

超级高精度 加减乘

```
1  #include<iostream>
2  #include<string>
3  #include<cstring>
4  #include<cstdio>
5  using namespace std;
6  const int BIT = 2005;
7  const int N = BIT*BIT;
8  struct bign
9  {
10     int len,s[N];
11     bign() { memset(s,0,sizeof(s)); len=1; }
12     bign(int num) { *this=num; }
13     bign(char *num) { *this=num; }
14     bign operator =(int num)
15     {
16         char c[N];
17         sprintf(c,"%d",num);
18         *this=c;
19         return *this;
20     }
21     bign operator =(const char *num)
22     {
23         len=strlen(num);
24         for (int i=0;i<len;i++) s[i]=num[len-1-i]-'0';
25         return *this;
26     }
27     string str()
28     {
29         string res="";
30         for (int i=0;i<len;i++) res=(char)(s[i]+'0')+res;
31         return res;
32     }
33     void clean()
34     {
35         while (len>1&&!s[len-1]) len--;
36     }
37     bign operator +(const bign &b)
38     {
39         bign c;
40         c.len=0;
41         for (int i=0,g=0;i<len||i<b.len;i++)
42         {
43             int x=g;
44             if (i<len) x+=s[i];
45             if (i<b.len) x+=b.s[i];
```

```

46         c.s[c.len++]=x%10;
47         g=x/10;
48     }
49     return c;
50 }
51 bign operator -(const bign &b)
52 {
53     bign c;
54     c.len=0;
55     int x;
56     for (int i=0,g=0;i<len;i++)
57     {
58         x=s[i]-g;
59         if (i<b.len) x-=b.s[i];
60         if (x>=0) g=0;
61         else{
62             x+=10;
63             g=1;
64         };
65         c.s[c.len++]=x;
66     }
67     c.clean();
68     return c;
69 }
70 bign operator *(const bign &b)
71 {
72     bign c;
73     c.len=len+b.len;
74     for (int i=0;i<len;i++) for (int j=0;j<b.len;j++) c.s[i+j]+=s[i]*b.s[j];
75     for (int i=0;i<c.len-1;i++) { c.s[i+1]+=c.s[i]/10; c.s[i]%10; }
76     c.clean();
77     return c;
78 }
79 bool operator <(const bign &b)
80 {
81     if (len!=b.len) return len<b.len;
82     for (int i=len-1;i>=0;i--)
83         if (s[i]!=b.s[i]) return s[i]<b.s[i];
84     return false;
85 }
86 bool operator ==(const bign &b)
87 {
88     if (len!=b.len) return false;
89     for (int i=len-1;i>=0;i--)
90         if (s[i]!=b.s[i]) return false;
91     return true;
92 }
93 bool operator !=(const bign &b)
94 {

```

```

95         if (len!=b.len) return true;
96         for (int i=len-1;i>=0;i--)
97             if (s[i]!=b.s[i]) return true;
98         return false;
99     }
100     bign operator +=(const bign &b)
101     {
102         *this=*this+b;
103         return *this;
104     }
105     bign operator --(const bign &b)
106     {
107         *this=*this-b;
108         return *this;
109     }
110 };
111 istream& operator >>(istream &in,bign &x)
112 {
113     string s;
114     in>>s;
115     x=s.c_str();
116     return in;
117 }
118 ostream& operator <<(ostream &out,bign &x)
119 {
120     out<<x.str();
121     return out;
122 }
123 int main(){
124     bign a,b,c;
125     //ios::sync_with_stdio(false);
126     cin>>a>>b;
127     c=a*b;
128     cout<<c<<endl;
129     return 0;
130 }

```

高精度加减乘 高-低除

```

1  #include<iostream>
2  #include<cstring>
3  using namespace std;
4
5  string prec_plus(string plus_s1,string plus_s2){
6      int plus_i1[10100],plus_i2[10100];
7      int l1=plus_s1.length(),l2=plus_s2.length();
8      string ans="";
9      int len=max(l1,l2);

```

```

10     memset(plus_i1,0,sizeof(plus_i1));
11     memset(plus_i2,0,sizeof(plus_i2));
12     for(int i=l1-1;i>=0;i--)
13         plus_i1[l1-i-1]=plus_s1[i]-'0';
14     for(int i=l2-1;i>=0;i--)
15         plus_i2[l2-i-1]=plus_s2[i]-'0';
16     for(int i=0;i<len;i++){
17         plus_i1[i]+=plus_i2[i];
18         plus_i1[i+1]+=plus_i1[i]/10;
19         plus_i1[i]%=10;
20     }
21     if(plus_i1[len]!=0) len++;
22     while(plus_i1[len-1]==0 and len>1)
23         len--;
24     for(int i=len-1;i>=0;i--)
25         ans=ans+char(plus_i1[i]+'0');
26     return ans;
27 }
28
29 string prec_minus(string minus_s1,string minus_s2){
30     int minus_i1[10100],minus_i2[10100];
31     int l1=minus_s1.length(),l2=minus_s2.length();
32     string ans="";
33     int len=max(l1,l2);
34     memset(minus_i1,0,sizeof(minus_i1));
35     memset(minus_i2,0,sizeof(minus_i2));
36     for(int i=l1-1;i>=0;i--)
37         minus_i1[l1-i-1]=minus_s1[i]-'0';
38     for(int i=l2-1;i>=0;i--)
39         minus_i2[l2-i-1]=minus_s2[i]-'0';
40     for(int i=0;i<len;i++){
41         minus_i1[i]-=minus_i2[i];
42         if(minus_i1[i]<0){
43             minus_i1[i]+=10;
44             minus_i1[i+1]--;
45         }
46     }
47     while(minus_i1[len-1]==0 and len>1)
48         len--;
49     for(int i=len-1;i>=0;i--)
50         ans=ans+char(minus_i1[i]+'0');
51     return ans;
52 }
53
54 string prec_multiply(string multiply_s1,string multiply_s2){
55     int multiply_i1[1010],multiply_i2[1010],multiply_i3[1010];
56     int l1=multiply_s1.length(),l2=multiply_s2.length();
57     string ans="";
58     int len=(l1+l2);

