МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ

ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ им. А. И. ГЕРЦЕНА»



**Курсовая работа**

«Вычислительный эксперимент по изучения движения тела по вертикальной плоскости с учётом трения»

Обучающегося 2 курса

очной формы обучения

направление подготовки:

09.03.01 Информатика и вычислительная техника

направленность (профиль):

Технологии разработки программного обеспечения

Волжанин Александр Павлович

Руководитель курсовой работы:

Доктор педагогических наук, профессор

Власова Елена Зотиковна

Санкт-Петербург

2024

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ……………………………………………………………...3

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ…………...…………………………………………………....4

ЗАДАНИЕ 1……………………………………………………………..5  
ЗАДАНИЕ 2……………………………………………………………..7  
ЗАДАНИЕ 3……………………………………………………………..9

ЛИТЕРАТУРА…………………………………………………………..11

**ВВЕДЕНИЕ**

Моделирование прыжка с парашютом играет важную роль в изучении и оптимизации этого процесса. Оно позволяет анализировать различные аспекты прыжка, такие как траектория падения, скорость, влияние атмосферных условий и действие парашюта на движение. Такие исследования помогают улучшить безопасность прыжков, разработать более эффективные обучающие программы и создать новые технологии и оборудование для парашютного спорта.

Анализ моделирования прыжка с парашютом имеет важное значение не только для понимания физических аспектов этого процесса, но и для развития парашютного спорта в целом. Полученные результаты могут быть использованы для улучшения тренировочных программ, повышения безопасности прыжков и создания новых технологий, способствующих развитию данной области спорта.

Постановка задачи: человек совершает прыжок с парашюта с некоторой высоты; требуется построить математическую модель, описывающую процесс спуска парашютиста; с помощью модели исследовать зависимости высоты и скорости парашютиста от времен, а также ответить на ряд практических вопросов.

Цель: провести вычислительный эксперимент и написать пакет прикладных программ для решения следующих задач:

Задача 1: построить зависимости координаты и скорости парашютиста от времени при прыжке с высоты 1 км при разных временах открытия парашюта (3 с, **5 с**, 10 с).

Задача 2: определить максимальное время свободного падения (время до раскрытия парашюта) при прыжке с высоты 3 км, при котором обеспечивается безопасное приземление (скорость приземления <10 м/с).

Задача 3: определить минимальную высоту, с которой можно совершить прыжок, если парашют раскрывается через 3 сек после прыжка. Безопасной скоростью в момент соприкосновения с поверхностью земли будем считать менее 10

Предмет: программы написаны на языке программирования Python и с использованием принципов Объектно ориентированной парадигмы для проведения вычислительного эксперимента. Для реализации были использованы библиотеки. [1, 2]

Результаты вычислительного эксперимента представлены в числовом и графическом вариантах.

Курсовая работа написана на 13 страницах. Содержит 4 рисунка, 3 источника.

**Математическая (теоретическая) модель:**

Для проведения вычислительного эксперимента разработана математическая модель решаемой задачи.  
В её основе 2-ой закон Ньютона.

Для наглядного преставления действующих на систему сил использован рисунок рисунок 1.

На рассматриваемую силу действует сила тяжести и сила сопротивления воздуха.



Рис. 1. Движение парашютиста по вертикальной

, где ) – формула 1.

*– плотность воздуха (t=20˚, p = 105Па)*

*S – площадь поперечного сечения*

*– коэффициент обтекаемости (*

, где – время, в которое раскрыли парашют

**Задание 1:**

Построить зависимости координаты и скорости парашютиста от времени при прыжке с высоты 1 км при разных временах открытия парашюта (3 с, **5 с**, 10 с).

***Текст программы:***

import matplotlib.pyplot as plt

import numpy as np

class RocketModel:

def \_\_init\_\_(self):

# Исходные параметры модели

self.g = 9.8

self.m0 = 70

self.mp = 15

self.m = self.m0 + self.mp

self.r = 0.3

self.S0 = np.pi \* self.r \*\* 2

self.R = 5

self.S1 = np.pi \* self.R \*\* 2

self.ro = 1.3

self.Cx0 = 0.5

self.Cx1 = 1.28

self.k0 = self.Cx0 \* self.ro \* self.S0 / 2

self.k1 = self.Cx1 \* self.ro \* self.S1 / 2

self.H = 1000

self.tr = 5

self.v0 = 0

# Расчёт математической (теоретической) модели

self.T = 500

self.N = 20000

self.dt = self.T / self.N

self.y = np.zeros(self.N)

self.v = np.zeros(self.N)

self.t = np.zeros(self.N)

self.y[0] = self.H

self.v[0] = self.v0

self.t[0] = 0

def calculate\_trajectory(self):

k = self.k0

for i in range(self.N - 1):

if self.t[i] >= self.tr:

k = self.k1

self.y[i + 1] = self.y[i] + self.v[i] \* self.dt

self.v[i + 1] = self.v[i] - (self.g + k / self.m \* abs(self.v[i]) \* self.v[i]) \* self.dt

