12 - AMALIY MASHG‘ULOT. *HISOBLASH GEOMETRIYASI ALGORITMLARI.*

Ishdan maqsad: Talabalarda hisoblash geometriyasi algoritmlari bo‘yicha ma‘lumotlar berish va bilimini oshirish. C++ dasturlash tilida mavzuga oid masalalarning dasturini tuzish.

Nazariy qism: Hisoblash geometriyasi (Computational Geometry) matematika va kompyuter injineringi yo’nalishida geometrik ma’lumotlarni hisoblash va tahlil qilish uchun ma’lumotlar aniqlash va ma’lumotlar o’zgaruvchanliklarini yechish bilan shug’ullanadi. Bu soha matematik va kompyuter ilmi bilan aloqador bo’lib, algoritmlar, axborot nazoratlari, va dasturlash tilining bir qismi sifatida yaxshi tushunchalar talab qiladi.

Hisoblash geometriyasi, quyidagi misollar uchun yordam beradi:

1. ***3D Modellash:*** 3D obyektlarni modellash va ularning geometrik xususiyatlarini aniqlashda hisoblash geometriyasidan foydalaniladi. Bu kompyuter grafikasi, animatsiya va virtual reallik sohalarida juda muhimdir.

2. ***Nazorat va Navigatsiya:*** GPS tizimlari va navigatsiya dasturlari geografik joylashuvning aniqlanishi, navigatsiya yo’nalishlari uchun hisoblash geometriyasidan foydalanadi.

3. Robototexnika: Robotlarning harakat yo’nalishlari, obyektlarni aniqlash va navigatsiyasi uchun hisoblash geometriyasidan foydalaniladi.

4. GIS (Geographic Information Systems): Geografik axborot tizimlari ko’p toifa ma’lumotlarni jamlash, to’plash, tasvirlash va aniqlashda hisoblash geometriyasidan foydalanadi.

5. Texnologiya va Inshootlar: Qurilishlarda va inshootlar sohasida tasavvurlarini amalga oshirish uchun hisoblash geometriyasidan foydalaniladi. Misol uchun, binolar va strukturalar yaratish, transport tarmoqlarini tashkil etish va energiya axborotlari tahlili.

6. Biologiya va Kimyo: Biologiyaviy va kimyoviy strukturalarni tahlil qilishda hisoblash geometriyasidan foydalaniladi, jumladan, molekulyar biologiyada va kimyoviy sintezda.

7. Xavfsizlik va Soha Xavfsizligi: Soha xavfsizligi va xavfsizlik nazorat tizimlarida hisoblash geometriyasidan foydalaniladi, masalan, harakat sensorlari va ko’rsatkichlarini tahlil qilish.

8. Telekommunikatsiya: Kabellarning joylashuvi, qo’rg’oshinlar va boshqa telekommunikatsiya tarmoqlarini rivojlantirishda hisoblash geometriyasidan foydalaniladi.

Hisoblash geometriyasi kompyuter dasturlash va matematikning integratsiyasiga misoldir va ko’pgina masalalar uchun qiyinchiliklar tug’diradi. Uning muhim miqyosdagi amaliyoti

va teorik asoslari bor va bu soha endi kompyuter dasturlashning muhim bo'lgan qismlaridan biri hisoblanadi.

Amaliy qism:

1-masala: Nuqta va liniya orasidagi minimum masofa topish masalasi, matematikada "n-ta nuqta bilan liniya orasidagi minimum masofa topish" masalasi sifatida ham mashhur. Ushbu masalani ishlab chiqish uchun (Closest Point) algoritmi ishlatiladi.

