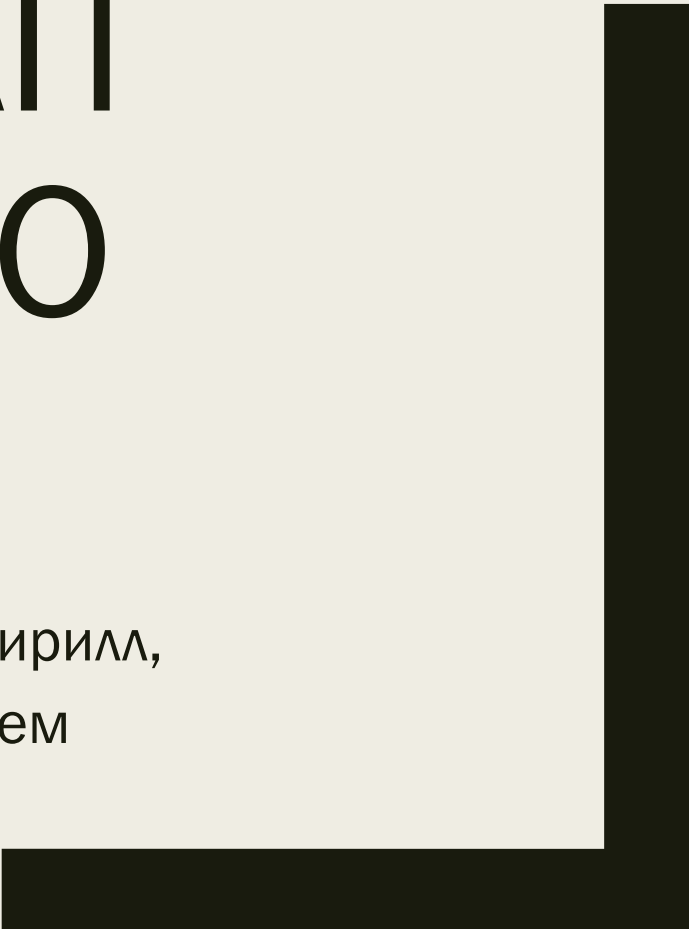




ТРЕТИЙ ЭТАП ГРУППОВОГО ПРОЕКТА

Выполнили: Абакумов Егор, Сухарев Кирилл,
Калинина Кристина, Еременко Артем



Цель этапа

Написать программу, реализующую модель образования планетарной системы.

Описание программы

Константы:

- Количество частиц(N);
- Гравитационная постоянная(G). В целях ускорения процесса может отличаться от существующей;
- Коэффициент отталкивания(k);
- Временной промежуток между итерациями (dt).
- Коэффициент радиуса для слипания (α)
- Коэффициент потенциальной энергии (γ)

Описание программы

Каждая частица имеет следующие характеристики:

- Координаты (x, y) , определяющие положение частицы в пространстве;
- скорость (v_x, v_y) ;
- ускорение (a_x, a_y) ;
- радиус (r) , определяющий размеры частицы;
- масса (m) ;
- коэффициент трения (μ) .

Описание программы

Ход программы:

1. Для каждой частицы:

- создаём переменные для нового ускорения (a_{n+1})
- в них помещаем ускорение, образованное векторной суммой следующих сил:
 - $a_1 = G \frac{m}{r^2}$, направлено вдоль прямой, соединяющей 2 частицы;
 - $a_2 = \frac{k}{m} \left(\left(\frac{a}{b} \right)^8 - 1 \right)$, направлено против прямой, соединяющей 2 частицы;
 - $a_3 = \mu_1 \mu_2 a_2$, направлено перпендикулярно прямой, соединяющей 2 частицы;
 - данные ускорения считаются для каждой пары частиц.

