

```
File Edit Search Source Run Debug Consoles Projects Tools View Help

Editor - /home/BenGfoyle/Documents/Coding/Python/EP305/integrateCompare16306203.py

county.py x integrateCompare16306203.py x

109 """
110 Overview: Use the trapazoidal rule to integrate
111 """
112 def trapRule(dx,x):
113     fun_x = f(x)
114     fun_nextx = f(x + dx)
115     segment = (dx / 2) * (fun_x + fun_nextx)
116     return segment
117 #=====
118
119 #=====
120 """
121 Overview: Calcualte numeric answer based off trapazoidal rule
122 """
123 def trap(lower,upper,steps,dx):
124     segments = []
125     if steps % 2 == 1: #steps must be even
126         raise ValueError("Steps must be an even integer.")
127     for x in np.arange(lower,upper,dx):
128         currentSegment = trapRule(dx,x)
129         segments.append(currentSegment)
130     trapAns = sumSegments(segments)
131     return trapAns
132 #=====
133
134 #=====
135 """
136 Overview: Calcualte numeric answer based off simpson rule slightly different
137 method to trap rule same process different method
138 """
139 def simpson(lower,upper,steps,dx):
140     if steps % 2 == 1: #steps must be even
141         raise ValueError("Steps must be an even integer.")
142     x = np.linspace(lower,upper,steps+1) #get line space
143     y = f(x)
144     simpAns = dx/3 * np.sum(y[0:-1:2] + 4*y[1:2:2] + y[2::2]) #sum over ranges
145     return simpAns
146 #=====
147
148 #=====
149 """
150 Overview: Main method to call other functions and format the function calls
151 """
152 def main(lower,upper,steps,growth,numericError):
153
154     dx = (upper - lower)/steps #size of division
155     trapsAns = trap(lower,upper,steps,dx)
156     simpAns = simpson(lower,upper,steps,dx)
157     knownAns = analytical(lower,upper)
158
159     #calcualte errors
160     numericError = abs(trapsAns - simpAns)
161     simpError = abs(knownAns - simpAns)
162     trapsError = abs(knownAns - trapsAns)
163
164     printing(steps,lower,upper,trapsAns,simpAns,knownAns,numericError,\
```

```
Spyder (Python 2.7)

Python console

Python 3.7.2 (default, Jan 16 2019, 19:49:22)
Type 'copyright', 'credits' or 'license' for more information.
Python 3.7.0 -- An enhanced Interactive Python.
quickref -> Quick reference.
help -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.
NOTE: Spyder *can't* set your selected Matplotlib backend because there is a previous backend already in use.
Your backend will be Qt4Agg

In [11]: runfile('/home/BenGfoyle/Documents/Coding/Python/EP305/integrateCompare16306203.py', wdir='/home/BenGfoyle/Documents/Coding/Python/EP305')
This program will calculate the integral of sin(x) using the trapazoidal rule, you will enter limits and number of steps, a,b,n
Enter a value for lower limit, upper limit separate by commas (eg: 0,3.1415):0,3.1415
You have entered lower limit: 0.0 upper limit: 3.1415
Is this correct? [Y/N]y
Thank you

Steps | Lower Limit | Upper Limit | Trapezoidal | Simpson | Analytical | Numeric Error | Simpson Error | Trap Error
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----
2 | 0.00000 | 3.14159 | 1.570827766e+00 | 2.084581845e+00 | 1.999999999e+00 | 5.235590768e-01 | 9.438184725e-02 | 4.291772366e-01
4 | 0.00000 | 3.14159 | 1.863224805e+00 | 2.084581845e+00 | 1.999999999e+00 | 1.084341054e-01 | 4.550195156e-03 | 1.037493193e-01
8 | 0.00000 | 3.14159 | 1.974231322e+00 | 2.080269134e+00 | 1.999999999e+00 | 2.603601284e-02 | 2.691378999e-04 | 2.576687434e-02
16 | 0.00000 | 3.14159 | 1.993707190e+00 | 2.080015555e+00 | 1.999999999e+00 | 6.445658008e-03 | 1.659986623e-05 | 6.429276722e-03
32 | 0.00000 | 3.14159 | 1.998334513e+00 | 2.080001029e+00 | 1.999999999e+00 | 1.607577492e-03 | 1.032247440e-06 | 1.066544245e-03
64 | 0.00000 | 3.14159 | 1.999590408e+00 | 2.080000060e+00 | 1.999999999e+00 | 4.016521318e-04 | 6.452233877e-08 | 4.015876694e-04
128 | 0.00000 | 3.14159 | 1.999898603e+00 | 2.080000000e+00 | 1.999999999e+00 | 1.003979223e-04 | 4.031781797e-09 | 1.003939355e-04
256 | 0.00000 | 3.14159 | 1.999974897e+00 | 1.999999999e+00 | 1.999999999e+00 | 2.509853637e-05 | 2.519726505e-10 | 2.509828440e-05
512 | 0.00000 | 3.14159 | 1.999992721e+00 | 1.999999999e+00 | 1.999999999e+00 | 6.274575088e-06 | 1.571847474e-11 | 6.274559201e-06
1024 | 0.00000 | 3.14159 | 1.999998477e+00 | 1.999999999e+00 | 1.999999999e+00 | 1.568640080e-06 | 9.841016890e-13 | 1.568630044e-06
2048 | 0.00000 | 3.14159 | 1.999999604e+00 | 1.999999999e+00 | 1.999999999e+00 | 3.921597849e-07 | 6.158635550e-14 | 3.921597234e-07
4096 | 0.00000 | 3.14159 | 1.999999898e+00 | 1.999999999e+00 | 1.999999999e+00 | 9.803983844e-08 | 2.998882899e-15 | 9.803982644e-08
8192 | 0.00000 | 3.14159 | 1.999999971e+00 | 1.999999999e+00 | 1.999999999e+00 | 2.450997560e-08 | 4.448892099e-16 | 2.450997516e-08
16384 | 0.00000 | 3.14159 | 1.999999990e+00 | 1.999999999e+00 | 1.999999999e+00 | 6.127487486e-09 | 2.228446049e-16 | 6.127487184e-09

