	}
	for(i=pp.size()-1;i>=0;i--)
		cout <<pp[i];// 倒序输出
	return 0;
}
```

```
100 16
64


30 16
1E

1024 2
10000000000
```

5.2-3// 本题选自《一本通》

```
#include<iostream>// 十进制转 16 进制之内
using namespace std;
char d[16]={'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'};
int a,b;
int zhuanhuan(int x,int y)// 十进制数值 x 转 y 进制
{
	int r;
	r=x%y;
	x=x/y;
	if(x!=0) zhuanhuan(x,y);
	cout<<d[r];
}
int main()
{
	cin>>a>>b;
	zhuanhuan(a,b);
	return 0;
}
```

```
100 16
64


30 16
1e

1024 2
10000000000
```

5.2-4 注意：此段代码仅供参考，此函数不是标准函数，在低版本 dev 不支持

```
#include<cstdlib>// 十进制转其他进制
#include <iostream>
using namespace std;
int main()
{
	int n,r;
	char str[30];
	cin>>n>>r;//r 为需要转换的进制
	itoa(n,str,r);
	cout<<str;
}
```

高精度计算（一本通训练指导）

【题目描述】求两个不超过 200 位的非负整数的和。

【输入】有两行，每行是一个不超过 200 位的非负整数，可能有多余的前导 0。

【输出】相加后的结果，结果里不能有多余的前导 0，如果结果是 342，不能输出为 0342。

【输入样例】　　【输出样例】

2222222222222222222　　3333333333333333333　　55555555555555555555

9.0-1a

```
#include<iostream>//1. 大整数加法
#include<cstring>
using namespace std;
int a[200],b[200],c[200];
char aa[2000],bb[2000];
int main()
{
	int i,la,lb,lc,x=0;
	cin>>aa>>bb;
	la=strlen(aa);lb=strlen(bb);
	for(i=0;i<la;i++) a[la-i]=aa[i]-48;// 字符转成数值，倒序，储存在 a,b 数组
	for(i=0;i<lb;i++) b[lb-i]=bb[i]-48;
	for(i=0;i<=la;i++) cout<<a[i]<<" "<<b[i]<<"     ";// 观察转存后的字符串
	lc=1;//lc 是位数，从个位数开始操作，然后是十位数，百位数
	while(lc<=la||lc<=lb)
	{
		c[lc]=a[lc]+b[lc]+x;
		x=c[lc]/10;//x 是进的数字
		c[lc]%=10;
		lc++;
	}
	c[lc]=x;// 最后一个数字
	while(c[lc]==0) lc--;// 如果最后数字是 0，去掉
	cout<<endl<<" 答案：";
	for(i=lc;i>=1;i--) cout<<c[i];
	return 0;
}
```

9.0-1b

```
#include<iostream>// 课课通 1. 高精度加法
#include<cstring>
using namespace std;
char sa[1010],sb[1010];
int i,la,lb,lc,a[1010],b[1010],c[1010];
int main()
{
	cin>>sa>>sb;
	la=strlen(sa);lb=strlen(sb);
	memset(a,0,sizeof(a));
	memset(b,0,sizeof(b));// 初始化数组
	for(i=0;i<la;i++)
	{
		a[la-i-1]=sa[i]-'0';// 读入的字符转成数字存储
	}
	for(i=0;i<lb;i++)
	{
		b[lb-i-1]=sb[i]-'0';
	}
	lc=la>lb?la:lb;
	memset(c,0,sizeof(c));
	for(i=0;i<lc;i++)
	{
		c[i]=a[i]+b[i]+c[i];// 对应位相加，包括上一次运算产生的进位
		if(c[i]>=10)// 处理进位
		{
			c[i+1]=1;
			c[i]-=10;
		}
	}
	if(c[lc]>0) lc++;
	for(i=lc-1;i>=0;i--) cout<<c[i];
	return 0;
}
```

```
#include<iostream>//2. 高精度减法
#include<cstring>
using namespace std;
int main()
{
	int a[256],b[256],c[256],lena,lenb,lenc,i;
	char a1[256],a2[256],n[256];
	cin>>a1>>a2;
	memset(a,0,sizeof(a));memset(b,0,sizeof(b));memset(c,0,sizeof(c));
	lena=strlen(a1);lenb=strlen(a2);
	if(lena<lenb||lena==lenb&&strcmp(a1,a2)<0)
	{// 第一个数字比第二个小，互换，并且输出一个负号
		swap(a1,a2);
		cout<<"-";
	}
	for(i=0;i<lena;i++) a[lena-i]=a1[i]-48;// 字符变成数字，倒序储存在 a,b 数组
	for(i=0;i<lenb;i++) b[lenb-i]=a2[i]-48;// 下标 1-lenb(lena)
	i=1;
	while(i<=lena||i<=lenb)// 结果肯定小于其中一个数，所以位数也要小于等于
	{
		if(a[i]<b[i])
		{
			a[i]+=10;
			a[i+1]--;
		}// 如果对应的一位被减数比减数小，前面一位减 1，本位加 10，借 1 当 10
		c[i]=a[i]-b[i];
		i++;
	}
	lenc=i;
	while((c[lenc]==0)&&(lenc>1))
	lenc--;// 如果有 0，就删掉
	for(i=lenc;i>=1;i--)
	cout<<c[i];
	return 0;
}
```

【题目描述】求两个大的正整数相减的差。

【输入】共 2 行，第 1 行是被减数 a，第 2 行是减数 b(a ＞ b)。每个大整数不超过 200 位，不会有多余的前导零。

【输出】一行，即所求的差。

【输入样例】