```

```

59     memset(multiply_i1,0,sizeof(multiply_i1));
60     memset(multiply_i2,0,sizeof(multiply_i2));
61     memset(multiply_i3,0,sizeof(multiply_i3));
62     for(int i=l1-1;i>=0;i--)
63         multiply_i1[l1-i-1]=multiply_s1[i]-'0';
64     for(int i=l2-1;i>=0;i--)
65         multiply_i2[l2-i-1]=multiply_s2[i]-'0';
66     for(int i=0;i<l1;i++){
67         for(int j=0;j<l2;j++){
68             multiply_i3[i+j]+=multiply_i1[i]*multiply_i2[j];
69             multiply_i3[i+j+1]+=multiply_i3[i+j]/10;
70             multiply_i3[i+j]%=10;
71         }
72     }
73     while(multiply_i3[len-1]==0 and len>1)
74         len--;
75     for(int i=len-1;i>=0;i--)
76         ans=ans+char(multiply_i3[i]+'0');
77     return ans;
78 }
79
80 string prec_division(string div_s1,int div_i2){
81     int div_i1[10100];
82     memset(div_i1,0,sizeof(div_i1));
83     int l1=div_s1.length();
84     for(int i=0;i<l1;i++)
85         div_i1[i]=div_s1[i]-'0';
86     int div_t=0;
87     for(int i=0;i<l1;i++){
88         div_t=div_t*10+div_i1[i];
89         div_i1[i]=div_t/div_i2;
90         div_t%=div_i2;
91     }
92     bool div_f=false;
93     string ans;
94     for(int i=0;i<l1;i++){
95         if(div_i1[i]) div_f=true;
96         if(div_f or i==l1-1) ans=ans+char(div_i1[i]+'0');
97     }
98     return ans;
99 }
100

```

归并排序

```

1  #pragma GCC optimize(2)
2  #include<bits/stdc++.h>
3  #define abss(x) ((x)>(0)?(x):(-1)*(x))

```

```

4  #define maxs(a,b) ((a)>(b)?(a):(b))
5  #define mins(a,b) ((a)<(b)?(a):(b))
6  #define FOR(i,a,b) for(register int i=(a);i<=(b);i++)
7  #define ROF(i,a,b) for(register int i=(a);i>=(b);i--)
8  #define mem(a) memset(a,0,sizeof(a))
9  const int INF (1<<30);
10 const int inf (-1<<30);
11 using namespace std;
12
13 int tmp[int(1e5)]={};
14 void merge_sort(int q[],int l,int r){
15
16     if(l>=r) return;
17     int mid=l+r>>1;
18     merge_sort(q,l,mid);
19     merge_sort(q,mid+1,r);
20     int k=0,i=l,j=mid+1;
21     while(i<=mid and j<=r){
22         if(q[i]<=q[j]) tmp[k++]=q[i++];
23         else tmp[k++]=q[j++];
24     }
25     while(i<=mid) tmp[k++]=q[i++];
26     while(j<=r) tmp[k++]=q[j++];
27     for(i=l,j=0;i<=r;i++,j++) q[i]=tmp[j];
28 }
29
30 int main(){
31     int n,a[int(1e5)];
32     cin>>n;
33     FOR(i,0,n-1) scanf("%d",a+i);
34     merge_sort(a,0,n-1);
35     FOR(i,0,n-1) printf("%d ",a[i]);
36     return 0;
37 }
38

```

快速排序

```

1  #pragma GCC optimize(2)
2  #include<bits/stdc++.h>
3  #define abss(x) ((x)>(0)?(x):(-1)*(x))
4  #define maxs(a,b) ((a)>(b)?(a):(b))
5  #define mins(a,b) ((a)<(b)?(a):(b))
6  #define FOR(i,a,b) for(register int i=(a);i<=(b);i++)
7  #define ROF(i,a,b) for(register int i=(a);i>=(b);i--)
8  #define mem(a) memset(a,0,sizeof(a))
9  const int INF (1<<30);
10 const int inf (-1<<30);

```

```
11 using namespace std;
12
13 void qsort(int a[],int l,int r){
14     int mid=a[(l+r)/2];
15     int i=l,j=r;
16     while(i<j){
17         while(a[i]<mid) i++;
18         while(a[j]>mid) j--;
19         if(i<=j){
20             swap(a[i],a[j]);
21             i++;j--;
22         }
23     }
24     if(l<j) qsort(a,l,j);
25     if(r>i) qsort(a,i,r);
26 }
27
28 int main(){
29     int n,a[int(1e5)];
30     cin>>n;
31     FOR(i,0,n-1) scanf("%d",a+i);
32     qsort(a,0,n-1);
33     FOR(i,0,n-1) printf("%d ",a[i]);
34     return 0;
35 }
36
37
```