int v,V,w,h,H,i,j;

int main(int c, char \*\*a)

{

V=atoi(a[1]); /\* Volume of bowl \*/

v=atoi(a[2]); /\* Volume of water \*/

for(;++H;) /\* Make the bowl taller \*/

{

for(h=0;h++<H;) /\* Make the water taller \*/

{

for(w=1;h\*h+w\*h-h<v;++w); /\* Make the bowl wider until the water volume matches \*/

if(H\*H+w\*H-H==V) /\* if the bowl volume matches, then we're good \*/

{

for(;H--;) /\* Print out the bowl, one line at a time \*/

{

printf("%\*s",++i,"\\"); /\* Print the left edge \*/

/\* Print the inside (either with air/water, the top of the water, or the bottom of the bowl \*/

for(j=0;j++<w-1+2\*H;)

putchar(H?H==h?'~':32:95);

/\* Print the right edge of the bowl \*/

puts("/");

}

exit(0); /\* die, we're done \*/

}

}

}

}

--------------------------------------------------------------------------------------

#include<stdio.h>

#include<math.h>

int x[20],y[20],r[20],st,ed,n,i,j,k,cases;

char dr[3],mark[20];

double dc[20][20],arg[20][20],len[20][20],xi[20][20],yi[20][20],xj[20][20],yj[20][20];

double pxa[20],pya[20],pxb[20],pyb[20],eps,pi,d,cur,ans;

double dis(double x,double y)

{

return sqrt(x\*x+y\*y);

}

double angle(double x,double y)

{

if(y>0)

return acos(x/dis(x,y));

if(y<0)

return 2\*pi-acos(x/dis(x,y));

if(x>0)

return 0;

return pi;

}

int cross(double xa,double ya,double xb,double yb,double xc,double yc,double xd,double yd)

{

double cra,crb;

cra=(xb-xa)\*(yc-ya) - (xc-xa)\*(yb-ya);

crb=(xb-xa)\*(yd-ya) - (xd-xa)\*(yb-ya);

if(cra\*crb>-eps)

return 0;

cra=(xd-xc)\*(ya-yc) - (xa-xc)\*(yd-yc);

crb=(xd-xc)\*(yb-yc) - (xb-xc)\*(yd-yc);

if(cra\*crb>-eps)

return 0;

return 1;

}

void go(int v,double a)

{

int j,k;

double add;

for(j=0;j<n;j++)

if(mark[j]&&len[v][j]>0)

{

for(k=0;k<i;k++)

if(cross(pxa[k],pya[k],pxb[k],pyb[k],xi[v][j],yi[v][j],xj[v][j],xj[v][j]))

break;

if(k<i)

continue;

pxa[i]=xi[v][j];

pya[i]=yi[v][j];

pxb[i]=xj[v][j];

pyb[i]=yj[v][j];

add=len[v][j];

if(a>-1)

if(r[v]<0)

{

if(arg[v][j]>a-eps)

add-=r[v]\*(arg[v][j]-a);

else

add-=r[v]\*(2\*pi+arg[v][j]-a);

}

else

{

if(a>arg[v][j]-eps)

add+=r[v]\*(a-arg[v][j]);

else

add+=r[v]\*(2\*pi+a-arg[v][j]);

}

mark[j]=0;

cur+=add;

i++;

if(cur<ans)

if(j==ed)

ans=cur;

else

go(j,arg[v][j]);

i--;

cur-=add;

mark[j]=1;

}

}

int main()

{

eps=1e-8;

pi=acos(-1.0);

while(scanf("%d",&n)&&n)

{

for(i=0;i<n;i++)

{

scanf("%d%d%d %s",&x[i],&y[i],&r[i],dr);

if(dr[1])

r[i]=-r[i];

mark[i]=1;

}

scanf("%d%d%lf",&st,&ed,&d);

mark[st]=0;

for(i=0;i<n;i++)

for(j=0;j<n;j++)

dc[i][j]=dis(x[j]-x[i],y[j]-y[i]);

for(i=0;i<n;i++)

for(j=0;j<n;j++) if(i!=j)