```
99999999999999999999999999999999999999
9999999999999
```

【输出样例】

```
99999999999999999999999990000000000000
```

【题目描述】任意给定一个正整数 N(N<=100)，计算 2 的 n 次方的值。

【输入】输入一个正整数 N。

【输出】输出 2 的 N 次方的值。

【输入样例】5

【输出样例】32

9.0-3a

```
#include<iostream>//3.b 计算 2 的 n 次方
using namespace std;
int a[10000];
int main()
{
	int n,i,j,x,len=1;//1 表示得数位数，初始化为 1
	cin>>n;
	a[1]=1;// 得数初始化为 1
	for(i=1;i<=n;i++)
	{
		x=0;//x 进位，初始化为 0
		for(j=1;j<=len;j++)
		{
			a[j]=a[j]*2+x;// 得数的每一位都 *2 再十进位
			x=a[j]/10;// 计算进位
			a[j]%=10;// 保留个位数
			if(x!=0&&j==len) len++;
		}
	}
	for(i=len;i>=1;i--) cout<<a[i];
	return 0;
}
```

9.0-3b

```
#include<iostream>//3.a 计算 2 的 n 次方
#include<cmath>
using namespace std;
int n,s,len=1,a[10001];
int i,k;
int main()
{
	cin>>n;
	a[1]=1;// 因为是乘法，答案预设为 1
	for(k=1;k<=n;k++)// 进行 n 次连乘
	{
		for(i=1;i<=len;i++) a[i]*=2;// 每个位数都乘以 2
		for(i=1;i<=len;i++)
		{
			s=0;
			if(a[i]>9)// 判断是否需要进位
			{
				a[i+1]+=a[i]/10;// 进位
				a[i]%=10;// 只留下个位
				s=max(s,i+1);//max 函数，s 等于 s 和 i+1 中大的数
			}
		}
		len=max(len,s);// 判断位数，避免多乘
	}
	for(i=len;i>=1;i--) cout<<a[i];
	return 0;
}
```

【题目描述】

已知正整数 k 满足 2 ≤ k ≤ 9，现给出长度最大为 30 位的十进制非负整数 c，求所有能整除 c 的 k。

【输入】一个非负整数 c，c 的位数≤ 30。

【输出】

若存在满足 c%k == 0 的 k，从小到大输出所有这样的 k，相邻两个数之间用单个空格隔开；若没有这样的 k，则输出 "none"。

【输入样例】

30

【输出样例】

2 3 5 6

9.0-4

```
#include<iostream>//4.b 大整数因子
#include<cstring>
using namespace std;
const int N=31;
int i,k,l,h;
bool flag;
int c[N],b[N];
char s[N];

bool div(int k)
{
	for(i=0;i<l;i++) b[i]=c[i];
	for(i=i-1;i>=1;i--)
	{
		b[i-1]+=(b[i]%k)*10;// 将上一次计算得到的余数与被除数的下一位构成新的被除数
		b[i]/=k;
	}
	if(b[0]%k==0) return 1;// 判断是否满足
	return 0;
}
```

```
int main()
{
	cin>>s;
	h=l=strlen(s);// 先用字符数组读入
	for(i=0;i<l;i++)// 将字符数组转成数字存于数组
	{
		h--;
		c[i]=s[h]-'0';
	}
	for(k=2;k<=9;k++)
	{
		if(div(k))
		{
			cout<<k<<" ";
			flag=1;
		}
	}
	if(!flag) cout<<"none";
	return 0;
}
```

【题目描述】

求 10000 以内 n 的阶乘。

【输入】

只有一行输入，整数 n（0 ≤ n ≤ 10000）。

【输出】

一行，即 n! 的值。

【输入样例】

4

【输出样例】

24

9.0-5

```
#include<iostream>//5. 阶乘
#include<cstring>
using namespace std;
int i,j,a[40010],n,x,len=1;
int main()
{
	cin>>n;
	a[1]=1;// 初始化为 1
	for(i=1;i<=n;i++)
	{
		x=0;
		for(j=1;j<=len;j++)//len 当前整数的长度
		{
			a[j]=a[j]*i+x;
			x=a[j]/10;//x 处理上一位的进位
			a[j]=a[j]%10;
			if(x>0&&j>=len) len++;// 判断进位后整数长度是否增加
		}
	}
	for(i=len;i>=1;i--) cout<<a[i];
	return 0;
}
```

【题目描述】

用高精度计算出S=1!+2!+3!+…+n!（n≤100），其中“!”表示阶乘，例如：5!=5×4×3×2×1。

输入正整数n，输出计算结果S。

【输入】一个正整数n。【输出】计算结果S。

【输入样例】5【输出样例】153

9.0-6