Quyidagi C++ kodida, bir liniyaning (Ax, Ay)-(Bx, By) nuqtalari berilgan bo'lsa va n-ta nuqta (Px, Py) berilgan bo'lsa, liniya va n-ta nuqta orasidagi minimum masofa topiladi.

```
#include <iostream>
#include <cmath>
using namespace std;
// Funksiya, nuqta A va B orasidagi masofani hisoblaydi
double distance(int Ax, int Ay, int Bx, int By, int Px, int Py) {
   // Liniya uzunligi
   double lineLength = sqrt((Bx - Ax) * (Bx - Ax) + (By - Ay) * (By - Ay));
   // Agar liniya uzunligi nolga teng bo'lsa, n-ta nuqta liniya bo'ylab emas
   if (lineLength == 0)
      return sqrt((Px - Ax) * (Px - Ax) + (Py - Ay) * (Py - Ay));
   // Liniyaning parametrik tavsifi
   double u = ((Px - Ax) * (Bx - Ax) + (Py - Ay) * (By - Ay)) / (lineLength *
lineLength);
   // Agar u qiymati 0 dan katta va 1 dan kichik bo'lsa, n-ta nuqta liniya
orasida joylashgan
   if (u >= 0 && u <= 1)
      return abs((Bx - Ax) * (Ay - Py) - (Ax - Px) * (By - Ay)) / lineLength;
   // A va B nuqtalari orasidagi minimum masofa topilishi
   return min(sqrt((Px - Ax) * (Px - Ax) + (Py - Ay) * (Py - Ay)),
         sqrt((Px - Bx) * (Px - Bx) + (Py - By) * (Py - By)));
}
int main() {
   int Ax, Ay, Bx, By, Px, Py;
   cout << "A nuqtaning koordinatalarini kiriting: ";
   cin >> Ax >> Ay;
   cout << "B nuqtaning koordinatalarini kiriting: ";
   cin >> Bx >> By;
   cout << "P nuqtaning koordinatalarini kiriting: ";
   cin >> Px >> Py;
   double minDistance = distance(Ax, Ay, Bx, By, Px, Py);
   cout << " P nuqtasi va AB chizig'i orasidagi minimal masofa: " <<
minDistance << endl;
   return 0;
}
```

Bu kod liniya ustida n-ta nuqta yoki nuqta A va B orasidagi minimum masofani topadi. Ushbu algoritma geometrik masalalarni yechishda va ilovalarni yaratishda juda foydali bo'ladi.

2-masala: Diametr va radiusni topish uchun, uchrashuvning eng uzoq nuqtalarini aniqlash kerak. Ushbu nuqtalar ustida boshqa boshqa nuqtalar bilan uzunliklarini hisoblash va eng katta uzunlikni topish bilan hisoblanadi.

Quyidagi C++ kodida, nuqtalarning koordinatalari berilgan bo'lsa, eng uzoq nuqtalarni topish uchun Brute Force (quvvatli urinish) algoritmasi ishlatiladi:

```
#include <iostream>
#include <cmath>
using namespace std;
// Funksiya, ikki nuqta orasidagi masofani hisoblaydi
double distance(int x1, int y1, int x2, int y2) {
    return sqrt((x2 - x1) * (x2 - x1) + (y2 - y1) * (y2 - y1));
}
// Funksiya, diametr va radiusni topadi
void findDiameterAndRadius(int x[], int y[], int n) {
    double maxDistance = 0; // Eng uzoq masofa
    double currentDistance; // Joriy masofa
    int point1, point2; // Eng uzoq nuqtalar indekslari
    // Barcha nuqtalar orasidagi eng uzoq masofani topish
    for (int i = 0; i < n; ++i) {
        for (int j = i + 1; j < n; ++j) {
            currentDistance = distance(x[i], y[i], x[j], y[j]);
            if (currentDistance > maxDistance) {
                maxDistance = currentDistance;
                point1 = i;
                point2 = j;
            }
        }
    }
    // Diametr va radiusni chiqarish
    cout << "Diametr: " << maxDistance << endl;
    cout << "Radius: " << maxDistance / 2 << endl;
    cout << "Eng uzoq nuqtalar: (" << x[point1] << ", " << y[point1] << ")
va (" << x[point2] << ", " << y[point2] << ")" << endl;
}
int main() {
    int n;
    cout << " Ballar sonini kiriting: ";
```

```
    cin >> n;
    int x[n], y[n];
    cout << "Nuqtalarning koordinatalarini kiriting:" << endl;
    for (int i = 0; i < n; ++i) {
        cout << "Point " << i + 1 << ": ";
        cin >> x[i] >> y[i];
    }
    findDiameterAndRadius(x, y, n);
    return 0;
}
```

Bu dasturda, foydalanuvchi nuqtalar sonini kiritadi va ulardan eng uzoq masofadagi nuqtalarni aniqlaydi. Diametr va radius esa shu eng uzoq nuqtalarning masofasining nisbati bilan topiladi.

