САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Математический анализ

Отчет

по лабораторной работе №1

«Интеграл Римана»

Выполнил(а): Ступников Александр Сергеевич

студ. гр. M3135

Санкт-Петербург

2020

Аналитическая часть

Дана функция $f(x) = 4^x$. Найдём интеграл Римана данной функции на отрезке $[0, 2]$, как предел интегральной суммы

$$\delta_\tau = \sum_{i=1}^{n} f(\xi_i)\Delta x_i \text{ при } \lambda(\tau) \to 0.$$

Пусть $\Delta x_i = \frac{2}{n}\ \forall i, \xi_i = \frac{2i}{n}$ (правые точки).

$$\text{Тогда } \delta_\tau = \sum_{i=1}^{n} f\left(\frac{2i}{n}\right) \cdot \frac{2}{n} = \frac{2}{n}\sum_{i=1}^{n}\left(4^{\frac{2}{n}}\right)^i = \frac{2}{n} \cdot \frac{4^{\frac{2}{n}}\left(1 - \left(4^{\frac{2}{n}}\right)^n\right)}{1 - 4^{\frac{2}{n}}} = \frac{30 \cdot 4^{\frac{2}{n}}}{n\left(4^{\frac{2}{n}} - 1\right)}.$$

$$I = \int_0^2 4^x = \lim_{\lambda(\tau)\to 0} \delta_\tau = \lim_{n\to\infty} \delta_\tau = \lim_{n\to\infty} \frac{30 \cdot 4^{\frac{2}{n}}}{n\left(4^{\frac{2}{n}} - 1\right)} = \lim_{n\to\infty} \frac{30 \cdot 4^{\frac{2}{n}}}{n\left(e^{\frac{2\ln 4}{n}} - 1\right)} =$$

$$= \lim_{n\to\infty} \frac{30 \cdot 4^{\frac{2}{n}}}{n\frac{2\ln 4}{n}} = \frac{15}{\ln 4}.$$

Итак, интеграл функции $f(x) = 4^x$ на отрезке $[0, 2]$ существует и равен $\frac{15}{\ln 4}$.

Найдём интеграл данной функции по формуле Ньютона-Лейбница.

$$\int 4^x\,dx = \frac{4^x}{\ln 4} + C$$

$$\int_0^2 4^x\,dx = \frac{4^x}{\ln 4}\bigg|_0^2 = \frac{15}{\ln 4}$$

Как видно, значение интеграла, полученное по формуле Ньютона-Лейбница, совпадает с найденным ранее.

Практическая часть

Программе на вход поступает два параметра: число точек разбиения и способ выбора оснащения (левые (left), правые (right), средние (center), случайные (random) точки). На выходе программа выдаёт ответ в виде:

```
Calculated sum: 10.785632996237721 (10 iterations, center tagged)
```

После двоеточия выводится значение вычисленной суммы для выбранного разбиения и оснащения (указаны в скобках). Разбиение всегда равномерное.

