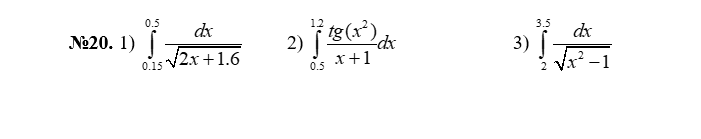
Пальонка Анастасія,2-8

Варіант 20

**Завдання:** з точністю до 0.0001 обчислитизначеннявизначенихінтегралів:  
1)методом прямокутників за умови *п*=10 (1 інтеграл) – 3 методи!;   
2) методом Сімпсона за умови *п*=8 (2 інтеграл);   
3)методом трапецій за умови *п*=20 (3 інтеграл);

4) написати код для цих методів.



1)Для обчислення інтегралу методом прямокутників за умови  розіб’ємо відрізок інтегрування на 10 рівних частин з кроком

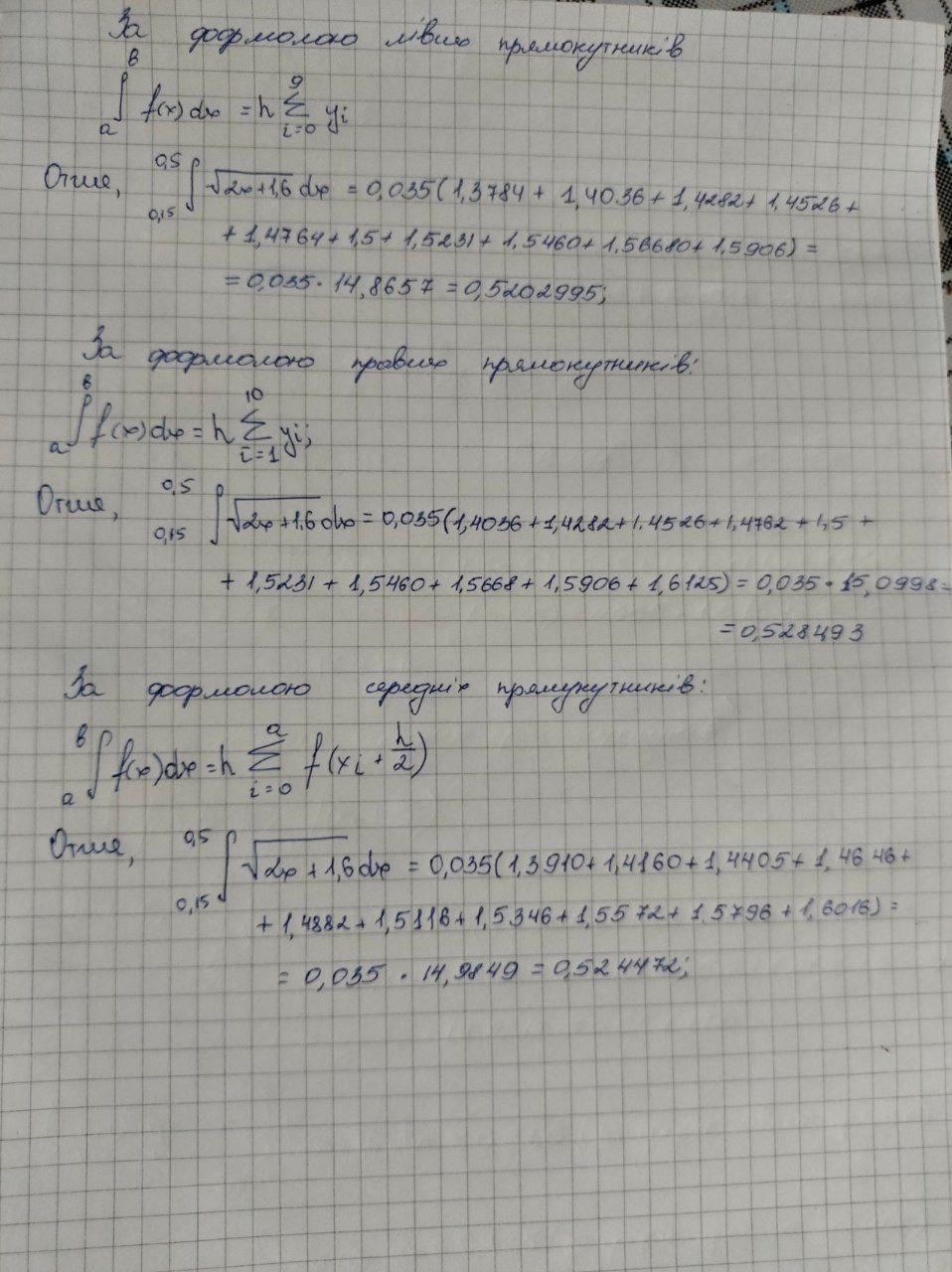
.

