***Гульмира Нурбекова, Тлеумагамбетова Данара, Дыбыспаева Нуржамал***

***(Астана, Казахстан)***

**Олимпиадалық есептерді шешудегі геометриялық есептердің орны**

Есептеу геометриясы- геометриялық тапсырмаларды шешу үшін алгоритмдерді қарастыратын информатиканың бір бөлігі. Мұндай есептер компьютерлік графика, интегралды сызбаларды жабалау, техникалық құрылғылар сияқты тақырыптарда кездеседі. Осындай типті есептерге нүкте, кесінділер, көпбұрыштар жатады. Осындай есептердің нәтижесі қандай да бір сұрақтың жауабы немесе геометриялық объект болуы мүмкін.

Кеңістіктегі кез келген нүкте вектор болып табылады. Мұндайвектор ноль мәнді вектор деп аталады. Нольге тең емес вектордың ұзындығы АВ кесіндінің ұзындығы деп аталады. Екі нольге тең емес векторлар коллинеар вектор деп аталады, егер де де екеуі де бір түзудің бойында жатса немесе паралель түзулердің бойында жатса. Егере де АВ және СД векторларыколлинеар деп аталса, онда олардың сәулелері бір бағытта және оны бір бағыттағы векторлар деп атайды.

А,В,С, нүктелерінің координаталары бойынша үшбұрыштың ауданын табатын есеп қарастырайық.

int x1,x2,x3,y1,y2,y3;

double s,p,a,b,c;

cout<<"\n x1="; cin>>x1;

cout<<"\n x2="; cin>>x2;

cout<<"\n x3="; cin>>x3;

cout<<"\n y1="; cin>>y1;

cout<<"\n y2="; cin>>y2;

cout<<"\n y3="; cin>>y3;

a = sqrt((x1-x2)^2+(y1-y2)^2);

b = sqrt((x1-x3)^2+(y1-y3)^2);

c = sqrt((x2-x3)^2+(y2-y3)^2);

p = (a+b+c)/2;

s=(p\*(p-a)\*(p-b)(p-c))^(1/2); cout<<"s="<<s;

}

Бұл есептерді жеңіл типті есептер деп қарастыруға болады. Енді олимпиадалық деңгейдегі есептерді шешіп көрейік.

№2 мысал. Егер де берілген нүкте арқылы горизонталь түзу жүргізсек, онда осы түзудің көпбұрышпен қиылысуы мүмкін. Берілген нүктеден абциссасы кіші болатындай барлық қиылысуларды қарастырайық. Егер де осы нүктелер саны тақ болса, онда берілген нүкте көпбұрыштың ішінде жатыр.

Шығарылуы:

bool point\_in\_polygon (point t, vector < point > p)

{

int i, j;

int count = 0;

for (i = 0; i < p.size(); ++ i)

{

j = (i + 1) % p.size();

if (min (p[i].y, p[j].y) < t.y && t.y <= max (p[i].y, p[j].y) &&

ccw (min\_py (p[i], p[j]), max\_py (p[i], p[j]), t))

{// егер де нүктенің проекциясы кесіндіде жатса және нүкте кесіндінің оң //жағында жатса, онда нүктенің көпбұрышқа жату санын көбейтеміз

++ count;

}

}

return count % 2;

}

Ван Гог алгоритмін қарастырайық. Ван Гог алгоритмінің негізгі идеясы үшбұрышты кезекпен қию. Оның функциясы мына түрде болады:

VanGog(p,n,T), мұндағы

Т- массив.

bool isTriangCut(point p[],int n,int i,int j,int k)

{

if(cw(p[i],p[j],p[k]))return 0;

for(int m=0;m<n;++m)

if(m != i && m != j && m != k && inTriangPoint(p[i],p[j],p[k],p[m]))

return 0;

return 1;

}

int VanGog(point p[],int n,int T[][3])

{

int \*l=new int[n];

int \*r=new int[n];

int i,j,count=0;

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

{

l[i]=(i-1+n)%n;

r[i]=(i+1+n)%n;

}

for(i=r[n-1];count<n-2;i=r[i])

if(isTriangCut(p,n,l[i],i,r[i]))

{

T[count][0]=l[i];

T[count][1]=i;

T[count][2]=r[i];

l[r[i]]=l[i];

r[l[i]]=r[i];

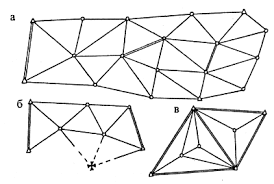
++count;

}

return count;}

Есептеу геометрия бөлімінде жиі қарастырылатын бөлімдердің бірі триангуляция.