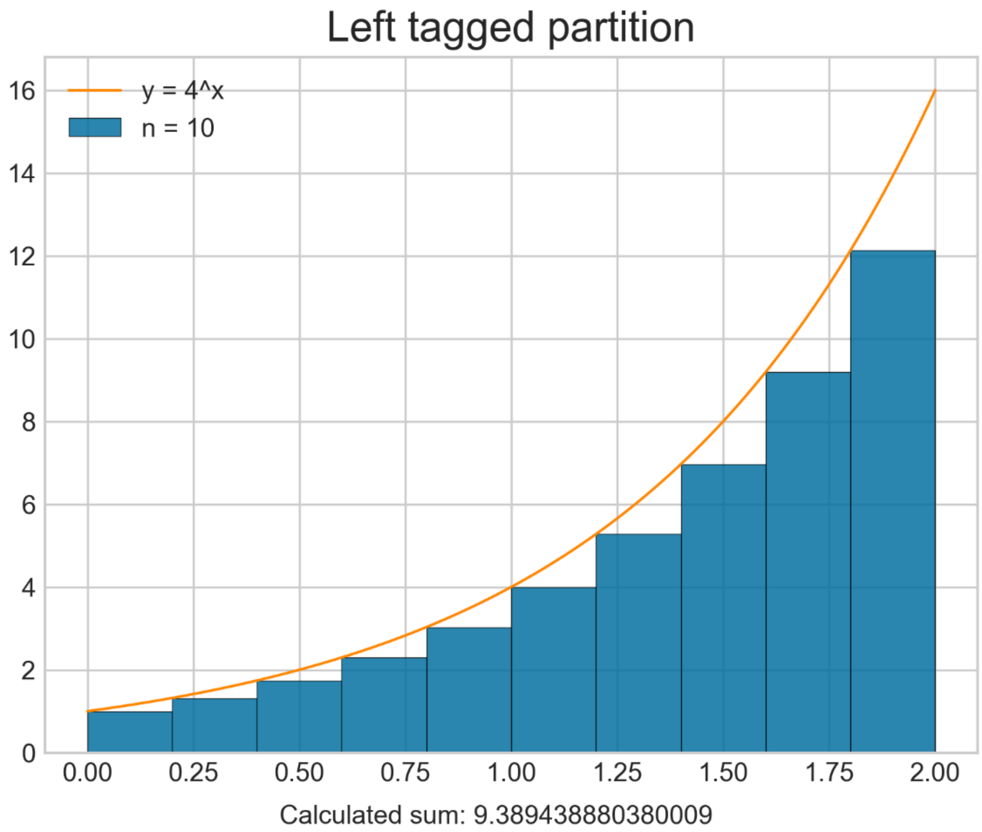
В директории, в которой была запущена программа создаётся файл вида:

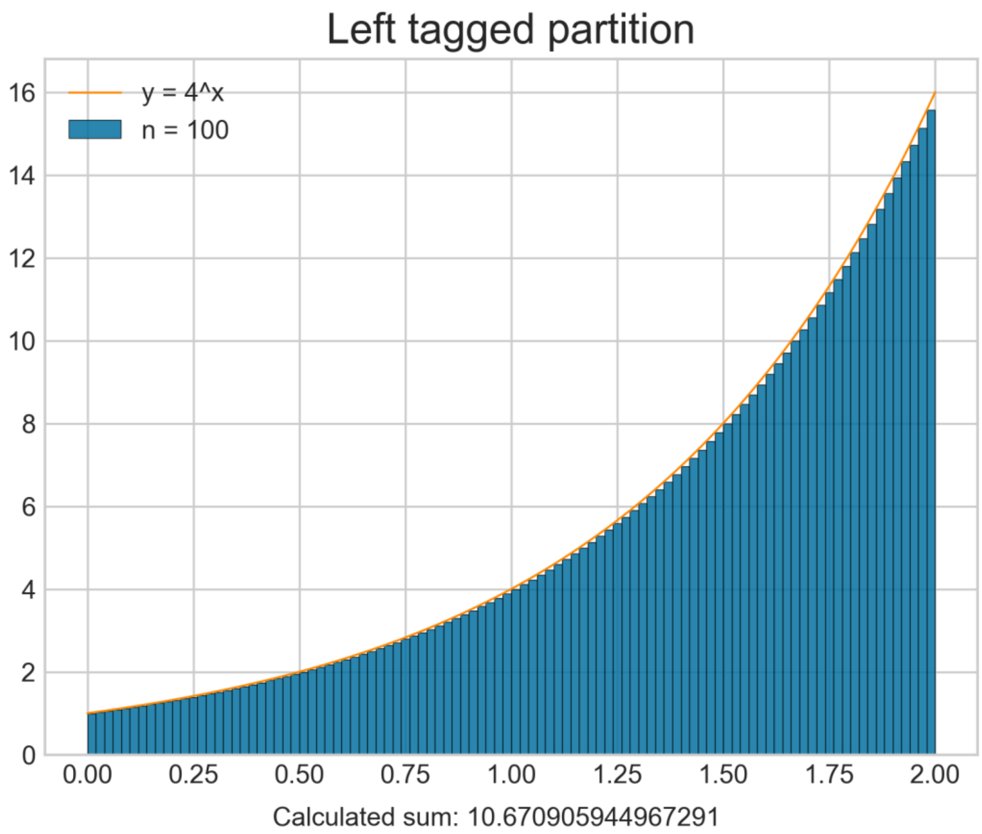
```
[способ оснащения]_tagged_partition(n=[число точек разбиения]).png
```

