Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики
УЧЕБНЫЙ ЦЕНТР ОБЩЕЙ ФИЗИКИ ФТФ

УНИВЕРСИТЕТ ИТМО

Группа М3111 К работе допущен

Студент Эседулаева З.А. Работа выполнена

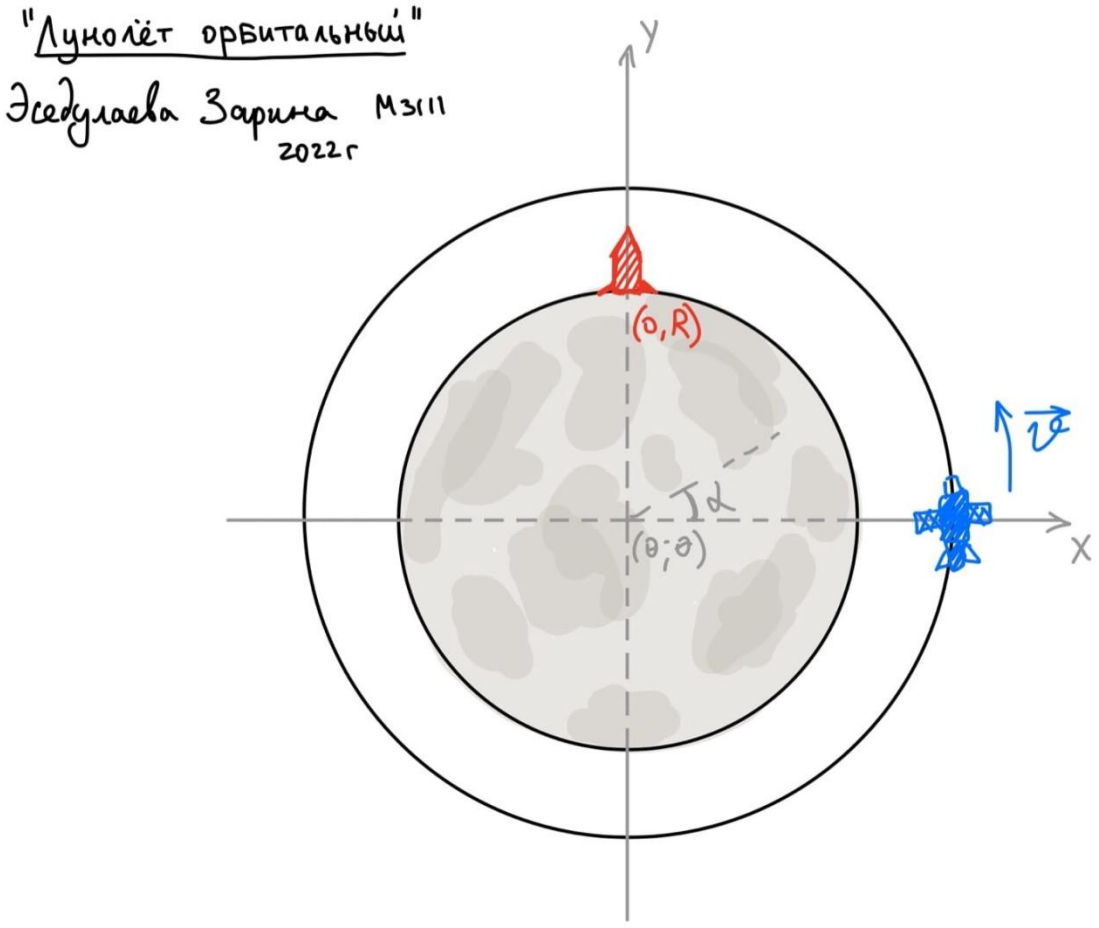
Преподаватель Прохорова У.В. Отчет принят

Отчёт по численному моделированию по физике

№1. Лунолёт орбитальный

Постановка задачи.