Steps | Lower Limit | Upper Limit | Trapezoidal | Simpson | Analytical | Numeric Error | Simpson Error | Trap Error
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----
2 | 0.00000 | 3.14159 | 1.570827766e+00 | 2.084581845e+00 | 1.999999999e+00 | 5.235590768e-01 | 9.438184725e-02 | 4.291772366e-01
4 | 0.00000 | 3.14159 | 1.863224805e+00 | 2.080269134e+00 | 1.999999999e+00 | 1.084341054e-01 | 4.550195156e-03 | 1.037493193e-01
20 | 0.00000 | 3.14159 | 1.995888211e+00 | 2.080086779e+00 | 1.999999999e+00 | 4.120568160e-03 | 6.783648282e-06 | 4.113784520e-03
200 | 0.00000 | 3.14159 | 1.999830972e+00 | 1.999999999e+00 | 1.999999999e+00 | 1.630242008e-04 | 6.762918314e-10 | 1.630235344e-04
2000 | 0.00000 | 3.14159 | 1.999999584e+00 | 1.999999999e+00 | 1.999999999e+00 | 4.112934151e-07 | 6.772360450e-14 | 4.112927376e-07
20000 | 0.00000 | 3.14159 | 1.999999992e+00 | 1.999999999e+00 | 1.999999999e+00 | 1.000000000e+00 | 0.000000000e+00 | 4.112128208e-09

In [12]:
```

```
Activities Spyder

File Edit Search Source Run Debug Consoles Projects Tools View Help

Editor - /home/BenGfoyle/Documents/Coding/Python/EP305/integrateCompare16306203.py

county.py x integrateCompare16306203.py x

4 Created on Tue Mar 26 13:32:37 2019
5
6 @author: BenGfoyle
7
8 Overview: This program calculates the integral of sin(x) for a given interval
9 and number of steps, using the trapazoid rule, and simpson 1/3 rule, then a
10 comparison will be made into the two rules
11 """
12
13 import numpy as np #used for mathematical functions
14
15 #=====
16 """
17 Overview: Inform the user what the program will do
18 """
19 def informUser():
20     print("This program will calculate the integral of sin(x) using the",\
21           "trapazoid rule, you will enter limits and number of steps, a,b,n")
22 #=====
23
24 #=====
25 """
26 Overview: This function gets the user inputs for limits and number of steps
27 """
28 def newInput(): #user input with cluases for bad input
29     prompt = "Enter a value for lower limit, upper limit\
30             " seperate by commas (eg: 0,3.1415):"
31     values = input(prompt)
32     try: #try split input into parts
33         values = values.split(",")
34         lower = float(values[0])
35         upper = float(values[1])
36         print("You have entered lower limit:",lower,"upper limit:",upper)
37         ans = input("Is this correct? [Y/N]") #confirm input is correct
38         if ans == "Y" or ans == "y":
39             print("Thank you")
40             return lower,upper #return values
41         else:
42             print("Please enter your values")
43             newInput()
44     except ValueError: #bad values or an inforseen error occurred
45         print("\nYou may have entered an invalid value please try again!")
46         return newInput()
47 #=====
48
49 #=====
50 """
51 Overview: Sum up all elements in a lists
52 """
53 def sumSegments(addUp):
54     x = 0
55     for i in range(0,len(addUp)):
56         x += addUp[i]
57     return x
58 #=====
```

```
Editor - /home/BenGfoyle/Documents/Coding/Python/EP305/integrateCompare16306203.py

county.py x integrateCompare16306203.py x

51 """
52 Overview: Sum up all elements in a lists
53 """
54 def sumSegments(addUp):
55     x = 0
56     for i in range(0,len(addUp)):
57         x += addUp[i]
58     return x
59 #=====
60
61 #=====
62 """
63 Overview: Used to calculate an answer forma known solution
64 """
65 def analytical(lower,upper):
66     ans = np.cos(lower) - np.cos(upper)
67     return ans
68 #=====
69
70 #=====
71 """
72 Overview: Print the output in a visually appealing way
73 """
74 def printing(head1,head2,head3,head4,head5,head6,head7,head8,head9):
75
76     try: #for printing numbers
77         print("{0:>15}".format((head1)),',',\
78               "{0:>15.5f}".format((head2)),',',\
79               "{0:>15.5f}".format((head3)),',',\
80               "{0:>15.9e}".format((head4)),',',\
81               "{0:>15.9e}".format((head5)),',',\
82               "{0:>15.9e}".format((head6)),',',\
83               "{0:>15.9e}".format((head7)),',',\
84               "{0:>15.9e}".format((head8)),',',\
85               "{0:>15.9e}".format((head9)))
86     except: #for printing strings
87         print("{0:>15}".format((head1)),',',\
88               "{0:>15}".format((head2)),',',\
89               "{0:>15}".format((head3)),',',\
90               "{0:>15}".format((head4)),',',\
91               "{0:>15}".format((head5)),',',\
92               "{0:>15}".format((head6)),',',\
93               "{0:>15}".format((head7)),',',\
94               "{0:>15}".format((head8)),',',\
95               "{0:>15}".format((head9)))
96
97 #=====
98
99 #=====
100 """
101 Overview: Calculate sin(x) using numpy
102 """
103 def f(x):
104     ans = np.sin(x)
105     return ans
106 #=====
```

