《算法竞赛备战资料》

TianMaXingKong

**文件输入**

#include<iostream>

using namespace std;

int main()

{

freopen("test.txt","r",stdin);

fclose(stdin);

}

**高精度问题**

注意点：

1. #include<cstring>而不是#include<string>
2. 少用string因为运行速度较慢
3. 以字符型输入后要ASCII码转换 减48
4. Class内使用的#define maxn 不能太大
5. 倒序赋值，倒序输出，正序操作
6. I要减减
7. Vector<int> number;为空时要先push\_back(```);

代码：

#include<iostream>

#include<cstring>

#include<cmath>

#include<vector>

#define maxn 10000000 //不能太大

using namespace std;

char setnum[maxn];

class BigNum

{

public:

BigNum(){

memset(setnum,0,sizeof(setnum));

}

void Setnum();

void display();

friend BigNum operator + (BigNum &,BigNum &);

friend BigNum operator - (BigNum &,BigNum &);

friend BigNum operator \* (BigNum &,BigNum &);

friend BigNum operator / (BigNum &,BigNum &);

//private:

vector<int> num;

int longth;

};

int main()

{

BigNum num1,num2,num3;

num1.Setnum(),num2.Setnum();

/\*

if(num1.longth>num2.longth)

num3=num1-num2;

else

{

num3=num2-num1;//还要取反

//cout<<"-";

}\*/

num3=num1\*num2;

num3.display();

return 0;

}

void BigNum::display() //容易错

{

int i=num.size()-1;

for(;i>=0;i--)

if(num[i]==0) continue;

else break;

for(;i>=0;i--)

cout<<num[i];

}

void BigNum::Setnum()

{

cin>>setnum;longth=strlen(setnum);

for(int i=longth-1;i>=0;i--)

num.push\_back(setnum[i]-48);

//num[i]=setnum[i]-48; //只能压入

}

BigNum operator +(BigNum &num1,BigNum &num2) //高精度加法

{

BigNum num3;

int longth1=num1.num.size(),longth2=num2.num.size();

int maxl=max(longth1,longth2);

for(int i=0;i<maxl**+1**;i++) num3.num.push\_back(0); //初始化很重要

for(int i=0;i<maxl;i++)

{

if(i<longth1) num3.num[i]+=num1.num[i];

if(i<longth2) num3.num[i]+=num2.num[i];

if(num3.num[i]>=10)

{

num3.num[i+1]+=num3.num[i]/10;

num3.num[i]%=10;

}

}

return num3;

}

BigNum operator -(BigNum &num1,BigNum &num2) //高精度减法

{

BigNum num3;

int longth1=num1.num.size(),longth2=num2.num.size();

int maxl=max(longth1,longth2);

for(int i=0;i<maxl+1;i++) num3.num.push\_back(0);

for(int i=0;i<maxl;i++)

{

if(i<longth1) num3.num[i]+=num1.num[i];

if(i<longth2) num3.num[i]-=num2.num[i];

if(num3.num[i]<0)

{

num3.num[i+1]--;

num3.num[i]+=10;

}

}

return num3;

}

BigNum operator \*(BigNum &num1,BigNum &num2) //高精度乘法

{

BigNum num3;

int longth1=num1.num.size(),longth2=num2.num.size();

int maxl=longth1+longth2;

for(int i=0;i<maxl;i++) num3.num.push\_back(0);

for(int i=0;i<longth1;i++)

for(int j=0;j<longth2;j++)

num3.num[i+j]+=num1.num[i]\*num2.num[j];

for(int i=0;i<maxl;i++)

if(num3.num[i]>=10)

{

num3.num[i+1]+=num3.num[i]/10;

num3.num[i]%=10;

}

return num3;

}

//高精度除法

//高精度进制转换

**二叉树**

#include<iostream>

#include<cstdio>

#include<cstring>

#include<cstdlib>

#define maxn 1000

using namespace std;

class tree

{

public:

int data;

tree \*left,\*right;

};

**tree \***creat(int a[],int i,int n)

{

if(i>=n) return NULL;

else

{

**tree \*p=(tree\*)malloc(sizeof(tree));**

p->data=a[i];

p->left=creat(a,2\*i,n);

p->right=creat(a,2\*i+1,n);

return p;

}

}

void Visit(tree \*p)

{

cout<<p->data;

}

void preorder(**tree \*p**) //前序遍历

{

if(p!=NULL)

{

Visit(p);

preorder(p->left);

preorder(p->right);

}

}

void inorder(**tree \*p**) //中序遍历

{

if(p!=NULL)

{

inorder(p->left);

Visit(p);

inorder(p->right);

}

}

void postorder(**tree \*p**) //后序遍历

{

if(p!=NULL)

{

postorder(p->left);

postorder(p->right);

Visit(p);

}

}

int main()

{

int a[maxn];

for(int i=0;i<maxn;i++) a[i]=i;

**tree \*p**=creat(a,1,8);

postorder(p);

return 0;

}

**排列组合**

1. **子集----二进制法**

#include<iostream>

#include<algorithm>

using namespace std;

int main()

{

int a[]={1,2,3,4,5};

int choose[]={0,0,1,1,1};

do

{

for(int i=0;i<5;i++)

if(choose[i])

cout<<a[i]<<" ";cout<<endl<<endl;

}while(next\_permutation(choose,choose+5));

return 0;

}

1. **全排列**

Int n[maxn];

next\_permutation(n,n+maxn)；

**素数筛**

#include<iostream>

#include<cmath>

#define maxn 100001

using namespace std;

int prime[maxn];

void all\_prime()

{

for(int i=0;i<=3;i++) prime[i]=1;

for(int i=4;i<maxn;i++) prime[i]=i%2==0?0:1;

int t=sqrt(maxn);

for(int i=2;i<=t;i++)

if(prime[i])

for(int j=i\*i;j<maxn;j+=2\*i)

prime[j]=0;

}

int main()

{

all\_prime();

for(int i=2;i<=100;i++)

if(prime[i])

cout<<i<<endl;

return 0;

}

**二分查找：**

**lower\_bound**

**upper\_bound**

**binary\_search**

**欧拉回路**

**最大约数**