Программа вычисляет координаты лунолёта и станции после манёвра, расстояние между ними, а конкретно для лунолёта - угловые скорости, текущий угол наклона относительно OX, статус доставки груза и оставшийся объем газа. Посмотреть на текущие значения можно введя соответствующую команду “2” в меню (Show current situation). По сути, программа лишь помогает пилоту запрограммировать курс полёта, но сама не рассчитывает оптимальный путь. Предусмотрены и особые случаи, например, при перегрузке лунолёт будет лететь свободно, а при манёвре, для которого недостаточно газа, будет выведена соответствующая ошибка, и лунолёт будет лететь свободно.



Реализация.

Программа написана на языке программирования C++ и состоит из 3х файлов: modeling1.cpp, lunolet.h и lunolet.cpp. В основном файле реализовано меню взаимодействия пользователя с программой. В заголовочном файле объявлены классы лунолёта и орбитальной станции *Lunolet* и *Station* соответственно. В lunolet.cpp реализованы методы классов.

Подготовка к реализации.

Для решения задачи были использованы следующие значения и формулы:

1. **Константы.**
   a. Радиус Луны: moonRad = 1738 км.
   b. Высота орбиты: maxHeight = 47 км.
   c. Первая космическая скорость на высоте moonRad + maxHeight = 460.5 км/ч (по формуле $v = \sqrt{\frac{GM}{R}}$)
   d. Ускорение силы тяжести на Луне: g = 1.62 $\text{м}/\text{с}^2$ = 0,00162 $\text{км}/\text{с}^2$
   e. Масса лунолёта: mass = 2000 кг.
   f. Масса груза: cargoMas = 200 кг.
   g. Масса топлива: gasMas = 4000 кг.
   h. Скорость истечения продуктов сгорания из реактивного двигателя: Vp = 3660 м/с
   i. Максимальная перегрузка: a_max = 29.43
2. **Формулы для вычисления значений лунолёта после совершения манёвра.**
   a. Уравнение Мещерского: $m\frac{dV}{dt} = F + Vp\frac{dM}{dt}$, где:

      пренебрегаем F – сопротивлением среды; Vp – скорость истечения продуктов сгорания из реактивного двигателя; m – расход топлива; $\frac{dV}{dt}$ – ускорение, dt – скорость манёвра, dM – масса топлива.
   b. Формула вычисления нового ускорения: $a = \frac{Vp \cdot gasConsumtion}{gasMas \cdot maneuverTime}$ - выводится из уравнения Мещерского.
   c. Формулы вычисления скоростей:

$$lastV = \frac{Vp \cdot gasConsumption}{gasMas} - \text{ выводится из уравнения Мещерского}$$

$$V_x = lastV \cdot \cos\alpha$$

$$V_y = lastV \cdot \sin\alpha$$

      В вычислениях в программе для sin, cos, tan значения угла домножаются на $\frac{\pi}{180^\circ}$, чтобы перевести градусы в радианы.

      В свободном полёте:

$$V_x = V_x$$

$$V_y = V_y - g \cdot t$$

   d. Формулы изменения координат:

$$x = x_0 + V_0 t + \frac{at^2}{2}$$

$$y = y_0 + V_0 t + \frac{at^2}{2}$$

В свободном полёте:

$$x = x_0 + V_x t$$
$$y = y_0 - V_y t$$

e. Формула изменения угла:

$$\alpha = \tan\frac{V_y}{V_x}$$

f. Расход топлива:

$$gasMas = gasMas - gasConsumption \cdot t$$

3. Формулы для вычисления значений станции после совершения манёвра

a. Формула изменения угла:

$$\alpha = \alpha + \frac{\pi}{t}, \text{где } \pi \approx \omega$$

g. Формулы изменения координат:

$$x = x_0 + (moonRad + maxHeight)\cos(\alpha \cdot t)$$
$$y = y_0 + (moonRad + maxHeight)\sin(\alpha \cdot t)$$
$$\text{где } x_0, y_0 - \text{ координаты центра}, = 0$$

Детали реализации.

Файл lunolet.h

В файле объявляются классы Lunolet и Station.

Подключенные библиотеки: <iostream> для работы с консолью.

Класс Lunolet:

Все методы и переменные с модификатором доступа public.

