

题意

在一个2维平面上有两条传送带，每一条传送带可以看成是一条线段。两条传送带分别为线段AB和线段CD。lxhgww在AB上的移动速度为P，在CD上的移动速度为Q，在平面上的移动速度R。现在lxhgww想从A点走到D点，他想知道最少需要走多长时间

分析

代码

```
1  #include<algorithm>
2  #include<bitset>
3  #include<cstdio>
4  #include<cstring>
5  #include<cstdlib>
6  #include<cmath>
7  #include<deque>
8  #include<iostream>
9  #include<map>
10 #include<queue>
11 #include<set>
12 #include<stack>
13 #include<string>
14 #include<vector>
15 #include<list>
16 #define For(i,a,b) for(int i=(a); i<=(b) ; i++)
17 #define _For(i,a,b) for(int i=(a); i>=(b) ; i--)
18 #define Memset(a,b); memset((a),(b),sizeof((a)));
19 #define Cout(a,b); printf("%d",(a));printf(b);
20 #define Coutc(a,b); printf("%c",(a));printf(b);
21 #define Couts(a,b); printf("%s",(a));printf(b);
22 using namespace std;
23 const int INF = 0x3f3f3f3f;
24 typedef long long LL;typedef unsigned long long ULL;typedef long double
    LDB;
25 inline LL CinLL(){LL x=0,f=1;char ch=getchar();while(ch<'0' || ch>'9'){if(ch=='-'
    )f=-1;ch=getchar();}while(ch>='0'&&ch<='9'){x=x*10+ch-
    '0';ch=getchar();}return x*f;}
26 inline int Cin(){int x=0,f=1;char ch=getchar();while(!isdigit(ch)){if(ch=='-'
    )f=-1;ch=getchar();}while(isdigit(ch))x=x*10+ch-'0',ch=getchar();return f*x;}
```

```

27 const double eps = 1e-3;
28 struct Point{
29     double x,y;
30 };
31 struct Line{
32     Point a,b;
33     double v;
34 }l[3];
35 inline double dis(Point a,Point b){
36     return sqrt((a.x - b.x)*(a.x - b.x) + (a.y - b.y) *(a.y - b.y));
37 }
38 double js(Point p1,Point p2)
39 {
40     double tot = 0.0;
41     tot+=dis(l[0].a,p1)/l[0].v;
42     tot+=dis(p1,p2)/l[2].v;
43     tot+=dis(p2,l[1].b)/l[1].v;
44     return tot;
45 }
46 Point tf1(Point p1)
47 {
48     Point m1,m2;
49     Point low,up;
50     low = l[1].a;
51     up = l[1].b;
52     while(dis(low,up) >eps)
53     {
54         m1.x =low.x+(up.x - low.x) /3.0;m1.y = low.y+(up.y - low.y)/3.0;
55         m2.x =up.x -(up.x - low.x) /3.0;m2.y = up.y -(up.y - low.y)/3.0;
56         double kk = dis(low,up);
57         if(js(p1,m1)<=js(p1,m2))
58             up = m2;
59         else
60             low = m1;
61     }
62     Point res;
63     res.x = (low.x+up.x) * 0.5;
64     res.y = (low.y+up.y) * 0.5;
65     return res;
66 }
67 double choose(Point p1)
68 {
69     Point p2 = tf1(p1);
70     return js(p1,p2);
71 }
72 Point tf0()

```

```

73 {
74     Point m1,m2;
75     Point low,up;
76     low = l[0].a;
77     up = l[0].b;
78     while(dis(low,up) >eps)
79     {
80         m1.x =low.x+(up.x - low.x) /3.0;m1.y = low.y+(up.y - low.y)/3.0;
81         m2.x =up.x -(up.x - low.x) /3.0;m2.y = up.y -(up.y - low.y)/3.0;
82         if(choose(m1)<=choose(m2))
83             up = m2;
84         else
85             low = m1;
86     }
87     Point res;
88     res.x = (low.x+up.x) * 0.5;
89     res.y = (low.y+up.y) * 0.5;
90     return low;
91 }
92 double solve()
93 {
94     Point p1 = tf0();
95     return choose(p1);
96 }
97 int main()
98 {
99     For(i,0,1)
100         cin>>l[i].a.x>>l[i].a.y>>l[i].b.x>>l[i].b.y;
101     cin>>l[0].v>>l[1].v>>l[2].v;
102     double ans = solve();
103     printf("%.2lf\n",ans);
104 }

```