Activities

Chromium Web Browser

File Edit Search Source Run Debug Consoles Projects Tools View Help

Editor - /home/BenGroyle/Documents/Coding/Python/EP305/IntegrateCompare6506203.py

IntegrateCompare6506203.py

```
128 currentSegment = traps[1:dX+1]
129 segments.append(currentSegment)
130 return traps
131
132 #=====
133
134 #=====
135 """
136 Overview: Calculate numeric answer based off simpson rule slightly different
137 period to trap rule same process different method
138 """
139 def simpson(lower,upper,steps,dX):
140     if steps % 2 == 1: #steps must be even
141         x = math.ceil(steps/2)
142         y = f(x)
143         simpans = dX/2 * np.sum(y[0:-1:2] + 4*y[1:-1:2] + y[2:-1:2]) #sum over ranges
144         return simpans
145
146 #=====
147
148 """
149 Overview: Main method to call other functions and format the function calls
150 """
151 def main(lower,upper,steps,growth,numerError):
152     dX = (upper - lower)/steps #size of division
153     traps = trap(lower,upper,steps,dX)
154     simpans = simpson(lower,upper,steps,dX)
155     knowans = analytical(lower,upper,dX)
156     knowans = analytical(lower,upper)
157
158     #calculate errors
159     simpError = abs(traps - simpans)
160     simpError = abs(knowans - trapsans)
161     trapError = abs(knowans - trapsans)
162
163     printing(steps,lower,upper,trapsans,simpans,knowans,numerError,\
164             simpError,simpError)
165
166     if numerError > 1e-8: #run until defined level of uncertainty
167         return main(lower,upper,steps * growth,growth,numerError)
168
169 #=====
170
171
172 steps = 2 #default starting step value
173 lower = 0
174 upper = math.pi
175 printing("Steps","Lower Limit","Upper Limit","Trapsans","Simpson",\
176         "Analytical","Numeric Error","Simpson Error","Trap Error")
177 main(lower,upper,steps,2,0)
178 printing("Steps","Lower Limit","Upper Limit","Trapsans","Simpson",\
179         "Analytical","Numeric Error","Simpson Error","Trap Error")
180 main(lower,upper,steps,10,0)
181 print("\n")
```

Python type: Python3
Back to top
NOTE: This is a Python 3 script.
Enter your IPython command here.

Tue Apr 2 11:22

EDHRE: x EDHRE: x EDHRE: x WAR CE: x typele: x channel: x Will Wa: x Assign: x

https://2019.moocloud.university.lemod/assign/view.php?id=49001

Maynooth University
National University of Ireland Maynooth

MY DASHBOARD ARCHIVES HELP MY COURSES

BENJAMIN WILLIAM GUIROYLE

Grade 90.00 / 100.00

Grade Monday, 1 April 2019, 9:50 AM on

Grade by Frank Mulligan

Feedback comments

Hello Ben,

Very nice work. Each function is clearly coded.

Good use of user-defined functions, possibly taking the functions idea a bit too far! You might consider passing the function (f) to the function trap() and simpson()! I suggest this to you because you are well able to code.

Advise user to enter values of a and b in radians.

Each doc string should occur immediately after the def statement of each function. Then if you type the name of the function in the IPython Console, and enter Ctrl-H, the doc string that you have entered will appear in the help window.

Watch out for the correct spelling of "separate" in function newInput()

Other wise excellent code.

Well done.

Frank