if self.y[i + 1] < 0:

break

self.t[i + 1] = self.t[i] + self.dt

def plot\_results(self):

i = np.argmax(self.y < 0)

plt.figure(figsize=(10, 8))

plt.subplot(3, 1, 1)

plt.title("y = y(t)")

plt.xlabel('t, c', fontsize=14)

plt.ylabel('y, м', fontsize=14)

plt.plot(self.t[:i], self.y[:i])

plt.grid()

plt.ylim(0)

plt.subplot(3, 1, 2)

plt.title("v = v(t)")

plt.xlabel('t, c', fontsize=14)

plt.ylabel('v, м/с', fontsize=14)

plt.plot(self.t[:i], self.v[:i])

plt.grid()

plt.subplot(3, 1, 3)

plt.title("v = v(y)")

plt.xlabel('y, м', fontsize=14)

plt.ylabel('v, м/с', fontsize=14)

plt.plot(self.y[:i], self.v[:i])

plt.grid()

plt.tight\_layout()

plt.show()

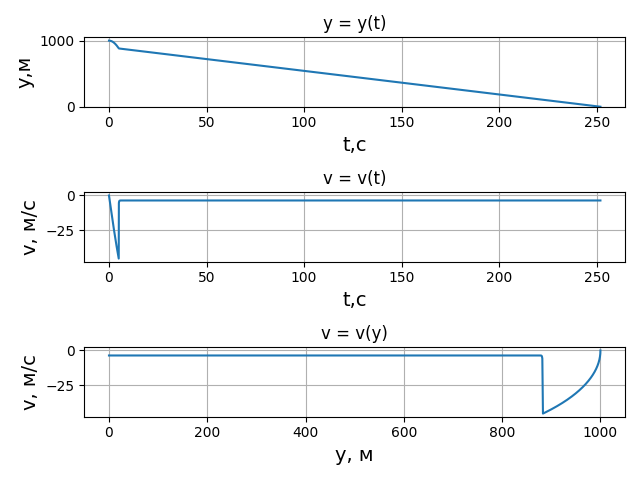
if \_\_name\_\_ == "\_\_main\_\_":

rocket = RocketModel()

rocket.calculate\_trajectory()

rocket.plot\_results()

***Результат работы программы:***



***Анализ полученных результатов:***

Исходя из данных построенного графика можно сделать вывод, что до раскрытия парашюта человек будет падать с ускорением, практически сразу после раскрытия парашюта скорость падения станет постоянной, что обеспечит человеку безопасный для его жизни спуск с большой высоты.

Задание 2:

Определить максимальное время свободного падения (время до раскрытия парашюта) при прыжке с высоты 3 км, при котором обеспечивается безопасное приземление (скорость приземления <10 м/с).

Текст программы:

import matplotlib.pyplot as plt

import numpy as np

class RocketSimulation:

def \_\_init\_\_(self):

# Исходные параметры модели

self.g = 9.8

self.m0 = 70

self.mp = 15

self.m = self.m0 + self.mp

self.r = 0.3

self.S0 = np.pi \* self.r \*\* 2

self.R = 5

self.S1 = np.pi \* self.R \*\* 2

self.ro = 1.3

self.Cx0 = 0.5

self.Cx1 = 1.28

self.k0 = self.Cx0 \* self.ro \* self.S0 / 2

self.k1 = self.Cx1 \* self.ro \* self.S1 / 2

self.H = 3000

self.tr = 38

self.v0 = 0

# Расчёт математической (теоретической) модели

self.T = 500

self.N = 20000

self.dt = self.T / self.N

self.y = np.zeros(self.N)

self.v = np.zeros(self.N)

self.t = np.zeros(self.N)

self.y[0] = self.H

self.v[0] = self.v0

self.t[0] = 0

def calculate\_trajectory(self):

k = self.k0

for i in range(self.N - 1):

if self.t[i] >= self.tr:

k = self.k1

self.y[i + 1] = self.y[i] + self.v[i] \* self.dt

self.v[i + 1] = self.v[i] - (self.g + k / self.m \* abs(self.v[i]) \* self.v[i]) \* self.dt

if self.y[i + 1] < 0:

break

self.t[i + 1] = self.t[i] + self.dt

def plot\_velocity(self):

i = np.argmax(self.y < 0)

plt.title("v = v(t)")

plt.xlabel('t, c', fontsize=14)

plt.ylabel('v, м/с', fontsize=14)

plt.plot(self.t[:i], self.v[:i])

plt.grid(linewidth='1')

plt.show()

