УДК 004

Разработка программы для симуляции солнечной системы на языке программирования Python

Жуков Дмитрий Сергеевич
Приамурский государственный университет им. Шолом-Алейхема
Студент

Аннотация
Целью данной статьи является разработки программы для симуляции солнечной системы. Для достижения этой цели был использован язык программирования Python. В этой статье была разработана программа для симуляции солнечной системы.
Ключевые слова: Python, солнечная система, программа, симуляция, разработка.

Development of a program for simulating the solar system in the Python programming language

Zhukov Dmitry Sergeevich
Sholom-Aleichem Priamursky State University
Student

Abstract
The purpose of this article is to develop a program for simulating the solar system. To achieve this goal, the Python programming language was used. In this article, a program for simulating the solar system was developed.
Keywords: Python, solar system, program, simulation, development

Данная программа поможет наглядно увидеть расположение планет в нужный период времени.

Цель исследования: разработка программы для симуляции солнечной системы на языке программирования Python.

И.Н. Пантелеймонов, А.Ю. Потюпкин, В.И. Ковалев, А.В. Баринов рассмотрели в своей статье проблему разработки основных направлений создания системы связи с космическими объектами, находящимися на орбите или поверхности планет Солнечной системы (или их спутников) с целью изучения и освоения. Для решения задачи повышения эффективности системы дальней космической связи предлагается применение группировки планетарных спутников-ретрансляторов, осуществляющих ретрансляцию информации от таких космических объектов на геостационарные спутники-ретрансляторы в оптическом диапазоне [2]. Р.Р Мухаметзянов в своей статье рассматривает базовые возможности использования языка программирования Python для изучения курса алгоритмизации и программирования.

Рассмотрены основные алгоритмические конструкции языка и особенности их реализации. В работе продемонстрированы объектно-ориентированные возможности языка. Приведены примеры реализации различных программ и классов в этом языке [1]. Язев С.А. В своей статье описал состав и структуру солнечной системы. [3]. Так же не мало важны иностранные источники [4].

Сначала импортируем библиотеки и устанавливаем дату начала симуляции, продолжительность в днях, массу земли (Рис.1)

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.animation as animation
from astropy.time import Time
from astroquery.jplhorizons import Horizons

sim_start_date = "2020-09-08"
sim_duration = 2 * 365
m_earth = 5.9722e24 / 1.98847e30
m_moon = 7.3477e22 / 1.98847e30
```

Рис.1.Импорт и параметры

Далее создаем класс объектов солнечной системы – планет (Рис.2)

```
class Object:
    def __init__(self, name, rad, color, r, v):
        self.name = name
        self.r = np.array(r, dtype=np.float)
        self.v = np.array(v, dtype=np.float)
        self.xs = []
        self.ys = []
        self.plot = ax.scatter(r[0], r[1], color=color, s=rad**2, edgecolors=None, zorder=10)
        self.line, = ax.plot([], [], color=color, linewidth=1.4)
```

Рис.2. Класс планет

Далее необходимо создать класс солнечной системы, в котором будет храниться информация о положении планет, а также вычисляться траектория движения (Рис.3).

```
class SolarSystem:
    def __init__(self, thesun):
        self.thesun = thesun
        self.planets = []
        self.time = None
        self.timestamp = ax.text(.03, .94, 'Дата: ', color='w', transform=ax.transAxes, fontsize='x
    def add_planet(self, planet):
        self.planets.append(planet)
    def evolve(self):
        dt = 1.0
        self.time += dt
        plots = []
        lines = []
        for p in self.planets:
            p.r += p.v * dt
            acc = -2.959e-4 * p.r / np.sum(p.r**2)**(3./2)  # в единицах AU/day^2
            p.v += acc * dt
            p.xs.append(p.r[0])
            p.ys.append(p.r[1])
            p.plot.set_offsets(p.r[:2])
            p.line.set_xdata(p.xs)
            p.line.set_ydata(p.ys)
            plots.append(p.plot)
            lines.append(p.line)
        self.timestamp.set_text('Дата: ' + Time(self.time, format='jd', out_subfmt='date').iso)
        return plots + lines + [self.timestamp]
```

Рис.3. Класс солнечной системы

Далее настроим отображение системы с помощью библиотеки matplotlib (Рис.4)