Описание программы

2. Рассчитываем новые координаты:

$$x_{n+1} = x_n + v_n dt + \frac{a_n dt^2}{2}$$

3. Рассчитываем новую скорость

$$v_{n+1} = v_n + \frac{a_{n+1} + a_n}{2} dt$$

4. При сближении частиц на близкое расстояние производим слипание частиц

$$x = \frac{m_i x_i + m_j x_j}{m_i + m_j}$$

$$v = \frac{m_i v_i + m_j v_j}{m_i + m_j}$$

$$a = \frac{m_i a_i + m_j a_j}{m_i + m_j}$$

$$R = \sqrt[3]{R_i^3 + R_j^3}$$

Реализация

Описание констант

```
9     SCREEN_WIDTH = 800
10    SCREEN_HEIGHT = 800
11
12    BACKGROUND_COLOR = (15, 13, 62)
13    PARTICLE_COLOR = (216, 213, 244)
14
15    IS_BORDERS_EXISTS = True
16
17    sc = pg.display.set_mode((SCREEN_WIDTH, SCREEN_HEIGHT))
18
19    N = 75
20    G = 6.6743015 * 10 ** (-11)
21    k = 5 * 10 ** 15
22    alpha = 0.6
23    gamma = 0.01
24    dt = 10 ** (-3)
```

Реализация

Начальные значения

```
26     x = nrand.uniform(1, SCREEN_WIDTH, N)
27     y = nrand.uniform(1, SCREEN_HEIGHT, N)
28     ax = np.zeros(N)
29     ay = np.zeros(N)
30     vx = nrand.sample(N)
31     vy = nrand.sample(N)
32     r = nrand.randint(5, 10, N)
33     m = np.array([10 ** (14) * i ** 3 for i in r])
34     mu = nrand.sample(N)
35     d = np.zeros((N, N))
36     dx = np.zeros((N, N))
37     dy = np.zeros((N, N))
38
39     t = np.array([])
40     E = np.array([])
41     c = 0
```


Реализация

Ход программы

```
58 for i in range(N):
59     d[i] = np.sqrt((x - x[i]) ** 2 + (y - y[i]) ** 2)
60     dx[i] = x - x[i]
61     dy[i] = y - y[i]
62     dx = np.divide(dx, d, where = d != 0)
63     dy = np.divide(dy, d, where = d != 0)
64
65     nax = np.zeros(N)
66     nay = np.zeros(N)
67
68     for i in range(N):
69         for j in range(N):
70             if (d[i][j] != 0):
71                 rs = r[i] + r[j]
72
73                 # Gravity force
74                 if (d[i][j] > rs):
75                     gravity_value = G * m[i] * m[j] / d[i][j] ** 2
76                     nax[i] += dx[i][j] * gravity_value
77                     nay[i] += dy[i][j] * gravity_value
78                 else:
79                     # Repulsive force
80                     repulsive_value = k * ((rs / d[i][j]) ** 8 - 1)
81                     nax[i] += dx[j][i] * repulsive_value
82                     nay[i] += dy[j][i] * repulsive_value
83                     # Friction force
84                     friction_value = repulsive_value * mu[i] * mu[j]
85                     nax[i] += -1 * dy[i][j] * friction_value
86                     nay[i] += dx[i][j] * friction_value
87
88     nax /= m
89     nay /= m
```

Реализация

Слипание частиц

```
107 while i < N - 1:
108     j = i + 1
109     while j < N:
110         if (i == j):
111             continue
112         rs = r[i] + r[j]
113         if (d[i][j] < alpha * rs):
114             m_c = m[i] + m[j]
115             x[i] = (x[i] * m[i] + x[j] * m[j]) / m_c
116             y[i] = (y[i] * m[i] + y[j] * m[j]) / m_c
117             vx[i] = (vx[i] * m[i] + vx[j] * m[j]) / m_c
118             vy[i] = (vy[i] * m[i] + vy[j] * m[j]) / m_c
119             ax[i] = (ax[i] * m[i] + ax[j] * m[j]) / m_c
120             ay[i] = (ay[i] * m[i] + ay[j] * m[j]) / m_c
121             r[i] = (r[i] ** 3 + r[j] ** 3) ** (1 / 3)
122             m[i] = m_c
123             m = np.delete(m, j)
124             x = np.delete(x, j)
125             y = np.delete(y, j)
126             vx = np.delete(vx, j)
127             vy = np.delete(vy, j)
128             ax = np.delete(ax, j)
129             ay = np.delete(ay, j)
130             r = np.delete(r, j)
131             d = np.delete(d, j, 0)
132             d = np.delete(d, j, 1)
133             dx = np.delete(dx, j, 0)
134             dx = np.delete(dx, j, 1)
135             dy = np.delete(dy, j, 0)
136             dy = np.delete(dy, j, 1)
137             j -= 1
138             N -= 1
139         j += 1
140     i += 1
```