Складемо таблицю значень підінтегральної функції в точках ділення відрізка:

|  |  |  |
| --- | --- | --- |
| I | Xi |  |
| 0 | 0.15 | 1.3784 |
| 1 | 0.185 | 1.4036 |
| 2 | 0.22 | 1.4282 |
| 3 | 0.255 | 1.4526 |
| 4 | 0.29 | 1.4764 |
| 5 | 0.325 | 1.5 |
| 6 | 0.36 | 1.5231 |
| 7 | 0.395 | 1.5460 |
| 8 | 0.43 | 1.5668 |
| 9 | 0.465 | 1.5906 |
| 10 | 0.5 | 1.6125 |

Складемо таблицю значень підінтегральної функції в середніх точках відрізків ділення:

|  |  |  |
| --- | --- | --- |
| i |  |  |
| 0 | 0.1675 | 1,3910 |
| 1 | 0.2025 | 1,4160 |
| 2 | 0.2375 | 1.4405 |
| 3 | 0.2725 | 1.4646 |
| 4 | 0.3075 | 1.4882 |
| 5 | 0.3425 | 1.5116 |
| 6 | 0.3775 | 1.5346 |
| 7 | 0.4125 | 1.5572 |
| 8 | 0.4475 | 1.5796 |
| 9 | 0.4825 | 1.6016 |



#Обчислення інтеграла методом прямокутників

from scipy import integrate

import numpy as np

eps = 0.001

a = 0.15

b = 0.5

def f1(x):

return np.sqrt(2\*x+1.6)

def left\_rec(f1,a,b,n):

h=(b-a)/n

sum=0

for i in range(0,n):

sum+=f1(a+i\*h)

return sum\*h

v,err = integrate.quad(f1,a,b)#Перевірка

#Перевірка точності за правилом Рунге:

if abs(left\_rec(f1,a,b,2\*10) - left\_rec(f1,a,b,10))/3. <=eps:

print("left rectangle:",round (left\_rec(f1,a,b,10), 4))

def right\_rec(f1,a,b,n):

h=(b-a)/n

sum=0

for i in range(1,n+1):

sum+=f1(a+i\*h)

return sum\*h

print("right rectangle:",round (right\_rec(f1,a,b,10), 4))

def aver\_rec(f1,a,b,n):

h=0.035

sum=0

for i in range(0,n):

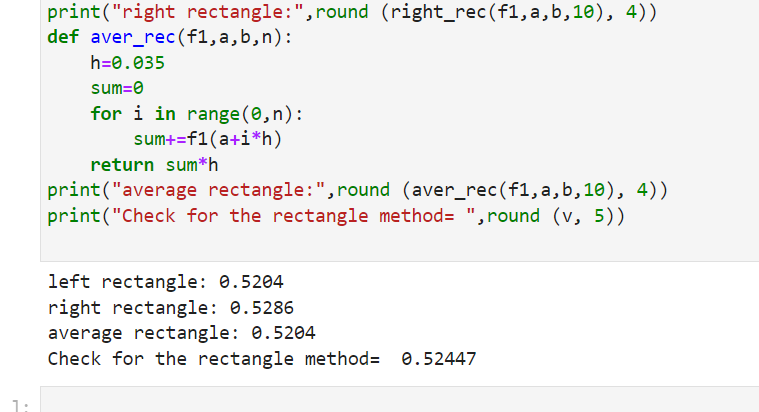
sum+=f1(a+i\*h)

return sum\*h

print("average rectangle:",round (aver\_rec(f1,a,b,10), 4))