```
plt.style.use('dark_background')
fig = plt.figure(figsize=[6, 6])
ax = plt.axes([0., 0., 1., 1.], xlim=(-1.8, 1.8), ylim=(-1.8, 1.8))
ax.set_aspect('equal')
ax.axis('off')
```

Рис.4. Код отображения солнечной системы

Инициализируем солнечную систему и планеты в ней (Рис.5)

```
ss = SolarSystem(Object("Sun", 28, 'orange', [0, 0, 0], [0, 0, 0]))
ss.time = Time(sim_start_date).jd
colors = ['gray', 'yellow', 'blue', 'chocolate']
sizes = [0.38, 0.95, 1., 0.53]
names = ['Меркурий', 'Венера', 'Земля', 'Марс']
texty = [.47, .73, 1, 1.5]
for i, nasaid in enumerate([1, 2, 3, 4]):
    obj = Horizons(id=nasaid, location="@sun", epochs=ss.time, id_type='id').vectors()
    ss.add_planet(Object(nasaid, 20 * sizes[i], colors[i],
                         [np.double(obj[xi]) for xi in ['x', 'y', 'z']],
                         [np.double(obj[vxi]) for vxi in ['vx', 'vy', 'vz']]))
    ax.text(0, - (texty[i] + 0.1), names[i], color=colors[i], zorder=1000, ha='center', fontsize='large')
def animate(i):
    return ss.evolve()
ani = animation.FuncAnimation(fig, animate, repeat=False, frames=sim_duration, blit=True, interval=20,)
plt.show()
```

Рис.5. Инициализация солнечной системы

Настраиваем анимацию (Рис.6)