```
#include<iostream>//6. 阶乘和
#include<cstring>
using namespace std;
int a[500],sum[500];
void mul(int x)// 高精乘
{
    int i;
    for(i=1;i<=a[0];i++)
        a[i]*=x;
    for(i=1;i<=a[0];i++)
    {
        a[i+1]+=a[i]/10;
        a[i]%=10;
    }
    i=a[0];
    while(a[i+1]>0)
        i++;
    a[0]=i;
    i=a[0];
    while(a[i]>10)
    {
        a[i+1]+=a[i]/10;
        a[i]%=10;
        i++;
    }
    a[0]=i;
}
```

```
void add()// 高精加
{
    int i;
    if(sum[0]>a[0])
        sum[0]=sum[0];
    else
        sum[0]=a[0];
    for(i=1;i<=sum[0];i++)
    {
        sum[i]+=a[i];
        sum[i+1]+=sum[i]/10;
        sum[i]%=10;
    }
    if(sum[sum[0]+1]>0)
        sum[0]+=1;
}
int main()
{
    int n;
    int i;

    cin>>n;
    a[0]=1;a[1]=1;sum[0]=1;sum[1]=0;
    for(i=1;i<=n;i++)
    {
        mul(i);// 计算阶乘
        add();// 计算阶乘和
    }
    for(i=sum[0];i>=1;i--)
        cout<<sum[i];
    cout<<endl;
    return 0;
}
```

【题目描述】

求两个不超过 200 位的非负整数的积。

【输入】

有两行，每行是一个不超过 200 位的非负整数，没有多余的前导 0。

【输出】

一行，即相乘后的结果。结果里不能有多余的前导 0，即如果结果是 342，那么就不能输出为 0342。

【输入样例】

12345678900

98765432100

【输出样例】

1219326311126352690000

9.0-7

```
#include<iostream> // 高精度乘法
#include<cstring>
#include<string>
using namespace std;
int main()
{
    char str1[256],str2[256];
    int a[256],b[256],c[256];
    int lena,lenb,lenc;
    int x;
    int i,j;

    memset(a,0,sizeof(a));
    memset(b,0,sizeof(b));
    memset(c,0,sizeof(c));

    cin>>str1;// 输入乘数 str1
    cin>>str2;// 输入乘数 str2
```

```
    lena=strlen(str1);
    lenb=strlen(str2);
    for(i=0;i<=lena-1;i++)// 乘数 str1 存入数组 a
        a[lena-i]=str1[i]-'0';
    for(i=0;i<=lenb-1;i++)// 乘数 str2 存入数组 b
        b[lenb-i]=str2[i]-'0';

    for(i=1;i<=lenb;i++)
    {
        x=0;// 用于存放进位
        for(j=1;j<=lena;j++)// 对乘数每一位进行处理
        {
            c[i+j-1]=a[j]*b[i]+x+c[i+j-1];// 当前乘积 + 上次乘积进位 + 原数
            x=c[i+j-1]/10;
            c[i+j-1]%=10;
        }
        c[i+lena]=x;// 进位
    }
    lenc=lena+lenb;
    while((c[lenc]==0)&&(lenc>1))// 删除前导 0
        lenc--;
    for(i=lenc;i>=1;i--)// 倒序输出
        cout<<c[i];
    cout<<endl;
    return 0;
}
```

【题目描述】

输入一个大于 0 的大整数 N，长度不超过 100 位，要求输出其除以 13 得到的商和余数。

【输入】一个大于 0 的大整数，长度不超过 100 位。

【输出】两行，分别为整数除法得到的商和余数。

【输入样例】2132104848488485

【输出样例】164008065268345　　0

9.0-8

```
#include<iostream>//8. 除以 13
#include<cstring>
using namespace std;
int a[101],c[101];
int main()
{
	char al[101],cl[101];
	int len,i,x=0,lena,lenc,b;
	cin>>al;
	b=13;
	lena=strlen(al);
	for(i=0;i<=lena-1;i++)
	{
		a[i+1]=al[i]-48;// 转换为整型数组
	}
	for(i=1;i<=lena;i++)
	{
		c[i]=(x*10+a[i])/b;// 将之前的余数加上新的一位再除以 13
		x=(x*10+a[i])%b;// 求出余数
	}
	lenc=1;
	while(c[lenc]==0&&lenc<lena)// 删除多余的 0
		lenc++;
	for(i=lenc;i<=lena;i++) cout<<c[i];
	cout<<endl<<x;
	return 0;
}
```