Звіт з предмету Алгоритми та аналіз складності
Пустовіта Дмитра
КА-32

1. **Умова Задачі**

   На прямій $\begin{pmatrix} x+2y+z-1=0 \\ 3x-y+4-29=0 \end{pmatrix}$ знайти точку, рівновіддалену від двох даних точок A (3, 11, 4) та B(-5, -13, -2).

2. **Математичне розв'язання задачі**

   Геометричне місце точок, рівновіддалених від двох наданих точок у просторі – це площина, перпендикулярна до відрізка, кінцями якого є ці точки. Знайдемо рівняння цієї площини (α) :

$$\text{т.} M\begin{pmatrix} x \\ y \\ z \end{pmatrix} \text{ є } \alpha \text{ тоді і тільки тоді, коли } |MA| = |MB|,$$

   що рівносильно рівнянню

$$(x-A_x)^2 + (y-A_y)^2 + (z-A_z)^2 = (x-B_x)^2 + (y-B_y)^2 + (z-B_Z)^2$$

   Розкривши дужки отримуємо що рівняння площини α виглядає наступним чином:

$$(B_x-A_x)x + (B_y-A_y)y + (B_Z-A_Z)z + \frac{1}{2}(A_x^2 + A_y^2 + A_z^2 - B_x^2 - B_y^2 - B_z^2) = 0$$

   Для заданих точок зокрема $4x + 12y + 3z + 13 = 0;$

   Отримали пряму $\begin{pmatrix} x+2y+z-1=0 \\ 3x-y+4-29=0 \end{pmatrix}$ та площину $4x + 12y + 3z + 13 = 0$, що й дають в перетині шукану точку. Перевіривши чи дана площина перетинається прямою , координати шуканої точки можна отримати як розв'язок системи 3 лінійних алгебраїчних рівнянь, розв'яжемо їх методом Крамера: $\begin{cases} x+2y+z-1=0 \\ 3x-y+4-29=0 \\ 4x+12y+3z+13=0 \end{cases}$

$$\Delta = \begin{vmatrix} 1 & 2 & 1 \\ 3 & -1 & 4 \\ 4 & 12 & 3 \end{vmatrix} = 3 \qquad \Delta 1 = \begin{vmatrix} 1 & 2 & 1 \\ 29 & -1 & 4 \\ -13 & 12 & 3 \end{vmatrix} = 6;$$

$$\Delta 2 = \begin{vmatrix} 1 & 1 & 1 \\ 3 & 29 & 4 \\ 4 & -13 & 3 \end{vmatrix} = -9; \qquad \Delta 3 = \begin{vmatrix} 1 & 2 & 1 \\ 3 & -1 & 29 \\ 4 & 12 & -13 \end{vmatrix} = 15;$$

$$x = \frac{\Delta_1}{\Delta} = 2 \quad y = \frac{\Delta_2}{\Delta} = -3 \quad z = \frac{\Delta_3}{\Delta} = 5$$

3. Результати роботи

```
X=2
Y=-3
Z=5
```

4. Код програми

```
#include "stdafx.h"
#include "fstream"
#include "iostream"
#include "math.h"
#include "geometry.h"
#define eps 0.00001
using namespace std;

ofstream fout("out.txt");
ifstream fin("in.txt");

TPlane plane1, plane2, plane3;
TPoint A, B, X;

void error(int n){
	fout << "Error # " << n << ":\n";
	switch (n) {
	case 1:
		fout << "Plane is incorect (|A|+|B|+|C|=0)";
		break;
	case 2:
		fout << "Dots are very close";
		break;
	case 3:
		fout << "Planes are paralel";
		break;
	case 4:
		fout << "Dots locasion is incorrect";
		break;
	}
	exit(n);
}

void inPoint(ifstream &fin, TPoint *point){
	fin >> point->x >> point->y >> point->z;
}

void outPoint(ofstream &fout, TPoint *point){
	fout << "X=" << point->x << "\nY=" << point->y << "\nZ=" << point->z;
}

void inPlane(ifstream &in, TPlane *plane){
	fin >> plane->A >> plane->B >> plane->C >> plane->D;
	if ((fabs(plane->A) < eps) && (fabs(plane->B) < eps) && (fabs(plane->C) < eps))
		error(1);
}

void createPlane(TPlane *plane, double a, double b, double c, double d){
	plane->A = a;
	plane->B = b;
	plane->C = c;
```

```
	plane->D = d;
	if ((fabs(plane->A) < eps) && (fabs(plane->B) < eps) && (fabs(plane->C) < eps))
		error(2);
}

int paralPlanes(TPlane *p1, TPlane *p2){
	if ((fabs((p1->A)*(p2->B) - (p2->A)*(p1->B)) < eps) & (fabs((p1->A)*(p2->C) - (p2->A)*(p1->C) )< eps) & (fabs((p1->B)*(p2->C) - (p2->B)*(p1->C)) < eps)) return 1;
	return 0;}

void correct(TPlane *plane1, TPlane *plane2, TPlane *plane3){
	if (paralPlanes(plane1, plane2)) error(3);
	if ((paralPlanes(plane1, plane3)) | (paralPlanes(plane2, plane3))) error(4);
}

TPoint PlanesToPoint(TPlane *p1, TPlane *p2, TPlane *p3){
	double d0, d1, d2, d3;
	d0 = Deter3(p1->A, p1->B, p1->C, p2->A, p2->B, p2->C, p3->A, p3->B, p3->C);
	d1 = Deter3(-p1->D, p1->B, p1->C, -p2->D, p2->B, p2->C, -p3->D, p3->B, p3->C);
	d2 = Deter3(p1->A, -p1->D, p1->C, p2->A, -p2->D, p2->C, p3->A, -p3->D, p3->C);
	d3 = Deter3(p1->A, p1->B, -p1->D, p2->A, p2->B, -p2->D, p3->A, p3->B, -p3->D);
	TPoint X;
	X.x = d1 / d0;
	X.y = d2 / d0;
	X.z = d3 / d0;
	return X;
}
TPoint PointLine(TPoint *A, TPoint *B, TLine *L){
	TPlane p3;
	createPlane(&p3, B->x - A->x, B->y - A->y, B->z - A->z, (pow(A->x, 2) + pow(A->y, 2) + pow(A->z, 2) - pow(B->x, 2) - pow(B->y, 2) - pow(B->z, 2)) / 2);
	correct(&L->p1, &L->p2, &p3);
	return PlanesToPoint(&L->p1,&L->p2,&p3);
};

int _tmain(int argc, _TCHAR* argv[])
{
	inPlane(fin, &plane1);
	inPlane(fin, &plane2);
	inPoint(fin, &A);
	inPoint(fin, &B);
	L.p1 = plane1;
	L.p2 = plane2;
	X = PointLine(&A,&B,&L);
	outPoint(fout, &X);
	fin.close();
	fout.close();
	system("Pause");
	return 0;
}
```

5. Висновок

Було створено програму для розв’язання поставленої задачі в загальному випадку, з довільними параметрами та обробкою помилок.