```
def animate(i):
    return ss.evolve()
ani = animation.FuncAnimation(fig, animate, repeat=False, frames=sim_duration, blit=True, interval=20,)
plt.show()
```

Рис.6. Код анимации

Теперь запустим программу и увидим анимацию планет в солнечной системе в соответствии с датой (Рис.7 и Рис.8)

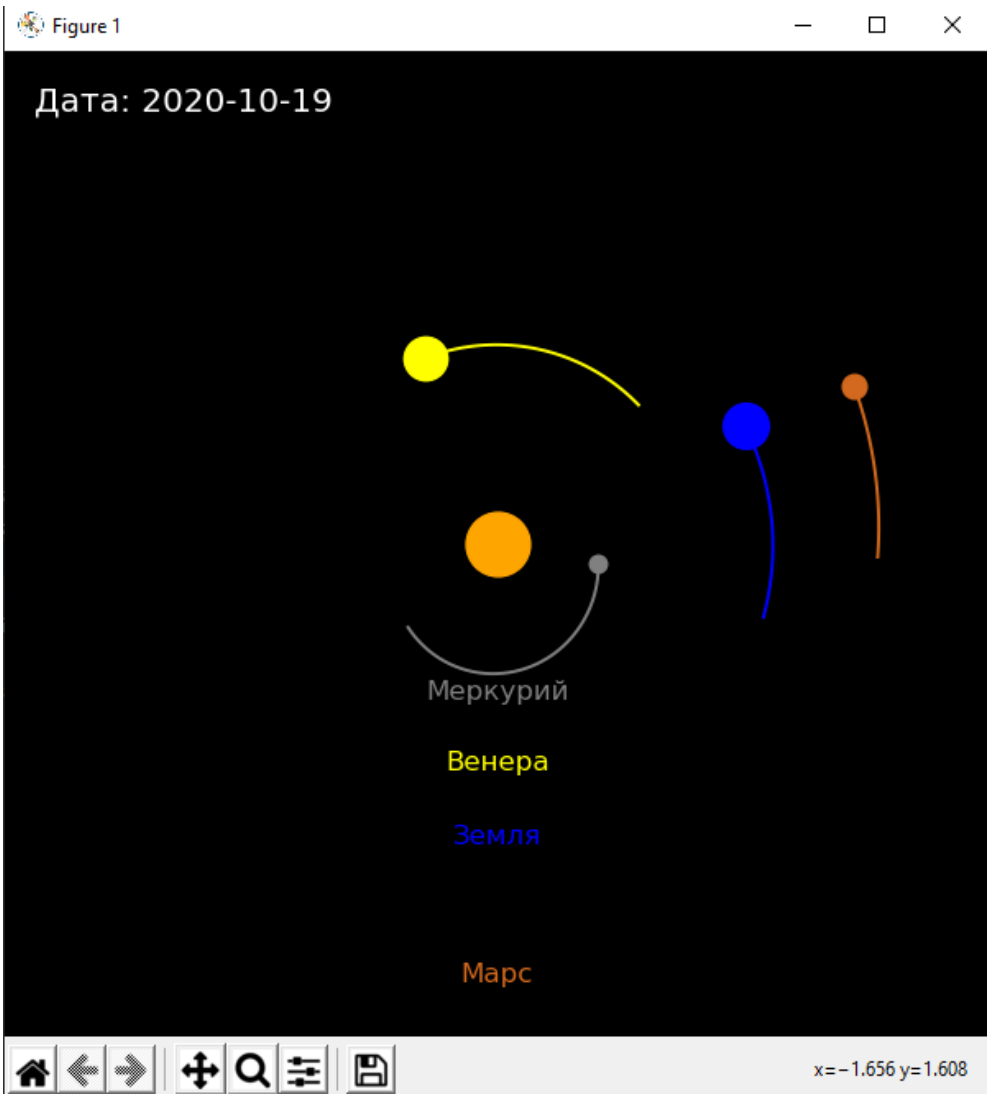

Рис.7. Анимация планеты 1

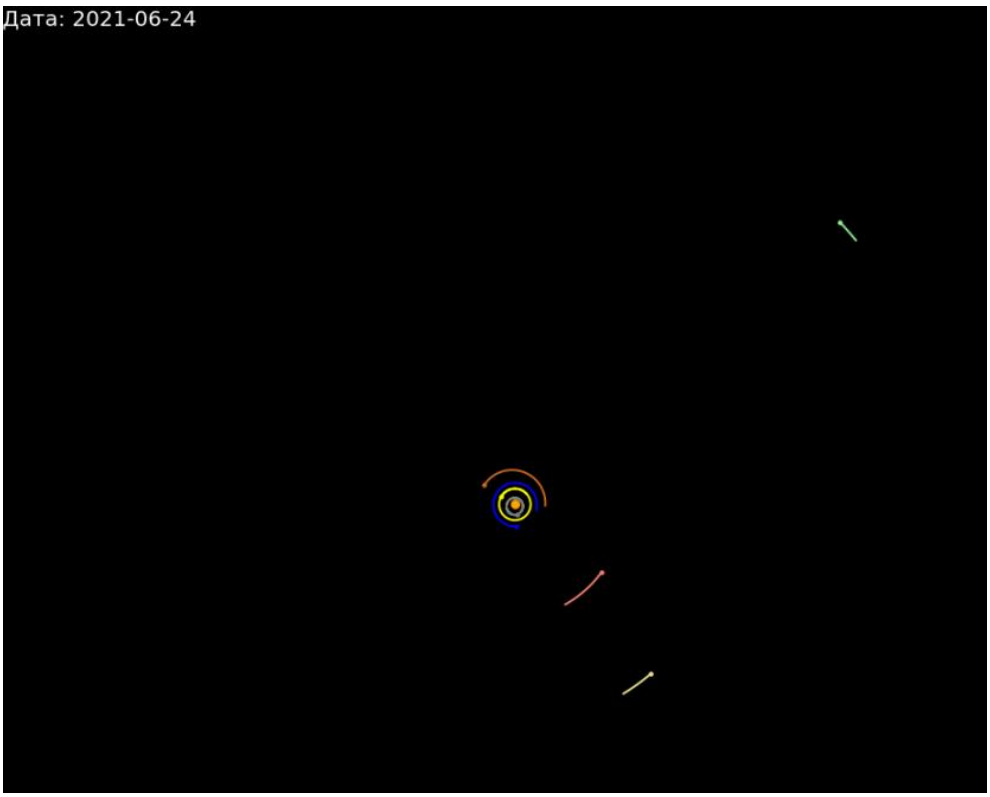

Рис.8. Анимация планет 2

Была произведена 3я симуляция с точкой старта на 3 года раньше (Рис.9)

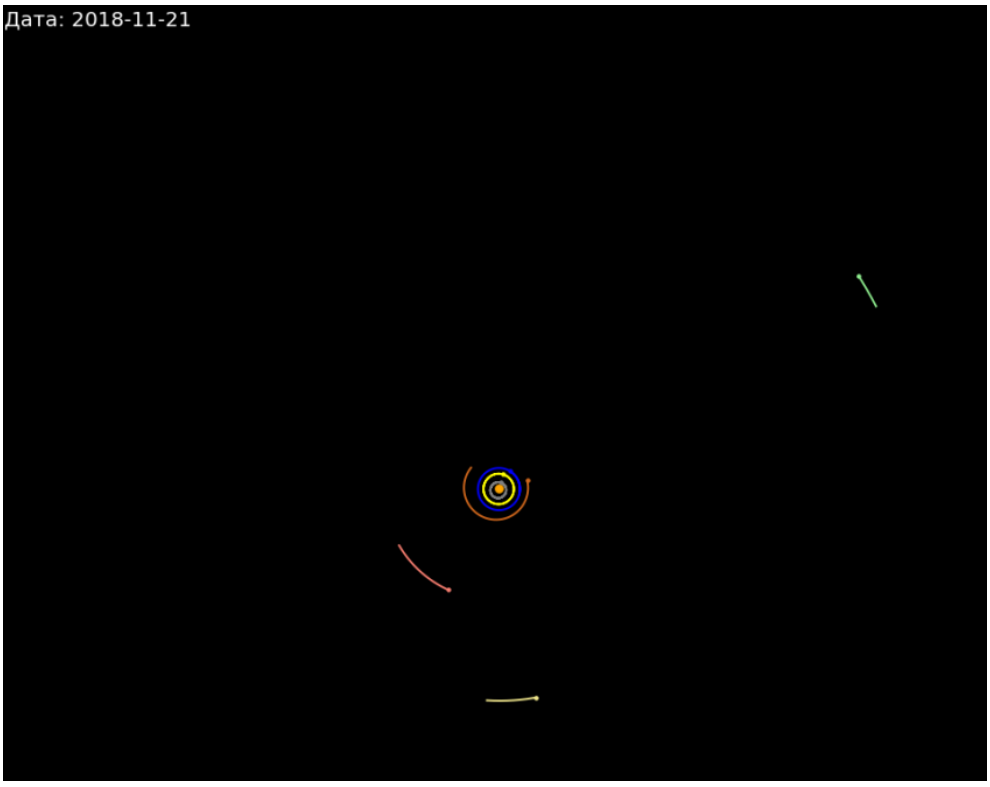

Рис.9. Анимация планет 3

Таким образом, была написана программа, для симуляции солнечной системы на языке Python.

Библиографический список

1. Мухаметзянов Р.Р. Использование языка python для изучения алгоритмизации и программирования // Педагогическая информатика. 2014. № 1. С. 79-87.
2. Пантелеймонов И.Н., Потюпкин А.Ю., Ковалев В.И., Баринов А.В., Филатов В.В. Модель системы связи с объектами, расположенными на орбите и поверхности планет солнечной системы или их спутников // Электросвязь. 2020. № 1. С. 22-26.
3. Язев С.А. Введение в астрономию // Лекции о солнечной системе учебное пособие. Иркутск : Иркутский гос. ун-т, 2010.
4. Cabrera J., Csizmadia Sz., Rauer H., Erikson A., Dreyer C., Eigmüller Ph., Lehmann H., Hatzes A., Dvorak R., Gandolfi D. The planetary system to kic 11442793: a compact analogue to the solar // The Astrophysical Journal. 2014. Т. 781. № 1. С. 18.