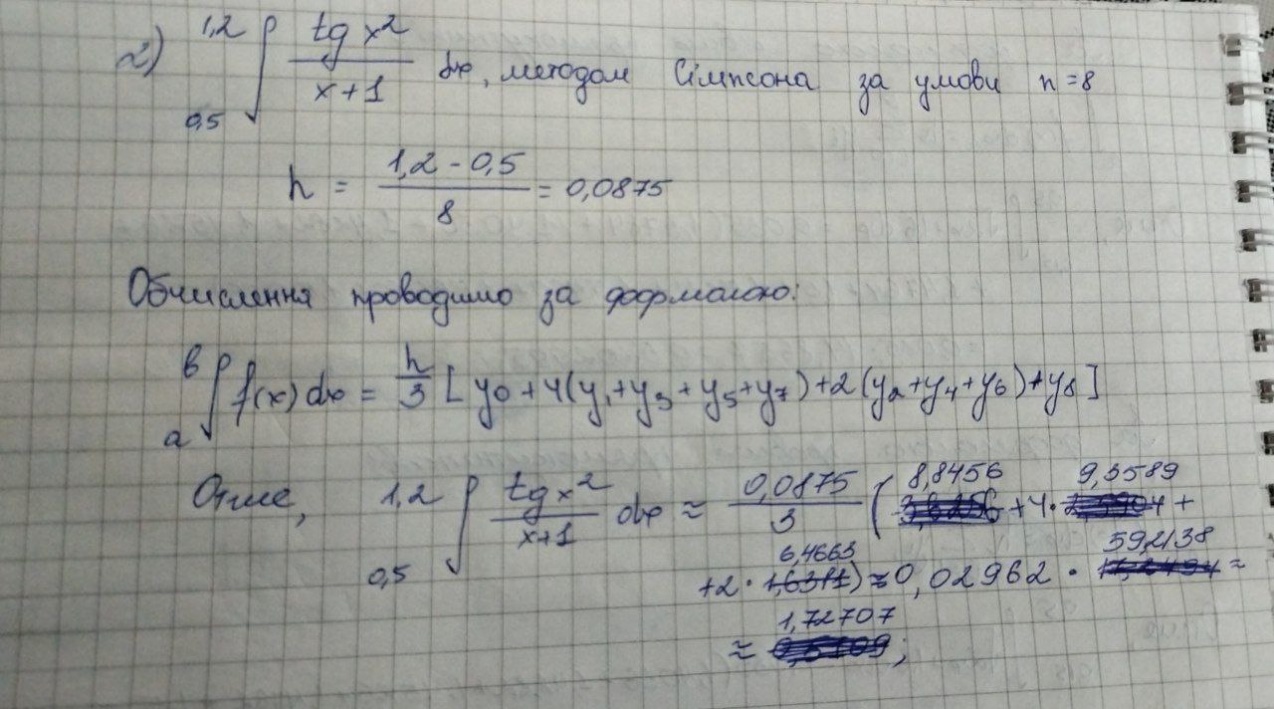
print("Check for the rectangle method= ",round (v, 5))





Складемо таблицю значень підінтегральної функції в точках ділення відрізка:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| 0 | 0.5 | 1.5107 |  |  |
| 1 | 0.5875 |  | 1.612 |  |
| 2 | 0.675 |  |  | 1.7259 |
| 3 | 0.7625 |  | 1.8619 |  |
| 4 | 0.85 |  |  | 2.0371 |
| 5 | 0.9375 |  | 2.2874 |  |
| 6 | 1.025 |  |  | 2.7033 |
| 7 | 1.1125 |  | 3.5976 |  |
| 8 | 1.2 | 7.3349 |  |  |
|  |  | 8.8456 | 9.3589 | 6.4663 |



#Обчислення інтеграла методом Симпсона

from scipy import integrate

import numpy as np

# Задаємо функцію, яку необхідно інтегрувати

def f(x):

return np.tan(x\*\*2)/x+1

# Задаємо межі інтегрування та початкову кількість розбиттів

a = 0.5

b = 1.2

n = 8

# Обчислюємо значення інтегралу методом Симпсона

def simpson\_rule(f, a, b, n):

h = (b - a) / n

integr = f(a) + f(b)

for i in range(1,n):

k = a + i\*h

if i%2 == 0:

integr += 2 \* f(k)

else:

integr += 4 \* f(k)

integr \*= h/3

return integr

# Обчислюємо значення інтегралу методом Сімпсона з точністю 0.001

integral1 = simpson\_rule(f, a, b, n)

n \*= 2

integral2 = simpson\_rule(f, a, b, n)

while abs(integral2 - integral1) / 15 > 0.001:

