

```

1  #!/usr/bin/env python
2
3  import sys
4  from math import *
5  import numpy as np
6  import matplotlib.pyplot as plt
7  from matplotlib import animation
8  #sila centralna=f(r)=-alfa/r^3
9
10
11 # funkcje pomocnicze dpo rozwiazania równań różniczkowych
12 # dr/dt=u -F1
13 #du/dt=L^2/mr^3+f(r)/m -F2
14 #dfi/dt=L/mr^3 -F3
15 #z1=r
16 #z2=dr/dt=u
17 #z3=fi
18
19 def F1(z1,z2,z3,t):
20     w1=z2
21     return w1
22
23 def F2(z1,z2,z3,t):
24     w2=1/(m*m*z1*z1*z1)-alfa*pow(z1,-3)/m
25     return w2
26
27 def F3(z1,z2,z3,t):
28     w3=1/(m*z1*z1)
29     return w3
30 # funkcje do policzenia x i y
31 def toxa(z1,z3):
32     return z1*cos(z3)
33
34 def toya(z1,z3):
35     return z1*sin(z3)
36
37
38 out = open("plik.dat" , 'w')
39 m=1.0 ##masa
40 alfa=3.75# alfa
41 kappa=1.0/4.0 #kappa
42 l=pow(m*alfa/(1-kappa*kappa),0.5) # zowa skladowa momentu pedu
43 rp=1.0# poczatkowe r
44 rdp=0.0 #poczatkowe dr/dt
45 fip=0.0 # poczatkowe fi
46 fidp=1/(m*rp*rp) # poczatkowe dfi/dt
47 ener=m*rdp*rdp/2.0+1*1/(2.0*m*rp*rp)-alfa*pow(rp,-2.0)/2.0 # energia będąca stałą ruchu
48
49
50 dt=0.00005 # krok calkowania
51 T=40#czas symulacji
52
53 # ustalenie warunków początkowych
54 r=rp
55 rd=rdp
56 fi=fip
57 fid=fidp
58
59
60 t=0
61 # algorytm Runge-Kutta 4 rzędu
62 while t < T:
63
64     wr1=F1(r,rd,fi,t)*dt
65     wrd1=F2(r,rd,fi,t)*dt
66     wfi1=F3(r,rd,fi,t)*dt
67     wr2=F1(r+wr1/2.0,rd+wrd1/2.0,fi+wfi1/2.0,t+dt/2.0)*dt
68     wrd2=F2(r+wr1/2.0,rd+wrd1/2.0,fi+wfi1/2.0,t+dt/2.0)*dt
69     wfi2=F3(r+wr1/2.0,rd+wrd1/2.0,fi+wfi1/2.0,t+dt/2.0)*dt
70     wr3=F1(r+wr2/2.0,rd+wrd2/2.0,fi+wfi2/2.0,t+dt/2.0)*dt
71     wrd3=F2(r+wr2/2.0,rd+wrd2/2.0,fi+wfi2/2.0,t+dt/2.0)*dt
72     wfi3=F3(r+wr2/2.0,rd+wrd2/2.0,fi+wfi2/2.0,t+dt/2.0)*dt
73     wr4=F1(r+wr3,rd+wrd3,fi+wfi2,t+dt)*dt
74     wrd4=F2(r+wr3,rd+wrd3,fi+wfi2,t+dt)*dt
75     wfi4=F3(r+wr3,rd+wrd3,fi+wfi2,t+dt)*dt
76
77     r=r+(1/6.0)*(wr1+2.0*wr2+2.0*wr3+wr4)
78     rd=rd+(1/6.0)*(wrd1+2.0*wrd2+2.0*wrd3+wrd4)
79     fi=fi+(1/6.0)*(wfi1+2.0*wfi2+2.0*wfi3+wfi4)
80
81     # zapis wyniku obliczen do pliku
82     s = "%f\t%f\t%f\t%f\t%f\t%f\n" % (t , toxa(r,fi), toya(r,fi), r , rd,fi )
83     out.write(s)
84     t+=dt
85 out.close()
86
87
88 data = np.loadtxt("plik.dat")
89 #odczyt z pliku t,x,y,r,dr/dt,fi
90 t, x1, y1, r1, rd1, fi1 = (data[:,i] for i in range(6))
91 #ustalenie wymiarów rysunku i grubosci toru
92 plt.subplot(211)
93 plt.plot(t,r1)
94 plt.xlabel('t')
95 plt.ylabel('r')
96 plt.subplot(212)
97 plt.plot(t,fi1)
98 plt.xlabel('t')

```

```

99 plt.ylabel('fi')
100 plt.show()
101 plt.subplot(211)
102 plt.plot(t,x1)
103 #plt.xlabel('t')
104 plt.ylabel('x')
105 plt.subplot(212)
106 plt.plot(t,y1)
107 plt.xlabel('t')
108 plt.ylabel('y')
109 plt.show()
110
111 fig = plt.figure(figsize=[6,6])
112 ax = plt.axes(xlim=(-6.0, 6.0), ylim=(-7.0, 5.0))
113 line, = ax.plot([], [], "r-", lw=1, alpha=0.5)
114 punkt, = ax.plot([], [], "ro-", lw=2)
115
116
117 #animacja
118 animate_n = 320 #ustala szybkość animacji
119
120 def animate(i):
121     i=i*animate_n
122     line.set_data(x1[:i],y1[:i])
123     punkt.set_data(x1[i-1:i], y1[i-1:i])
124     return line,punkt
125
126 frame_n=T/dt / animate_n
127 anim = animation.FuncAnimation(fig, animate, frames = int(frame_n), interval=10)
128 plt.show()
129

```