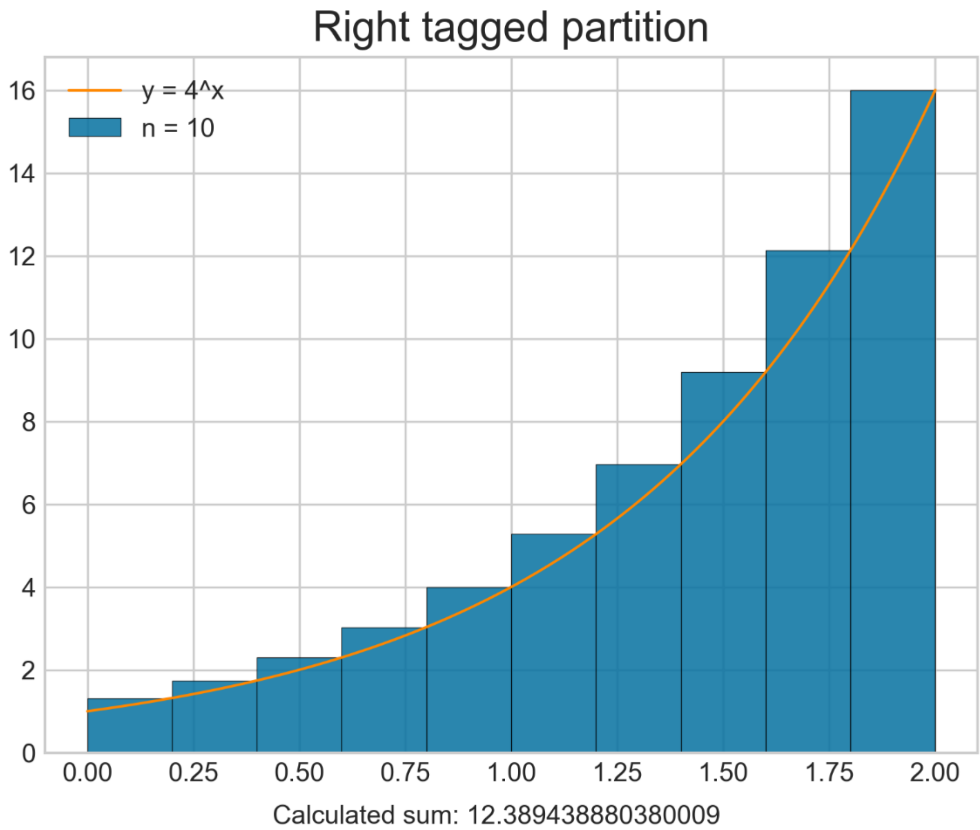
Файл содержит график функции $f(x) = 4^x$ (оранжевая линия), совмещённый с графиком слагаемых интегральной суммы (голубые прямоугольники) для этой функции. Все графики построены на отрезке $[0, 2]$ в соответствии с выбранным разбиением и оснащением. В нижней части изображения указано вычисленное значение интегральной суммы.

