**Федеральное государственное автономное образовательное учреждение**

**высшего образования**

**«Российский университет дружбы народов имени Патриса Лумумбы»**

Инженерная академия

*Департамент механики и процессов управления*

**ОТЧЕТ**

|  |  |
| --- | --- |
| **По** | **Лабораторной работе** |

|  |  |
| --- | --- |
| **Направление:** | **01.03.02 Прикладная математика и информатика** |
|  | (код направления / название направления) |
| **Профиль:** | **Математические методы механики полета ракет-носителей и космических аппаратов** |
|  | (название профиля) |

|  |  |
| --- | --- |
| **Тема:** | **Уравнение Кеплера** |
|  | (название лабораторной / курсовой) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Выполнено студентом:** | **Субботина Алёна Андреевна** | | |
|  | (ФИО) | | |
| **Группа:** |  | **Ипмбд-01-22** |  |
| **№ студенческого:** | | 1132226171 |  |

**Москва, 2023**

**Теоретическая часть для отчета**

Вариант:4

Миссия: Луна-10

**Вводные данные:**

R = 1737 км – радиус луны

H1 = 1017 км – высота в точке апоцентра

H2 = 350 км – высота в точке перицентра

ra = H1 +R = 2 754 км – радиус апоцентра

rp = H+ R = 2 087 км – радиус перицентра

**Расчёты:**

𝑎 = ra+rp /2 = (2754+2087)/2 = 2420,5 – большая полуось орбиты

e = ra-rp /2a = (2754-2087)/(2\*240,5) = 0,139193 – эксцентриситет орбиты

# Таблица соответствий переменных задачи и идентификаторам

|  |  |  |
| --- | --- | --- |
| **Наименование переменной** | **Обозначение в документе** | **Обозначение в программе** |
| гравитационный параметр Луны, / |  |  |
| большая полуось орбиты, км |  |  |
| эксцентриситет орбиты |  |  |
| Бесконечно малая | 𝜀 | 𝜀 |
| фокальный параметр орбиты, км |  |  |
| Средняя угловая скорость, км/с | *n* | *n* |
| период движения по орбите, c |  |  |
| текущее время, с |  |  |
| Эксцентрическая аномалия, рад |  |  |
| угол истинной аномалии, рад |  |  |
| Истинная аномалия, рад | 𝜗 | true\_anomaly |
| Средняя аномалия, рад | *M* | *M* |
| текущий радиус-вектор, км |  |  |
| Радиус апогея | a | *r\_a* |
| Радиус перицентра | p | *r\_p* |
| трансверсальная скорость, км/с |  |  |
| радиальная скорость, км/с |  |  |
| модуль скорости, км/с |  |  |

M(t) – зависимость средней аномалии (M) от времени прохождения (t) полного оборота от апоцентра;

E(t) – зависимость эксцентрической аномалии (E) от времени прохождения (t) полного оборота от апоцентра;

- зависимость истинной аномалии () от времени прохождения (t) полного оборота от апоцентра;

V(t) – вектор скорости спутника;

Vr(t)– радиальная компонента вектора скорости спутника (направлена вдоль радиус-вектора r);

Vn(t)–поперечная (трансверсальную) компонента вектора скорости спутника (направлена перпендикулярно радиус-вектору r).

**Изображение выглядит как текст, Шрифт, линия, снимок экрана

Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, Шрифт, чек

Автоматически созданное описание Графики зависимостей**

**Изображение выглядит как текст, чек, Шрифт, снимок экрана

Автоматически созданное описание**

**Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание**

Изображение выглядит как текст, снимок экрана, линия, График

Автоматически созданное описание

**Объяснение:**

1. График зависимости Vr(t);
2. График зависимости Vn(t);
3. График зависимости V(t);
4. График зависимости r(t);
5. График зависимости E(t).

**Исходный код:**

#include <iostream>

#include <cmath>

#include <fstream>

#define PI 3.14159265358979323846

using namespace std;

double newton\_method(double M, double e, double epsilon = 1e-6)

{

double E = M;

for (int i = 0; i < 40; i++)

{

double f = E - e \* sin(E) - M; // Значение функции

double f\_prime = 1 + e \* cos(E); // Значение производной функции

E -= f / f\_prime; // Вычисление следующего приближения для E

if (abs(f) < epsilon) // Проверка условия сходимости

{

break;

}

}

return abs(E);

}

int main() {

// Параметры орбиты

double r\_a = 1017 + 1737; // км радиус апоцентра (h+r)

double r\_p = 350 + 1737; // км радиус перицентра (h+r)

double epsilon = 1e-6;

double GM\_m = 4902.80003; //в км

double a = (r\_a + r\_p) / 2; // большая полуось

double n = sqrt(GM\_m / pow(a, 3));// средняя угловая скорость

double T = 2 \* PI / n;

double e = ((r\_a - r\_p) / (2 \* a)); // эксцентриситет орбиты 0,48793

// Создание файла

ofstream fout1;

fout1.open("graf.txt");

// Проверяем открытия файла

if (!fout1) {

cout << "Error opening file.";

return 0;

}

fout1 << "t, c\t";

fout1 << "M(t), рад\t";

fout1 << "E(t), рад\t";

fout1 << "Theta(t), рад\t" << endl;

fout1 << "r(t), км\t" << endl;

fout1 << "V\_r(t), м/с\t" << endl;

fout1 << "V\_n(t), м/с\t" << endl;

fout1 << "V(t), м/с\t" << endl; 1)

// Вычисление и запись значений аномалий в файл

Вычисления , Vr, Vn, V для каждого момента t:

for (int t = 0; t <= T; t++) {

fout1 << t << "\t";

double M = n \* t; // Средняя аномалия

double E = newton\_method(M, e, epsilon); // Эксцентрическая аномалия

double true\_anomaly = atan(sqrt((1 + e) / (1 - e)) \* (tan(E / 2))) \* 2; // Истинная аномалия

if (true\_anomaly < 0)

true\_anomaly += 2 \* PI;

double p = a \* (1 - e \* e); // фокальное расстояние

double r = (p / (1 + (e \* cos(true\_anomaly)))); // вычисление радиус-вектора

double V\_n = (sqrt(GM\_m / p)) \* (1 + e \* cos(true\_anomaly)); // трансверсальная скорость

double V\_r = (sqrt(GM\_m / p)) \* (e \* sin(true\_anomaly)); //радиальная сокрость

double V = pow(((V\_r \* V\_r) + (V\_n \* V\_n)) , 0.5); // скорость тела

fout1 << M << "\t";

fout1 << E << "\t";

fout1 << true\_anomaly << "\t";

fout1 << r << "\t";

fout1 << V\_r << "\t";

fout1 << V\_n << "\t";

fout1 << V << endl;

}

fout1.close();

cout << "The data was successfully written to the 'graf.txt' file.";

return 0;

}

Ссылка на гитхаб: https://github.com/Ekhest/MOSF