```cpp
#include<bits/stdc++.h>

using namespace std;

struct Point{
    double x,y;
    Point(double x=0,double y=0):x(x),y(y){}
};

const double eps = 1e-10;
const double Inf = 1e9;

typedef Point Vector;

Vector operator + (Vector A,Vector B){ return Vector(A.x+B.x,A.y+B.y); }
Vector operator - (Vector A,Vector B){ return Vector(A.x-B.x,A.y-B.y); }
Vector operator * (Vector A,double p){ return Vector(A.x*p,A.y*p); }
Vector operator / (Vector A,double p){ return Vector(A.x/p,A.y/p); }

bool operator < (const Point &a,const Point &b){
    return a.x < b.x || (a.x == b.x && a.y < b.y);
}

int dcmp(double x){
    if( fabs(x) < eps )  return 0;
    else return x < 0 ? -1 : 1;
}

bool operator == (const Point &a,const Point &b){
    return dcmp(a.x-b.x) == 0 && dcmp(a.y-b.y) == 0;
}

double Dot(Vector A,Vector B){ return A.x*B.x + A.y*B.y; }
double Length(Vector A){ return sqrt(Dot(A,A)); }
double Angle(Vector A,Vector B){ return acos( Dot(A,B)/Length(A)/Length(B) ); }

double Cross(Vector A,Vector B){ return A.x*B.y - A.y*B.x; }
double Area2(Point A,Point B,Point C){ return Cross(B-A,C-A); }
double Len(Point a,Point b){ return sqrt( (a.x-b.x)*(a.x-b.x) + (a.y-b.y)*(a.y-b.y) ); }

Point Ans[5],Text[5];

void ConvexHull(Point *p,int n,Point *ch){
    sort(p,p+n);
    int m = 0;
    for(int i=0;i<n;i++)
    {
        while( m > 1 && Cross(ch[m-1]-ch[m-2],p[i]-ch[m-2]) <= 0 ) m--;
        ch[m++] = p[i];
    }
    int k = m;
    for(int i=n-2;i>=0;i--)
    {
        while( m > k && Cross(ch[m-1]-ch[m-2],p[i]-ch[m-2]) <= 0 ) m--;
        ch[m++] = p[i];
    }

    if( n > 1 )  m--;
    double Min = Inf,Max = -Inf;
    int l=1,r=1,P=1;
    ch[m] = ch[0];
    double L,R,H,D;
    for(int i=0;i<m;i++)
    {
        D = Len(ch[i],ch[i+1]);
        while( dcmp(Cross(ch[i+1]-ch[i],ch[P+1]-ch[i])-Cross(ch[i+1]-ch[i],ch[P]-ch[i])) >= 0 ) P = (P+1)%m;
        while( dcmp( Dot(ch[i+1]-ch[i],ch[r+1]-ch[i]) - Dot(ch[i+1]-ch[i],ch[r]-ch[i]) ) >= 0 ) r = (r+1)%m;
        if( i == 0 )  l = r;
        while( dcmp( Dot(ch[i+1]-ch[i],ch[l+1]-ch[i]) - Dot(ch[i+1]-ch[i],ch[l]-ch[i]) ) <= 0 ) l = (l+1)%m;
        L = Dot(ch[i+1]-ch[i],ch[l]-ch[i])/D;
        R = Dot(ch[i+1]-ch[i],ch[r]-ch[i])/D;
        H = Cross(ch[i+1]-ch[i],ch[P]-ch[i])/D;
        H = H<0?-H:H;
        double S = (R-L)*H;
        if( S < Min )
        {
            Min = S;
            Ans[0] = ch[i] + (ch[i+1]-ch[i])*R/D;
            Ans[1] = Ans[0] + (ch[r]-Ans[0])*(H/Len(ch[r],Ans[0]));
            Ans[2] = Ans[1] - (Ans[0] - ch[i])*((R-L)/Len(Ans[0],ch[i]));
            Ans[3] = Ans[2] - Ans[1] + Ans[0];

        }
    }
    printf("%.5lf\n",Min);
    double Min_y = Inf,Min_x;
    int ii;
    for(int i=0;i<4;i++)
    {
        if( dcmp( Ans[i].y - Min_y ) < 0 )
            Min_y = Ans[i].y,ii = i,Min_x = Ans[i].x;
        else if( dcmp( Ans[i].y - Min_y ) == 0 )
        {
            if( dcmp( Ans[i].x - Min_x ) < 0 )
                Min_y = Ans[i].y,ii = i,Min_x = Ans[i].x;
        }
    }
    for(int i=0;i<4;i++)
    {
        Ans[ii].x = dcmp(Ans[ii].x) == 0 ? 0 : Ans[ii].x;
        Ans[ii].y = dcmp(Ans[ii].y) == 0 ? 0 : Ans[ii].y;
        printf("%.5lf %.5lf\n",Ans[ii].x,Ans[ii].y);
        ii = (ii+1)%4;
    }

}

int n;
```

```
Point p[50050],a[50050];

int main(){
    scanf("%d",&n);
    for(int i=0;i<n;i++)
        scanf("%lf %lf",&p[i].x,&p[i].y);
    ConvexHull(p,n,a);
    return 0;
}

/*

Luogu
P3187 [HNOI2007]最小矩形覆盖

包含所有点的最小矩形

*/
```