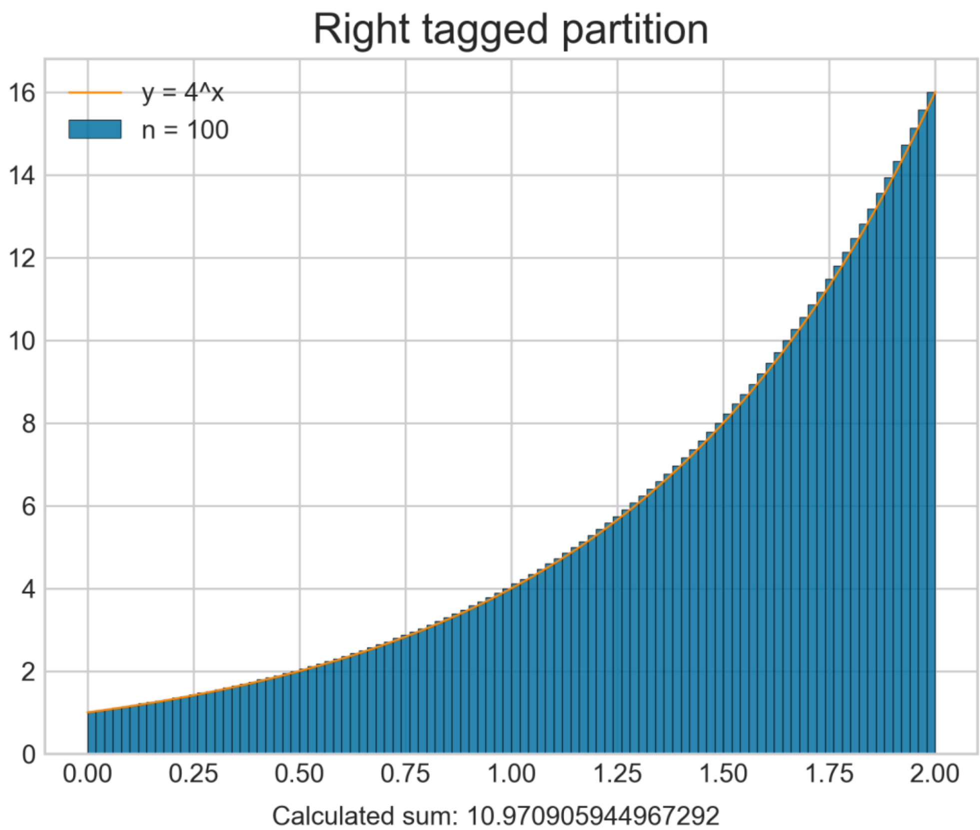
Нужно заметить, что если число точек разбиения превышает 300, то графики становятся приблизительными (это сделано для повышения производительности, и потому что при такой мелкости разбиения отдельные ступеньки фигур на графике в любом случаем становятся не различимы). Значение же интегральной суммы вычисляется точно в независимости от числа точек разбиения.

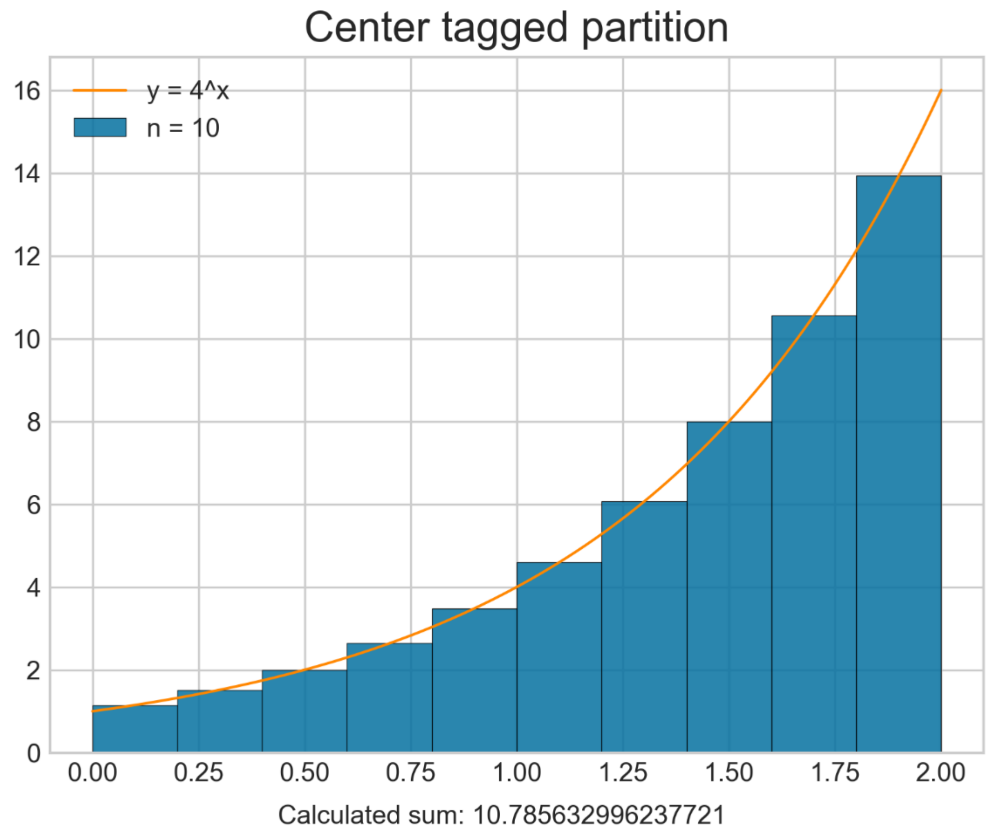
Далее приведены графики слагаемых интегральных сумм для различных разбиений и оснащений (указаны на графиках).