В классе объявлены переменные:

1. Текущие координаты: double x, y с начальными значениями 0 и 1738 соответственно – лунолёт находится на поверхности Луны.
2. Массы: double mass = 2000, cargoMas = 200, gasMas = 4000 – масса лунолёта, груза и топлива соответственно.
3. Константы, описанные разделом выше.
4. Начальные значения скоростей и ускорения: double Vx = 0, Vy = 0, lastV = 0, accel.
5. Переменные с вводимыми пользователем данными: double angle, gasConsumption, maneuverTime.
6. Статус доставки: bool delivered = false.

   Методы:

7. Обновления значений: void upate_speed(); void update_position(); void update_rotation(); и void update(), объединяющий предыдущие три.
8. Свободное падение: void free_fall(); задействуется в аварийных ситуациях.
9. Метод вычисления скоростей и топлива: void jet_effect().
10. Вывести статус доставки груза: void delivery_status();
11. Вывести текущие значения: void show();

Класс Station:

Все методы и переменные с модификатором доступа public.

В классе объявлены переменные:

1. Первая космическая скорость: double const firstCosmicSpeed = 460.5;
2. Текущие координаты: double x, y с начальными значениями 1785 и 0 соответственно – лунолёт находится на орбите и движется по направлению к лунолёту.
3. Переменные с вводимыми пользователем данными: double angle, maneuverTime.

Методы:

4. Обновление координат и угла: void update();
5. Вывести текущие значения: void show();

Файл lunolet.cpp

В файле реализуются методы Lunolet и Station.

Подключенные библиотеки: <cmath> для вычислений.

Класс Lunolet:

1. void update_speed() – вычисляет новое значение ускорения и вызывает метод jet_effect(), а при перегрузке – free_fall().
2. void update_position() – вычисляет новые координаты по приведенным параграфом выше формулам.
3. void update_rotation() – вычисляет новое значения угла
4. void free_fall() – вызывается при свободном падении, выводит состояние “Falling free.” и вычисляет новые значения координат, ускорения, угловых скоростей и вызывает метод update_rotation() для вычисления угла.
5. void show() – выводит условную таблицу с текущими координатами, угловыми скоростями, углом, статусом доставки и массой газа.

```
table: | coords | Vx | Vy | angle | delivered | gas |
       |  (x,y) | -- | -- |   --  |  yes/no   | --- |
```

6. void update() – объединяет три метода update’ов скорости, положения и угла.
7. void delivery_status() – выводит текущее значение доставки груза в виде строки.

Класс Lunolet:

1. void update() – вычисляет новое значение угла и координат.
2. void show() - выводит условную таблицу с текущими координатами и дистанцией между станцией и лунолётом.

```
table: | coords | distance |
       |  (x,y) |    --    |
```

Файл modeling1.cpp

В файле реализуется меню и логика программы.

Подключенные библиотеки: <cmath> для вычислений, <iostream> для работы с консолью и заголовочный файл “lunolet.h”.

Меню реализовано с помощью конструкции switch – case и бесконечного цикла while(true), который прекратится, только если ввести команду, отличную от 1 или 2. При выборе команды 2 “Show current situation” выведутся методы show() у экземпляров класса Lunolet lunolet и Station station. Пример:

```
--- Choose the command ---
1. New maneuver
2. Show current situation
3. Exit
Your command:2
 |   coords   | Vx | Vy | angle | delivered | gas |
| (0,1738) |  0  |  0  |   0  |     NO    | 4000 |

|   coords   |  distance  |
| (1785,0) |     2491.36   |
```

При выборе команды 1 “New maneuver” пользователь должен будет ввести значения угла, расхода топлива и времени манёвра, после чего будут произведены все необходимые вычисления. Чтобы посмотреть изменения, необходимо будет вызвать команду 2. Пример:

```
--- Choose the command ---
1. New maneuver
2. Show current situation
3. Exit
Your command:1
 Angle:45
 Gas consumption:10
 Maneuver time:10
Your command:2
 |   coords   | Vx | Vy | angle | delivered | gas |
| (110.45,1848.45) |  6.47  |  6.47  |   45  |     NO    | 3900 |

|   coords   |  distance  |
| (1782.32,97.82) |     2420.71   |
```