牛客多校第三场

本场感觉还算比较满意吧，过了不少题。

A

很容易。适合作为第一学期程序设计的压轴题，或者算法课贪心算法中的水题。

```
#include<stdio.h>

int t,n;
char s[2000005];

int main()
{
    scanf("%d",&t);
    while(t--)
    {
        scanf("%d",&n);
        scanf("%s",s);
        int cnt1=0,ans=0;
        int i;
        for(i=0;i<n;i++)
        {
            switch(s[i])
            {
            case'0':
                if(cnt1>0)
                {
                    ans++;
                    cnt1--;
                }
                break;
            case'1':
                cnt1++;
                break;
            default:
                ans++;
            }
        }
        printf("%d\n",ans+cnt1/2);
    }
    return 0;
}
```

B

这个更容易□C语言程序设计课的难度□□ps□不会平衡树）

```
#include<stdio.h>

char a[2000005];
int t,x,mod;
int n=0;

int main()
{
    scanf("%s",a);
    scanf("%d",&t);
    mod=strlen(a);
    while(t--)
    {
        char op;
        scanf("%s",&op);
        scanf("%d",&x);
        if(op=='A')
        {
            int tmp=x+n;
            printf("%c\n",a[(tmp-1)%mod]);
        }
        else
        {
            n=(n+x+mod)%mod;
        }
    }
    return 0;
}
```

C

这个比较难。我队采用了8-10-6定位的方法，并计算了向量外积（矢量叉乘）来判定向哪边拐。

注意边界循环和eps□

```
#include<stdio.h>
#include<math.h>

double a[25][5];
double edge[25];
int t;
int po6,po8,po106,po108;
```

```
double A,B,C;

double cal(int i,int j)
{
    double tmp=sqrt((a[i][0]-a[j][0])*(a[i][0]-a[j][0])+(a[i][1]-
a[j][1])*(a[i][1]-a[j][1]));
    return tmp;
}

int main()
{
    scanf("%d",&t);
    while(t--)
    {
        int flag=1;
        int i;
        for(i=0;i<20;i++)
        {
            scanf("%lf%lf",&a[i][0],&a[i][1]);
        }
        for(i=0;i<20;i++)
        {
            edge[i]=cal(i,(i+1)%20);
        }
        for(i=0;i<20;i++)
        {
            if(fabs((edge[i] + edge[(i + 1) % 20] + edge[(i + 2) % 20]) -
23) < 0.01)
            {
                if(fabs(edge[i] - 6) < 0.01)
                {
                    po6 = i;
                    po106 = (i + 1) % 20;
                    po108 = (i + 2) % 20;
                    po8 = (i + 3) % 20;
                }
                else
                {
                    po8 = i;
                    po108 = (i + 1) % 20;
                    po106 = (i + 2) % 20;
                    po6 = (i + 3) % 20;
                }
            }
        }
        A = a[po108][1] - a[po106][1];
        B = a[po106][0] - a[po108][0];
        C = -(B * a[po106][1] + A * a[po106][0]);
        if(fabs(A) < 0.01)
        {
            if(a[po6][1] > a[po106][1])
```

```
            {
                if(a[po6][0] < a[po8][0])    flag = 0;
            }
            else
            {
                if(a[po6][0] > a[po8][0])    flag = 0;
            }
        }
        else if (fabs(B) < 0.01)
        {
            if(a[po6][0] < a[po106][0])
            {
                if(a[po6][1] < a[po8][1])    flag = 0;
            }
            else
            {
                if(a[po6][1] > a[po8][1])    flag = 0;
            }
        }
        else if(A * B > 0)
        {
            if(A < 0)
            {
                A = fabs(A);
                B = fabs(B);
                C = -C;
            }
            if(A * a[po6][0] + B * a[po6][1] + C > 0)
            {
                if(a[po106][1] > a[po108][1])    flag = 0;
            }
            else
            {
                if(a[po106][1] < a[po108][1])    flag = 0;
            }
        }
        else
        {
            if(B < 0)
            {
                B = fabs(B);
                A = -A;
                C = -C;
            }
            if(A * a[po6][0] + B * a[po6][1] + C > 0)
            {
                if(a[po106][1] < a[po108][1])    flag = 0;
            }
            else
            {
```

```
                if(a[po106][1] > a[po108][1])    flag = 0;
            }
        }
        if(flag)
        {
            printf("left\n");
        }
        else
        {
            printf("right\n");
        }
    }
    return 0;
}
```

E

动态规划。图论那里的思考略微难一些，参见任韩《图论》第四章（？忘了第几章）第一节“ 对集 ”部分，有完全相同的例题和细节证明，因此对于数学竞赛生应当是小菜一碟。至于动态规划的部分只是熟练活。