Триангуляция(лат. traіngulum – үшбұрыш) — геодезиялық тірек пункттер (нүктелер) торын құру әдістерінің бірі және осы әдістермен жасалған сол тордың өзі. Ол бір-бірімен жалғасып жатқан үшбұрыштар жүйесінен немесе тордан құралады. Триангуляцияда берілген координаттар жүйесінде үшбұрыштар төбелерінің орны анықталады. Ол үшін бастапқы үшбұрыштың қабырғасының біреуі жүргізілген өлшеулер нәтижесі бойынша есептеледі және әрбір үшбұрыштың үш бұрышы өлшенеді. Егер бастапқы үшбұрыштың қабырғасы тікелей өлшеулер арқылы анықталса, онда ол базистік қабырға деп аталады. Триангуляция тәсілі Жер пішіні мен мөлшерін анықтау, жер қыртысының горизонталь қозғалыстарын зерттеу; әр түрлі топографиялық түсірулер мен геодезиялық жұмыстарды негіздеу, т.б. үшін кеңінен қолданылады.



Сурет 1- Триангуляция мысалы

№ 3 мысал. Көпбұрыш және нүктелер саны берілген. Берілген көпбұрышты үшбұрыштарға бөліңіз.

bool IsCrossS(point p1,point p2,point p3,point p4)

{ return SignAreaTriang(p1,p2,p3)\*SignAreaTriang(p1,p2,p4) < 0 &&

SignAreaTriang(p3,p4,p1)\*SignAreaTriang(p3,p4,p2) < 0;}

int count\_triangle=0;

void path\_triangle(polygon p,int K[][100],int i,int j,point T[][3])

{ if(i == j || i == (j-1)%p.n || i == -1 || j == -1)return;

T[count\_triangle][0]=p.p[i];

T[count\_triangle][1]=p.p[K[i][j]];

T[count\_triangle][2]=p.p[j];

++count\_triangle;

path\_triangle(p,K,i,K[i][j],T);

path\_triangle(p,K,K[i][j],j,T);}

int MinChordTriangulate(polygon p,point T[][3])

{ double D[100][100]={0},M[100][100]={0};

int i,j,k,s,K[100][100]={0};

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

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

{ D[i][j]=(DiagInPolygon(p.p,p.n,i,j) || i == (j+1)%p.n || i == (j-1)%p.n ? dist(p.p[i],p.p[j]) : 1e9);

K[i][j]=-1; }

for(i=0;i<p.n-1;++i)

{ M[i][i]=0; M[i][i+1]=D[i][i+1]; }

M[i][i]=0;

for(s=2;s<p.n;++s)

for(i=0;i<p.n-s;++i)

{ j=(i+s)%p.n;

M[i][j]=1e7;

for(k=i+1;k<j;++k)

{ double t=M[i][k]+M[k][j]+D[i][k]+D[k][j]+D[i][j];

if(t<M[i][j])

{ M[i][j]=t;

K[i][j]=k; } } }

path\_triangle(p,K,0,p.n-1,T);

return count\_triangle;}

**Қолданылған әдебиеттер тізімі:**

1. Аравинд Корера, Стивен Фрейзер, Сэм Джентайл, Ниранджан Кумар, Скотт Маклин, Саймон Робинсон, Д-р П.Г. Саранг - Visual C++ .NET пособие для разработчиков С++
2. «Қазақстан»: Ұлттық энцклопедия / Бас редактор Ә. Нысанбаев – Алматы «[Қазақ энциклопедиясы»](https://kk.wikipedia.org/wiki/%D2%9A%D0%B0%D0%B7%D0%B0%D2%9B_%D1%8D%D0%BD%D1%86%D0%B8%D0%BA%D0%BB%D0%BE%D0%BF%D0%B5%D0%B4%D0%B8%D1%8F%D1%81%D1%8B) Бас редакциясы, 1998 [ISBN 5-89800-123-9](https://kk.wikipedia.org/wiki/%D0%90%D1%80%D0%BD%D0%B0%D0%B9%D1%8B:%D0%9A%D1%96%D1%82%D0%B0%D0%BF_%D2%9B%D0%B0%D0%B9%D0%BD%D0%B0%D1%80%D0%BB%D0%B0%D1%80%D1%8B/5898001239), VIII том