3D obyektlarni modellash uchun ko'plab dasturlash tillari va kutubxonalar mavjud, ammo ko'p o'quvchilar OpenGL va DirectX kutubxonalari bilan ishlaganlar. Bu kutubxonalar grafik interfeyslarni yaratish, 3D obyektlarni joylash va ularni rasmga olishda yordam beradi.

Quyidagi C++ dasturida, OpenGL kutubxonasidan foydalaniladi, bu dastur yagona qizil kvadratni chizadi:

```
#include <GL/glut.h>
void display() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glLoadIdentity();
    // Kvadratni chizish
    glBegin(GL_QUADS);
        glColor3f(1.0, 0.0, 0.0); // Qizil rang
        glVertex3f(-0.5, -0.5, 0.0); // Pastki yuqori
        glVertex3f(0.5, -0.5, 0.0); // Yuqori yuqori
        glVertex3f(0.5, 0.5, 0.0); // Yuqori chap
        glVertex3f(-0.5, 0.5, 0.0); // Pastki chap
    glEnd();
    glFlush();
}
void reshape(int w, int h) {
    glViewport(0, 0, w, h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(45.0, (double)w / (double)h, 1.0, 200.0); // Burchaklar
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glTranslatef(0.0, 0.0, -5.0); // Obyektni boshlang'ich joylash
}
```

```
int main(int argc, char **argv) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_RGB | GLUT_SINGLE | GLUT_DEPTH);
    glutInitWindowSize(500, 500);
    glutCreateWindow("3D Obyekt Modellash");
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glEnable(GL_DEPTH_TEST);
    glutMainLoop();
    return 0;
}
```

Bu dastur OpenGL kutubxonasidan foydalanadi va yagona qizil kvadratni chizadi. Agar kundalik kompyuter grafikasi yaratishni o'rganmoqchi bo'lsangiz, OpenGL va DirectX bilan ishlashni o'rganishingiz lozim.

Mustaqil bajarish uchun topshiriqlar

1. Berilgan ikki nuqta orasidagi masofani hisoblash.
2. Berilgan nuqta va liniya orasidagi eng yaqin masofani topish.
3. Berilgan nuqta va doira orasidagi eng yaqin masofani topish.
4. Berilgan ikki liniya orasidagi kesishuvni aniqlash.
5. Berilgan ikki doira orasidagi kesishuvni aniqlash.
6. Yuzalar orasidagi qatlamalar sonini hisoblash.
7. Berilgan shaklning balandligini va engligini hisoblash.
8. Poligonning yuzasini va perimetrini hisoblash.
9. Berilgan to'rtburchak markazini aniqlash.
10. Berilgan nuqta va poligon orasidagi eng yaqin nuqtani topish.
11. Berilgan nuqta va poligonning ichida yoki tashqarida joylashganligini aniqlash.
12. Poligonning Convex Hull'ini topish.
13. Berilgan ikki liniya orasidagi nuqta orasidagi masofani hisoblash.
14. Poligonning shaklini qayta tiklash (polygon triangulation).
15. Voronoi diagrammasini chizish.
16. Berilgan ikki shaklini kesishish nuqtasini hisoblash.
17. Berilgan nuqta va poligon tomonlari orasidagi masofani topish.

18. O’zgaruvchan geometrik algoritmni yaratish (masalan, nuqta to’plamining Convex Hull’ini hisoblash).

19. Geometrik algoritmni istiqomatli optimallashtirish.

20. Amaliyotda hisoblash geometriyasi muammolarini hal qilish uchun boshqa dasturlash tillarini (masalan, Python yoki Java) ham ishlatish.