注意不要手误。方程敲错了谁也救不了（……）。

（适合作为算法课动态规划的压轴题之一——思维复杂度麻烦的类型）

另外，本题改成C代码的qsort貌似就过不了了，会RE□不知为何。

```
#include<stdio.h>
#include<stdlib.h>

#include<algorithm>

using namespace std;

int a[200005];
int b[100005];
int c[100005];

int main()
{
    int t;
    scanf("%d",&t);
    while(t--)
    {
        int n;
        scanf("%d",&n);
        int i;
        for(i=0;i<n;i++)
        {
            scanf("%d",&a[i]);
        }
```

```
        sort(a,a+n);
        int tmp=a[n-1]-a[0];
        int pmt=n/2-4;
        b[0]=a[4]-a[3];
        c[0]=0;
        for(i=1;i<=pmt;i++)
        {
            b[i]=(b[i-2]>c[i-1]?b[i-2]:c[i-1])+(a[2*i+4]-a[2*i+3]);
            c[i]=(b[i-1]>c[i-1]?b[i-1]:c[i-1]);
        }
        int mmm=(b[pmt]>c[pmt]?b[pmt]:c[pmt]);
        tmp-=mmm;
        printf("%lld\n",2*tmp);
    }
}
```

F

解一次不定方程题目。注意后面情况讨论以及调整为正数的操作。

```
#include<stdio.h>

char isprime[2000010];
int sq[2000010];

long long exgcd(long long a,long long b,long long *x,long long *y)
{
    if(!b)
    {
        (*x)=1;
        (*y)=0;
        return a;
    }
    long long r=exgcd(b,a%b,&(*x),&(*y)),t=(*x);
    (*x)=(*y);
    (*y)=t-a/b*(*y);
    return r;
}

long long gcd(long long a,long long b)
{
    return (b==0)?a:gcd(b,a%b);
}

char cal(long long a,long long b,long long c,long long *x,long long *y)
{
    if(c%gcd(a,b)!=0)
```

```
    {
        return 0;
    }
    long long q=exgcd(a,b,&(*x),&(*y));
    long long v=c/q;
    (*x)*=v;
    (*y)*=v;
    (*y)*=-1;
    return 1;
}

int main()
{
    isprime[0]=isprime[1]=1;
    int i;
    sq[1]=1;
    for(i=2;i<=20000;i++)
    {
        if(!isprime[i])
        {
            int j;
            for(j=i*i;j<=2000000;j+=i)
            {
                isprime[j]=1;
            }
        }
        if(i*i<=2000005)
        {
            sq[i*i]=i;
        }
    }
    for(i=1;i<2000005;i++)
    {
        if(!sq[i])
        {
            sq[i]=sq[i-1];
        }
    }
    int T;
    scanf("%d",&T);
    while(T--)
    {
        long long a,b;
        scanf("%lld%lld",&a,&b);
        if(b==1)
        {
            printf("-1 -1 -1 -1\n");
        }
        else if(a%b==0)
        {
            int x=a/b;
```

```
            printf("%d 1 %d 1\n",x+1,1);
        }
        else if(!isprime[b])
        {
            printf("-1 -1 -1 -1\n");
        }
        else
        {
            int flag=1;
            long long l,r;
            for(l=2,r=sq[b]+2;l<=r+1;l++,r--)
            {
                if(l>=2&&l<b&&b%l==0)
                {
                    long long x=l,y=b/x,c,e;
                    if(cal(y,x,a,&c,&e))
                    {
                        if(c<=0||e<=0)
                        {
                            if(c<=0&&e<=0)
                            {
                                printf("%lld %lld %lld %lld\n",-e+y,y,-
c+x,x);
                            }
                            else if(e<=0)
                            {
                                long long q=(-e)/y+2;
                                printf("%lld %lld %lld
%lld\n",c+x*q,x,e+y*q,y);
                            }
                        }
                        else
                        {
                            printf("%lld %lld %lld %lld\n",c,x,e,y);
                        }
                        flag=0;
                        break;
                    }
                }
                if(r>=2&&r<b&&b%r==0)
                {
                    long long x=r,y=b/x,c,e;
                    if(cal(y,x,a,&c,&e))
                    {
                        if(c<=0||e<=0)
                        {
                            if(c<=0&&e<=0)
                            {
                                printf("%lld %lld %lld %lld\n",-e+y,y,-
c+x,x);
```

```
                                }
                                else if(e<=0)
                                {
                                    long long q=(-e)/y+2;
                                    printf("%lld %lld %lld
%lld\n",c+x*q,x,e+y*q,y);
                                }
                            }
                            else
                            {
                                printf("%lld %lld %lld %lld\n",c,x,e,y);
                            }
                            flag=0;
                            break;
                        }
                    }
                }
                if(flag)
                {
                    printf("-1 -1 -1 -1\n");
                }
            }
        }
        return 0;
}
```

A

```

```

A

```

```

A

```

```