integral1 = integral2

n \*= 2

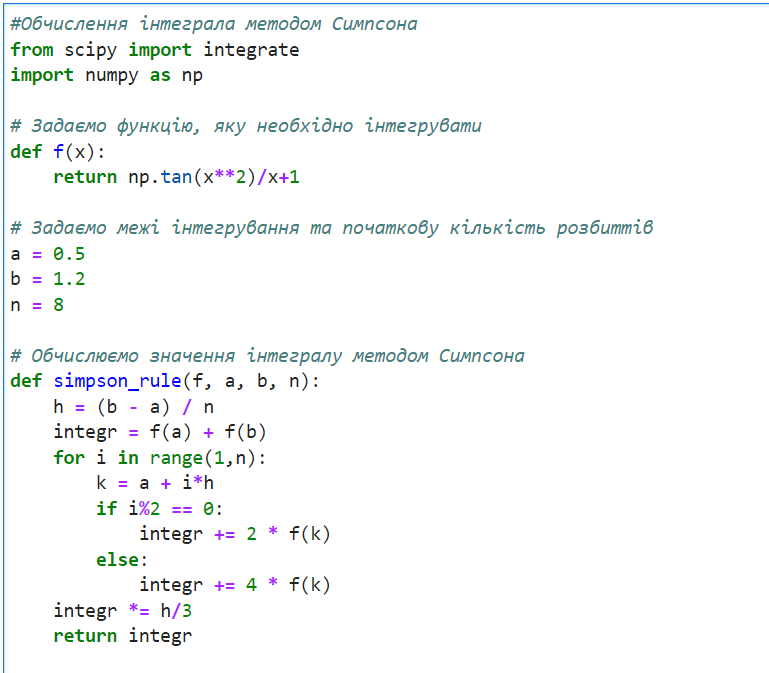
integral2 = simpson\_rule(f, a, b, n)

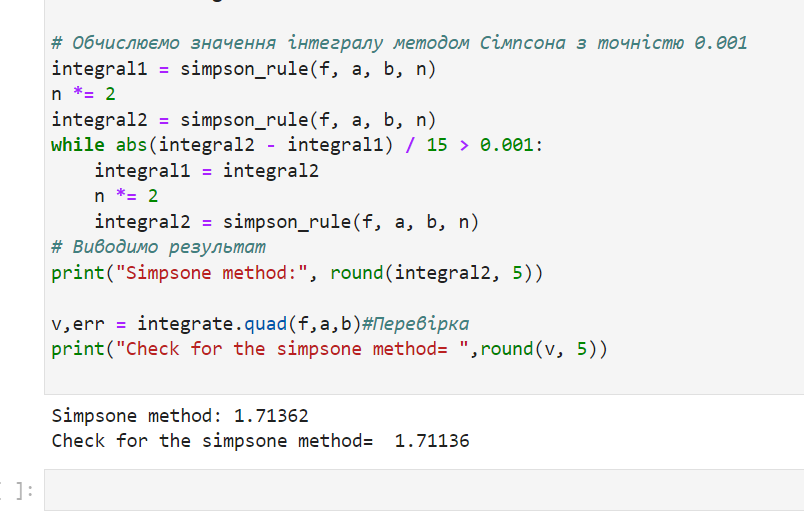
# Виводимо результат

print("Simpsone method:", round(integral2, 5))

