1 Music of the Spheres

1.1 Grading guidelines

10 points total

- 2 points for comments regardless of quality
- 3 points for correct radii computations
- 3 points for correct edge computations
- 2 for printing table in proper format (decimals are aligned, two places shown)

1.2 Sample code

```python
# Kepler.py
# Charles Van Loan
# Jan 1, 2015
#""" Computes the radii of the Kepler spheres
# and compares the ratios with reality.
"""

from math import sqrt

# The "true" orbit radii:
rMercuryTrue = 35.98
rVenusTrue = 67.24
rEarthTrue = 92.90
rMarsTrue = 141.60
rJupiterTrue = 483.80
rSaturnTrue = 890.70

# Start the nesting process
rMercury = 1
print '\nPlanet-Pair    Kepler    Actual'
```
print( '------------------------------------')

# The Edge of the smallest O that encloses S1
E0 = rMercury/(sqrt(6)/6)

# The radius of the smallest S2 that encloses O
rVenus = E0*(sqrt(2)/2)

rVM = rVenus/rMercury

rVenusTrue/rVenusTrue

print 'Venus:Mercury %6.2f %6.2f' % (rVM,rVMTrue)

# The Edge of the smallest I that encloses S2
EI = rVenus/((3*sqrt(3)+sqrt(15))/12)

# The radius of the smallest S3 that encloses I
rEarth = (sqrt(10+2*sqrt(5))/4)*EI

rEV = rEarth/rVenus

rEarthTrue/rVenusTrue

print 'Earth:Venus %6.2f %6.2f' % (rEV,rEVTrue)

# The Edge of the smallest D that encloses S3
ED = rEarth/(sqrt(250+110*sqrt(5))/20)

# The radius of the smallest S4 that encloses D
rMars = ((sqrt(15)+sqrt(3))/4)*ED

rME = rMars/rEarth

rMarsTrue/rEarthTrue

print 'Mars:Earth %6.2f %6.2f' % (rME,rMETrue)

# The Edge of the smallest T that encloses S4
ET = rMars/(sqrt(6)/12)

# The radius of the smallest S5 that encloses T
rJupiter = (sqrt(6)/4)*ET

rJM = rJupiter/rMars

rJupiterTrue/rMarsTrue

print 'Jupiter:Mars %6.2f %6.2f' % (rJM,rJMTrue)

# The Edge of the smallest C that encloses S5
EC = 2*rJupiter

# The radius of the smallest S6 that encloses C
rSaturn = (sqrt(3)/2)*EC

rSaturn/rJupiter

print( '------------------------------------')
rSJTrue = rSaturnTrue/rJupiterTrue
print 'Saturn:Jupiter %6.2f %6.2f' % (rSJ,rSJTrue)

# 10 points total
# 2 for any comments regardless of quality
# 3 for the radii computations (there are 5)
# 3 for the edge computations (there are 5)
# 2 for the table (1 for decimals aligned, for two decimal places shown)
2 You Are Late!

2.1 Grading guidelines

10 points total

- 2 points for correct string parsing (1 for hours, 1 for minutes)
- 2 points for correct hour and minute calculation when minutes $\geq 30$ (1 for h, 1 for m)
- 2 points for correct hour and minute calculation when minutes $< 30$ (1 for h, 1 for m)
- 1 point for correct hour calculation when going from PM to AM (i.e. 1PM to 11:30 AM)
- 1 point for correct hour string
- 1 point for correct minute string
- 1 point for properly concatenating hour and minute strings and printing the missed appointment message: "You missed your (earlier time) appointment!"

2.2 Sample code

```python
# YouAreLate.py
# Charles Van Loan
# January 1, 2015

""" Inputs a length-5 time string of the form 'hh:mm'
that encodes an arrival time in hours and minutes and
displays the 90-minute-earlier time string.
"""

time = raw_input('Enter the time in the form hh:mm : ')
# Extract the "show up" time information
hour = int(time[0:2])  # 1 point
minute = int(time[3:5])  # 1 point
# Compute the appoint information
if minute>=30:
    M = minute-30  # 1 point
    H = hour-1  # 1 point
else:
    M = minute+30  # 1 point
    H = hour-2  # 1 point
# If 1 o'clock is in between the arrival and appointment times then
```
# we have to correct H
if H<=0:
    H = H+12 # 1 point

# Produce length-2 encodings of H and M
if H<10: # 1 point
    hh = '0'+str(H)
else:
    hh = str(H)
if M<10: # 1 point
    mm = '0'+str(M)
else:
    mm = str(M)
Appointment = hh+':'+mm # 1 point
print 'You missed your %s appointment! '% Appointment
3 Pluralizer

3.1 Grading guidelines

10 points total

• 2 points for changing ‘y’ to ‘ies’
• 2 points for adding ‘oes’ when word ends with ‘ro’, ‘to’, or ‘no’
• 1 point for adding ‘es’ when word ends with ‘s’, ‘h’, or ‘x’
• 2 points for changing ‘fe’ to ‘vex’
• 1 point for adding ‘s’ to all other cases
• 2 points for correctly printing the line: "The plural of X is Xs"
• -1 point for any unnecessary case

3.2 Sample code

# Pluralizer.py
# Charles VanLoan
# January 1, 2015

""" Solicits a string using keyboard input and then displays its plural according to a set of five rules. """

noun = raw_input('Enter a string: ')
n = len(noun)
# These two substrings have a big role to play.
Last1 = noun[n-1]
Last2 = noun[n-2:]

# Step through the rules and assign to plural the "plural" of the string noun.
if Last1=='y':
    plural = noun[:(n-1)]+'ies' # 2 points
elif Last2=='no' or Last2=='ro' or Last2=='to':
    plural = noun+'es' # 2 points
elif Last1=='s' or Last1=='h' or Last1=='x':
    plural = noun + 'es' # 1 points
elif Last2=='fe':
    plural = noun + 'es' # 2 points
plural = noun[:n-2]+'ves'
else:                     # 1 point
    plural = noun + 's'
print 'The plural of "{}" is "{}".' % (noun, plural)  # 2 points