04/13/18 02:08:51 /home/jorrit/git/StarandPlanetform/Orbit_calculations/Kriekscode/initialOrbitals.py

```
 1
 2  import numpy as np
 3  import inspect
 4
 5
 6  # parent class for orbitals
 7  class Orbitals(object):
 8      """Initiation mainclass for all orbitals"""
 9
10      instances = []
11      def __init__(self, name, mass):
12          self.name = name
13          self.mass = mass
14          self.ax = 0
15          self.ay = 0
16          self.rungevalues = []
17          self.xlist = []
18          self.ylist = []
19          Orbitals.instances.append(self)
20
21      def CM(self):
22          # calculate center of mass based on amount of objects
23          # and reset the coordinate system to the center of mass
24          masses = 0
25          xpositions = 0
26          ypositions = 0
27          for i in Orbitals.instances:
28              masses += i.mass
29              xpositions += i.mass * i.x
30              ypositions += i.mass * i.y
31
32          rx = xpositions/masses
33          ry = ypositions/masses
34
35          self.x = self.x - rx
36          self.y = self.y -ry
37
38      def RKCM(self,dt):
39          # calculate center of mass based on amount of objects
40          # and reset the coordinate system to the center of mass
41          masses = 0
42          xpositions = 0
43          ypositions = 0
44          for i in Orbitals.instances:
45              masses += i.mass
46              xpositions += i.mass * (i.x + i.rungevalues[-1][0]*dt)
47              ypositions += i.mass * (i.y + i.rungevalues[-1][1]*dt)
48
49          rx = xpositions/masses
50          ry = ypositions/masses
51
52          return rx,ry
53
54      def InitialSpeed(self):
55
56  # If object is a planet calculate its velocity
57          if isinstance(self,Planet):
58              self.vy = (1/(1+q))*np.sqrt(G*(Ms+self.mass)/self.x)
59
60          if isinstance(self,PlanetHW):
61              self.vy = (1/(1+q))*np.sqrt(G*(Ms+self.mass)/self.x)
62
63
64  #  IF object is a star calculate its velocity
65          if isinstance(self,Star):
66              massvsum = 0
67
68              for i in Orbitals.instances:
69                  if i.name == 'Planet':
70                      massvsum +=i.mass * i.vy
71
72              self.vy = -1/self.mass * massvsum
73
74
75
```

```
76  # inherit Orbital parent qualities and define a planet
77  class Planet(Orbitals):
78      """Initiazion planet subclasses"""
79
80      def __init__(self,name, expl_name, a,q,e,mass, color):
81          self.x = a
82          self.e = e
83          self.vy = 0
84          self.y = 0
85          self.vx = 0
86          self.q = q
87          self.expl_name = expl_name
88          self.color = color
89
90
91          Orbitals.__init__(self,name,mass)
92
93  # inherit Orbital parent qualities and define a planet
94  class PlanetHW(Orbitals):
95      """Initiazion planet subclasses"""
96
97      def __init__(self,name, expl_name, a,q,e,mass, hw):
98          self.x = a
99          self.e = e
100         self.vy = 0
101         self.y = 0
102         self.vx = 0
103         self.q = q
104         self.headwind = hw
105         self.expl_name=expl_name
106
107
108         Orbitals.__init__(self,name,mass)
109
110 # inherit Orbital parent qualities and define a star
111 class Star(Orbitals):
112     """Initization star subclasses"""
113     def __init__(self,name, expl_name, mass, color):
114         self.x = 0
115         self.y = 0
116         self.vy = 0
117         self.vx = 0
118         self.color = color
119         self.expl_name = expl_name
120         Orbitals.__init__(self,name,mass)
121
122
123 # default values
124 Mp = 5.972e24
125 Ms = 1.989e30
126 q = Mp/Ms
127 dt = 3600*24*365.25*0.01
128 stepamount = int((10*365.25*3600*24)/dt)
129 a =1.496e11
130 e =0.0167
131 G = 6.67408e-11
132 tstop = 10.*(365.25*24*3600)
133 directdraw  = False
134 calcEuler = False
135 calcLeap = False
136 calcRK = False
137 gashead = 0.1
138 calcAll = False
139 timer = False
140 hourmonth = False
141 timehdwm = ""
```