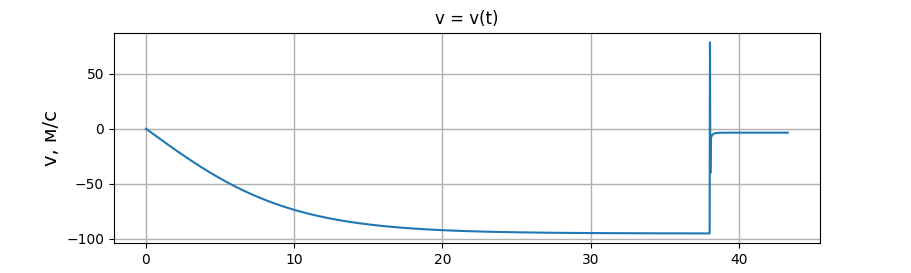
if \_\_name\_\_ == "\_\_main\_\_":

rocket = RocketSimulation()

rocket.calculate\_trajectory()

rocket.plot\_velocity()

Результат работы программы:



Анализ полученных результатов:

Исходя из данных построенного графика можно сделать вывод, что максимально допустимое время полёта до момента «окончательного» открытия парашюта – 38 секунд. Однако, данная модель не учитывает тот фактор, что во время открытия скорость падения изменится более чем на 150 , что окажется смертельным для парашютиста.

Задание 3:

Определить минимальную высоту, с которой можно совершить прыжок, если парашют раскрывается через 3 сек после прыжка. Безопасной скоростью в момент соприкосновения с поверхностью земли будем считать менее 10

Текст программы:

import matplotlib.pyplot as plt

import numpy as np

class RocketSimulation:

def \_\_init\_\_(self):

# Исходные параметры модели

self.g = 9.8

self.m0 = 70

self.mp = 15

self.m = self.m0 + self.mp

self.r = 0.3

self.S0 = np.pi \* self.r \*\* 2

self.R = 5

self.S1 = np.pi \* self.R \*\* 2

self.ro = 1.3

self.Cx0 = 0.5

self.Cx1 = 1.28

self.k0 = self.Cx0 \* self.ro \* self.S0 / 2

self.k1 = self.Cx1 \* self.ro \* self.S1 / 2

self.H = 46

self.tr = 3

self.v0 = 0

# Расчёт математической (теоретической) модели

self.T = 500

self.N = 20000

self.dt = self.T / self.N

self.y = np.zeros(self.N)

self.v = np.zeros(self.N)

self.t = np.zeros(self.N)

self.y[0] = self.H

self.v[0] = self.v0

self.t[0] = 0

def calculate\_trajectory(self):

k = self.k0

for i in range(self.N - 1):

if self.t[i] >= self.tr:

k = self.k1

self.y[i + 1] = self.y[i] + self.v[i] \* self.dt

self.v[i + 1] = self.v[i] - (self.g + k / self.m \* abs(self.v[i]) \* self.v[i]) \* self.dt

if self.y[i + 1] < 0:

break

self.t[i + 1] = self.t[i] + self.dt

def plot\_velocity\_vs\_height(self):

i = np.argmax(self.y < 0)

plt.title("v = v(y)")

plt.xlabel('y, м', fontsize=14)

plt.ylabel('v, м/с', fontsize=14)

plt.plot(self.y[:i], self.v[:i])

plt.grid()

plt.tight\_layout()

plt.show()

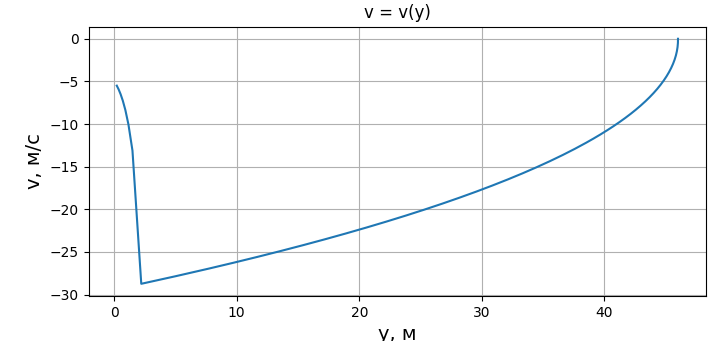
if \_\_name\_\_ == "\_\_main\_\_":

rocket = RocketSimulation()

rocket.calculate\_trajectory()

rocket.plot\_velocity\_vs\_height()

Результат работы программы:

******

Анализ полученных результатов:

Исходя из данных построенного графика можно сделать вывод, что минимальной безопасной высотой прыжка будет являться 46 метров, так как прыжок с такой высоты обеспечит скорость около 5 – 6 у поверхности земли, что является безопасной скоростью для «мягкой» посадки человека.

Литература

1. Лабораторная работа по Python [Электронный ресурс] / Московский физико-технический институт. – Режим доступа: http://cs.mipt.ru/python/lessons/lab1.html, свободный. – Загл. с экрана. – Яз. рус.

2. Курс по NumPy [Электронный ресурс] / МФТИ. – Режим доступа: https://mipt-stats.gitlab.io/courses/python/05\_numpy.html, свободный. – Загл. с экрана. – Яз. рус.

3. Механика: учебное пособие [Электронный ресурс] / Скачать книги по физике. – Режим доступа: https://scask.ru/j\_book\_mech.php?id=70, свободный. – Загл. с экрана. – Яз. рус.