v,err = integrate.quad(f,a,b)#Перевірка

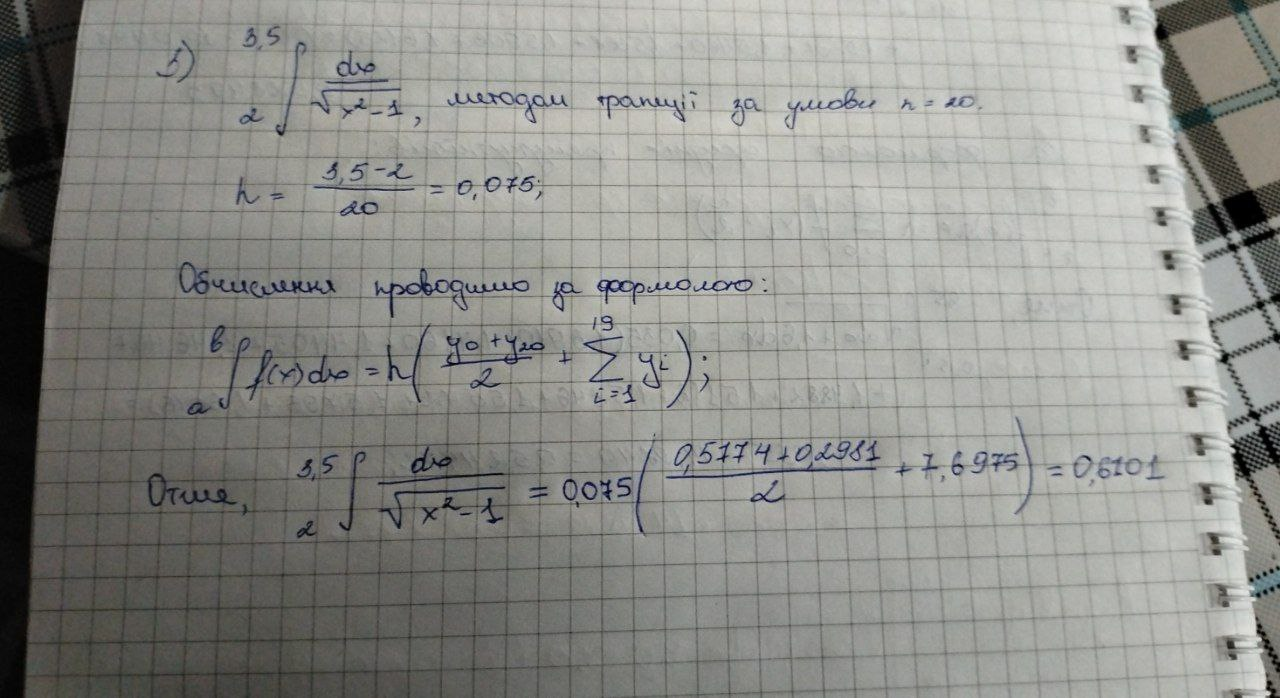
print("Check for the simpsone method= ",round(v, 5))





Складемо таблицю значень підінтегральної функції в точках ділення відрізка:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
| 0 | 2 | 0.5774 |  | 11 | 2.825 | 0.3785 |
| 1 | 2.075 | 0.55 |  | 12 | 2.9 | 0.3674 |
| 2 | 2.15 | 0.5254 |  | 13 | 2.975 | 0.3569 |
| 3 | 2.225 | 0.5031 |  | 14 | 3.05 | 0.3471 |
| 4 | 2.3 | 0.4828 |  | 15 | 3.125 | 0.3378 |
| 5 | 2.375 | 0.4642 |  | 16 | 3.2 | 0.329 |
| 6 | 2.45 | 0.4471 |  | 17 | 3.275 | 0.3207 |
| 7 | 2.525 | 0.4313 |  | 18 | 3.35 | 0.3128 |
| 8 | 2.6 | 0.4167 |  | 19 | 3.425 | 0.3053 |
| 9 | 2.675 | 0.4031 |  | 20 | 3.5 | 0.2981 |
| 10 | 2.75 | 0.3904 |  |  |  |  |



from scipy import integrate

import numpy as np

# Задаємо функцію, яку необхідно інтегрувати

def f(x):

return 1 / np.sqrt(x\*\*2 - 1)

# Задаємо межі інтегрування та точність

a = 2

b = 3.5

target\_precision = 0.001

# Початкова кількість розбиттів

n = 20

# Обчислюємо значення інтегралу методом трапецій

def trapezoidal\_rule(f, a, b, n):

h = (b - a) / n

x = a

integral\_prev = 0

integral\_current = f(a)

for i in range(1, n):

x += h

integral\_current += 2 \* f(x)

integral\_current += f(b)

integral\_current \*= h / 2

while abs(integral\_current - integral\_prev) > target\_precision:

n \*= 2

h = (b - a) / n

x = a

integral\_prev = integral\_current

integral\_current = f(a)

for i in range(1, n):

x += h

integral\_current += 2 \* f(x)

integral\_current += f(b)

integral\_current \*= h / 2

return integral\_current

# Обчислюємо і виводимо результати з заданою точністю

result1 = trapezoidal\_rule(f, a, b, n)

result2 = integrate.quad(f, a, b)[0] # Використання quad з бібліотеки scipy

print("Метод трапецій:", round(result1, 5))

print("Scipy quad:", round(result2, 5))