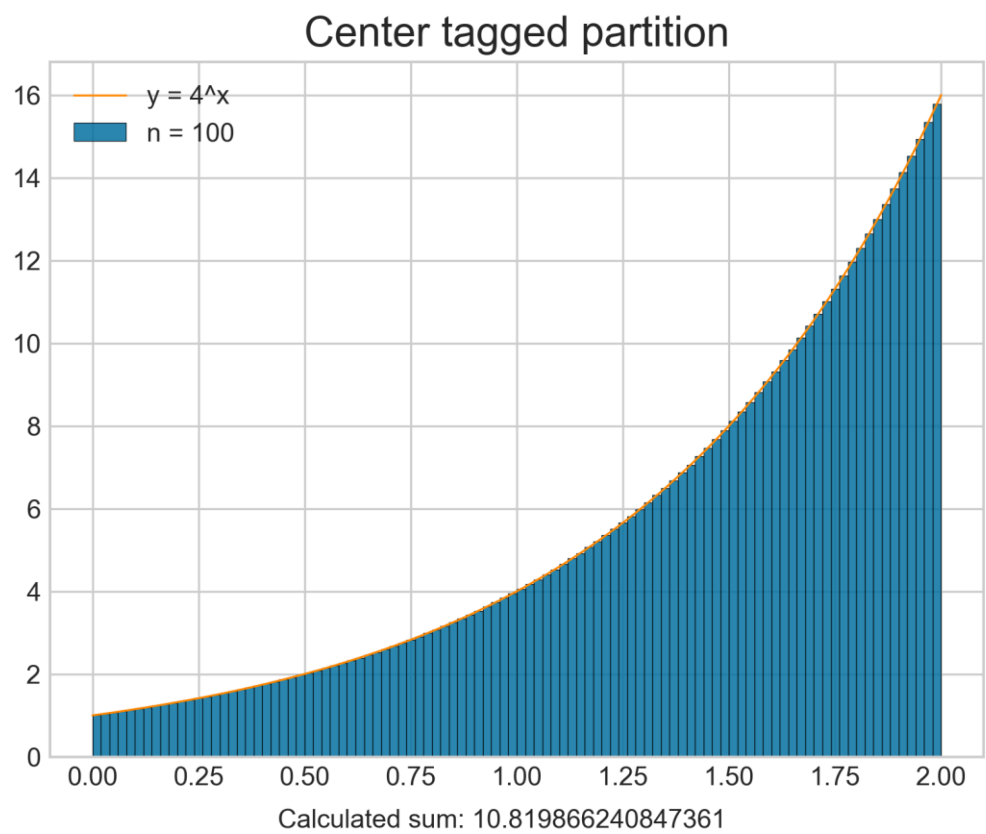

Left tagged partition
y = 4^x
n = 10
16
14
12
10
8
6
4
2
0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Calculated sum: 9.389438880380009