{

arg[i][j]=angle(x[j]-x[i],y[j]-y[i]) + acos((r[i]-r[j])/dc[i][j]);

if(arg[i][j]>=2\*pi)

arg[i][j]-=2\*pi;

if(arg[i][j]<0)

arg[i][j]+=2\*pi;

len[i][j]=sqrt(dc[i][j]\*dc[i][j] - (r[j]-r[i])\*(r[j]-r[i]));

if(len[i][j]>d)

len[i][j]=-1;

else

{

xi[i][j]=x[i]+r[i]\*cos(arg[i][j]);yi[i][j]=y[i]+r[i]\*sin(arg[i][j]);

xj[i][j]=x[j]+r[j]\*cos(arg[i][j]);yi[i][j]=y[j]+r[j]\*sin(arg[i][j]);

for(k=0;k<n;k++) if(k!=i&&k!=j)

{

if((xj[i][j]-xi[i][j])\*(x[k]-xi[i][j]) + (yj[i][j]-yi[i][j])\*(y[k]-yi[i][j])<=0)

continue;

if((xi[i][j]-xj[i][j])\*(x[k]-xj[i][j]) + (yi[i][j]-yj[i][j])\*(y[k]-yj[i][j])<=0)

continue;

if(fabs((xj[i][j]-xi[i][j])\*(y[k]-yi[i][j]) - (yj[i][j]-yi[i][j])\*(x[k]-xi[i][j]))/len[i][j] >labs(r[k])+eps)

continue;

len[i][j]=-1;

break;

}

}

}

cur=0;

ans=2e9;

i=0;

go(st,-2);

printf("Cases %d: ",++cases);

if(ans<1e9)

if(int(ans\*100+0.5)%10)

printf("length = %2.lf\n",ans);

else

if(int(ans\*100+0.5)%100)

printf("length = %1.lf\n",ans);

else

printf("length = %0.lf\n",ans);

else

printf("Cannot reach destination shaft");

}

return 0;

}

-------------------------------------------------------------------------------------------------------------------------------------

1. #include<iostream>
2. #include<vector>
3. #include<string.h>
4. #include<string>
5. #include<stdio.h>
6. #include<stdlib.h>
7. #include<algorithm>
8. **using** **namespace** std;
9. **const** **int** MaxN=109;
10. **const** **int** INF=1000000000;
11. **int** N,M;
12. vector<**int**> e[MaxN];
13. vector<pair<**int**,**int**>> Go[MaxN];
14. **int** A[MaxN],B[MaxN],F[MaxN],G[MaxN];
16. **void** init()
17. {
18. **for**(**int** i=1;i<=N;i++)
19. e[i].clear();
20. **for**(**int** i=1;i<=N;i++)
21. {
22. **int** x;
23. scanf("%d%d%d",&A[i],&B[i],&x);
24. B[i]+=x;
25. **if**(A[i]>=B[i]);
26. **else**
27. A[i]=B[i];
28. }
29. **for**(**int** i=1;i<N;i++)
30. {
31. **int** u,v;
32. scanf("%d%d",&u,&v);
33. e[u].push\_back(v);
34. e[v].push\_back(u);
35. }
36. }
38. **void** Dfs(**int** fa,**int** x)
39. {
40. F[x]=A[x];
41. G[x]=B[x];
42. Go[x].clear();
43. **for**(**int** i=0;i<e[x].size();i++)
44. **if**(e[x][i]!=fa)
45. {
46. Dfs(x,e[x][i]);
47. Go[x].push\_back(make\_pair(G[e[x][i]],F[e[x][i]]));
48. }
49. sort(Go[x].begin(),Go[x].end());
50. **for**(**int** i=Go[x].size()-1;i>=0;i--)
51. {
52. **if**(F[x]-G[x]<Go[x][i].second)
53. F[x]=G[x]+Go[x][i].second;
54. G[x]+=Go[x][i].first;
55. }
56. }
58. **void** solve()
59. {
60. **int** ans=INF;
61. **for**(**int** i=1;i<=N;i++)
62. {
63. **for**(**int** j=1;j<=N;j++)
64. F[j]=G[j]=-1;
65. Dfs(0,i);
66. **if**(F[i]<ans)
67. ans=F[i];
68. }
69. printf("%d\n",ans);
70. }
72. **int** main()
73. {
74. **int** T=0;
75. **while**(1==scanf("%d",&N)&&N!=0)
76. {
77. T++;
78. init();
79. printf("Case %d: ",T);
80. solve();
81. }
82. **return** 0;
83. }