Left tagged partition
y = 4^x
n = 100
16
14
12
10
8
6
4
2
0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Calculated sum: 10.670905944967291

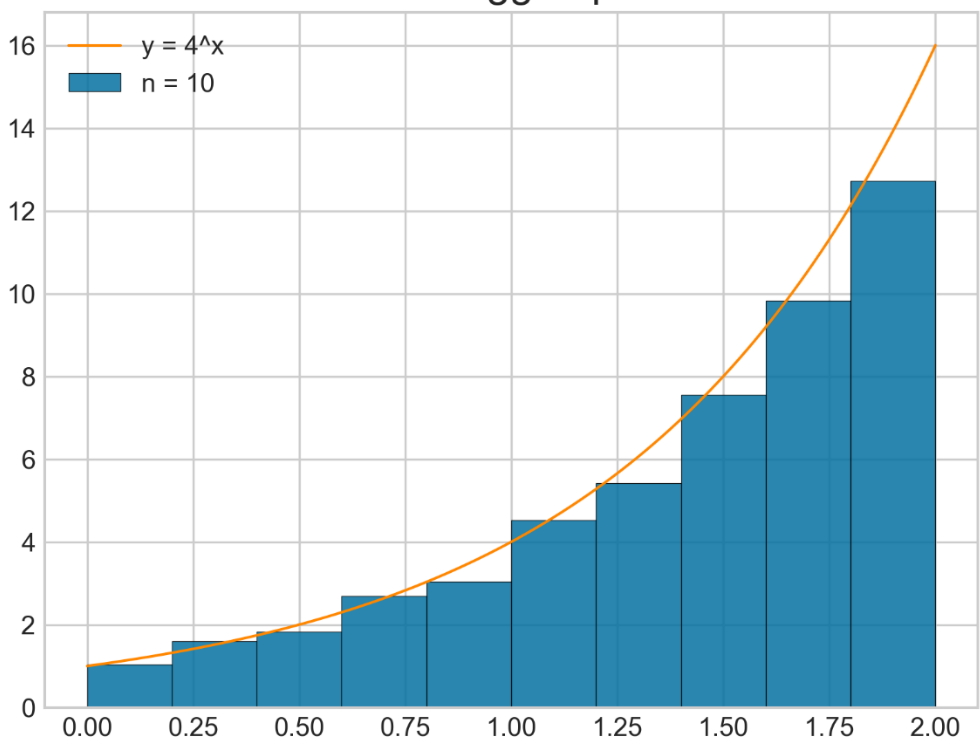

Right tagged partition
y = 4^x
n = 10
0
2
4
6
8
10
12
14
16
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Calculated sum: 12.389438880380009


Right tagged partition
y = 4^x
n = 100
0
2
4
6
8
10
12
14
16
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Calculated sum: 10.970905944967292


Center tagged partition
y = 4^x
n = 10
0
2
4
6
8
10
12
14
16
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Calculated sum: 10.785632996237721

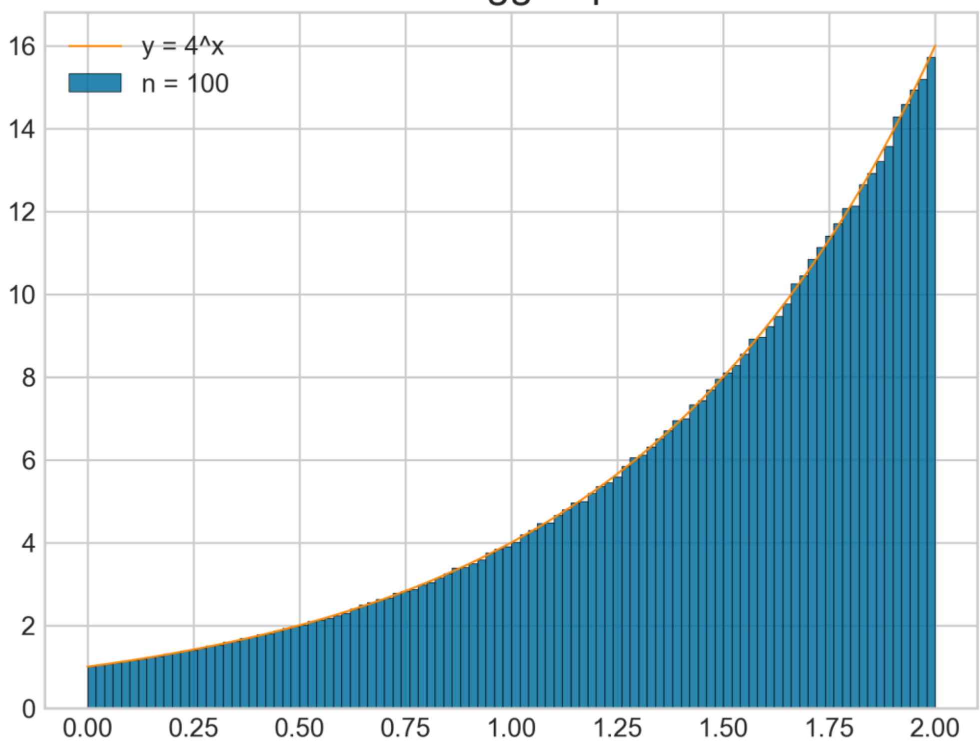

Center tagged partition
y = 4^x
n = 100
0
2
4
6
8
10
12
14
16
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Calculated sum: 10.819866240847361

Random tagged partition

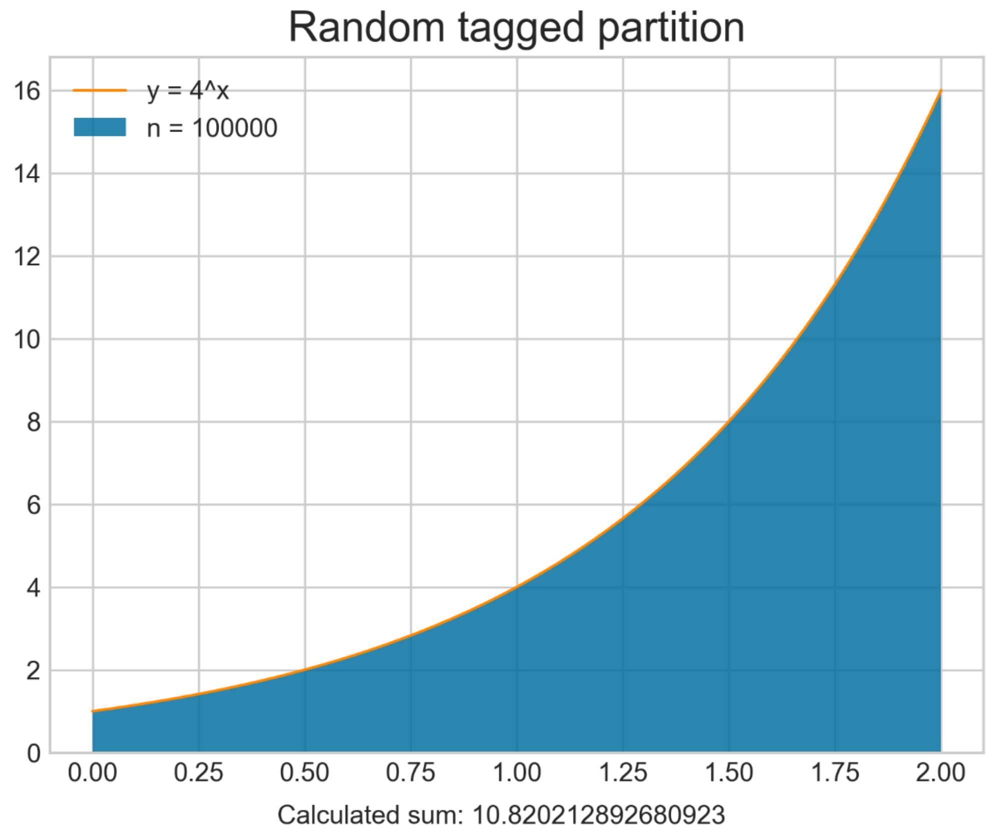


Calculated sum: 10.04968383784818

Random tagged partition



Calculated sum: 10.813451095939596



Листинг

Интерпретатор Python 3.9.2. Использованы библиотеки matplotlib 3.4.1 и numpy 1.20.1.

IntegralCalc.py

```
import numpy as np
import matplotlib.pyplot as plt
from math import pow
from sys import argv
plt.style.use('seaborn-whitegrid')

def genPartition(left, right, n):
      length = right - left
      part = np.array([])
      for i in range(n):
            part = np.append(part, left + i * length / n)
      part = np.append(part, right)
      return part

def draw_one(partition, tag, title = 'plot'):
      n = len(partition) - 1
      fig, ax = plt.subplots()

      res = sum(tag) * step
```

```
	m = n
	if n > 300:
		partition = genPartition(partition[0], partition[n], 500)
		n = len(partition) - 1
		tag = f(partition)[:n]
		ax.plot(x, fx, color = '#FF8700', linewidth = 1, label = 'y = 
4^x')
	else:
		ax.plot(x, fx, color = '#FF8700', linewidth = 1 - (n - 10) / 
400, label = 'y = 4^x')
	if n != m:
		ax.bar(partition[:n], tag, align = 'edge', width = 
(partition[n] - partition[0]) / n, color = '#0772A1',
			alpha = 0.86, label = f'n = {m}')
	else:
		ax.bar(partition[:n], tag, align = 'edge', width = 
(partition[n] - partition[0]) / n, color = '#0772A1',
			alpha = 0.86, edgecolor = 'black', linewidth = 0.3, label 
= f'n = {n}')
	ax.set_title(title, fontsize = 16)
	fig.suptitle(f'Calculated sum: {res}', x = 0.5, y = 0.025, 
verticalalignment = 'bottom', fontsize = 10)
	ax.legend()
	plt.savefig(f'./{title.replace(" ", "_")}(n={m}).png', dpi = 300)
	return res

def error(msg = ''):
	print(msg)
	print(f'Usage: "py {argv[0]} [number of dots in partition] [tag: 
left, right, center, random]"')

if len(argv) != 3:
	error('Invalid number of arguments')
else:
	if not(argv[1].isdigit()):
		error('Invalid number of dots in partition: ' + argv[1])
	else:
		f = lambda x: np.power(4, x)
		rng = [0, 2]
		n = int(argv[1])

		step = (rng[1] - rng[0]) / n
		partition = genPartition(rng[0], rng[1], n)

		x = genPartition(rng[0], rng[1], 10000)
		fx = f(x)
		if argv[2] == 'left':
			print(f'Calculated sum: {draw_one(partition, 
f(partition)[:n], "Left tagged partition")} ({n} iterations, {argv[2]} 
tagged)')
		elif argv[2] == 'right':
			print(f'Calculated sum: {draw_one(partition, f(partition 
+ step)[:n], "Right tagged partition")} ({n} iterations, {argv[2]} 
tagged)')
```

```
            elif argv[2] == 'center':
                  print(f'Calculated sum: {draw_one(partition, f(partition
+ step / 2)[:n], "Center tagged partition")} ({n} iterations, {argv[2]}
tagged)')
            elif argv[2] == 'random':
                  print(f'Calculated sum: {draw_one(partition, f(partition
+ np.random.random(n + 1) * step)[:n], "Random tagged partition")} ({n}
iterations, {argv[2]} tagged)')
            else:
                  error('Invalid tag: ' + argv[2])
```