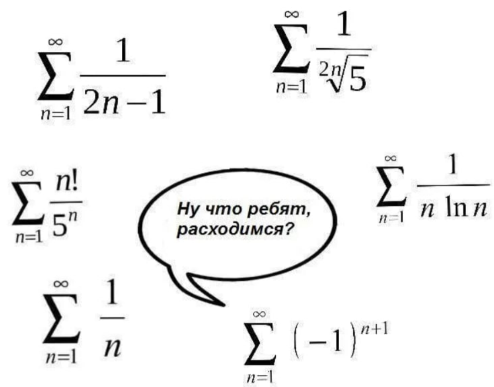


Лабораторная работа - 1. Интеграл Римана.
Вариант - 11.

Даниил Климов N3147

Аналитическая часть

Построить верхнюю и нижнюю суммы Дарбу для равномерного разбиения (на n частей)

Для функции $f(x) = \sin^2(2x)$ и равномерного разбиения интервала $[a, b]$ на n частей, верхняя и нижняя суммы Дарбу могут быть записаны в следующем виде:
Нижняя сумма Дарбу:

$$s_\tau(f) = \frac{b-a}{n}\sum_{i=1}^{n} m_i \Delta x_i = \frac{b-a}{n}\sum_{i=1}^{n}\left(\inf_{x\in[x_{i-1},x_i]} f(x)\right)$$

Верхняя сумма Дарбу:

$$S_\tau(f) = \frac{b-a}{n}\sum_{i=1}^{n} M_i \Delta x_i = \frac{b-a}{n}\sum_{i=1}^{n}\left(\sup_{x\in[x_{i-1},x_i]} f(x)\right)$$

Здесь a - левый конец отрезка, а b - правый конец отрезка $[a; b]$, $x_{i-1} = a + (i-1)\frac{b-a}{n}$ и $x_i = a + i\frac{b-a}{n}$ обозначают границы i-го подотрезка, а $\inf_{x\in[x_{i-1},x_i]} f(x)$ и $\sup_{x\in[x_{i-1},x_i]} f(x)$ являются инфинум и супремум значением функции $f(x)$ на i-ом подотрезке соответственно.

Заметим, что $f(x) = \sin^2(2x)$ имеет период равный $\frac{\pi}{2}$, поэтому для подсчёта суммы мы можем посчитать один период и умножить его на $2\pi \cdot \frac{2}{\pi} = 4$
Тогда нижняя и верхняя суммы Дарбу для $f(x) = \sin^2(2x)$ на отрезке $[0; 2\pi]$ примут такой вид:

$$s_\tau(f) = \frac{4\cdot 2\pi}{n}\sum_{i=1}^{\frac{n}{4}}\left(\inf_{x\in[x_{i-1},x_i]} f(x)\right)$$

$$S_\tau(f) = \frac{4\cdot 2\pi}{n}\sum_{i=1}^{\frac{n}{4}}\left(\sup_{x\in[x_{i-1},x_i]} f(x)\right)$$

Также заметим, что $f(x) = \sin^2(2x)$ возрастает на $[0; \frac{\pi}{4}]$ и убывает на $[\frac{\pi}{4}; \frac{\pi}{2}]$
Поэтому для подсчёта сумм мы можем разбить её на две подсуммы на отрезках $[0; \frac{\pi}{4}]$ и $[\frac{\pi}{4}; \frac{\pi}{2}]$. Перепишем их, учитывая это, и подставим индексы с учётом $\inf_{x\in[x_{i-1},x_i]} f(x)$ и $\sup_{x\in[x_{i-1},x_i]} f(x)$. На промежутке возрастания $\inf = f(x_{i-1}), \sup = f(x_i)$, а на промежутке убывания $\inf = f(x_i), \sup = f(x_{i-1})$.

$$s_\tau(f) = \frac{2\pi}{n}\cdot 4\cdot\left(\sum_{i=1}^{\frac{n}{8}}\left(f(x_{i-1}) + \sum_{i=\frac{n}{8}}^{\frac{n}{4}}\left(f(x_i)\right)\right)\right)$$

$$S_\tau(f) = \frac{2\pi}{n}\cdot 4\cdot\left(\sum_{i=1}^{\frac{n}{8}}\left(f(x_i) + \sum_{i=\frac{n}{8}}^{\frac{n}{4}}\left(f(x_{i-1})\right)\right)\right)$$

Для подсчёта сумм воспользуемся известной формулой суммы квадратов синусов для конечного набора углов с помощью формулы:

$$\sum_{i=1}^{n}\sin^2(\phi_i) = \frac{n}{2} - \frac{1}{2}\sum_{i=1}^{n}\cos(2\phi_i), \text{ где } \phi_i \text{ это угол из набора}$$

Перепишем формулы сумм, в конечном виде:

$$s_\tau(f) = \frac{2\pi}{n}\cdot 4\cdot\left(\frac{n}{16} - \frac{1}{2}\sum_{i=1}^{\frac{n}{8}}\cos(32\cdot\frac{i\pi}{n}) + \frac{n}{16} - \frac{1}{2}\sum_{\frac{n}{8}}^{\frac{n}{4}}\cos(32\cdot\frac{(i-1)\pi}{n})\right) =$$

$$= \frac{2\pi}{n}\left(\frac{n}{2} - 2(\sum_{i=1}^{\frac{n}{8}} \cos(32 \cdot \frac{i\pi}{n}) + \sum_{\frac{n}{8}}^{\frac{n}{4}} \cos(32 \cdot \frac{(i-1)\pi}{n}))\right) = \frac{2\pi}{n}\left(\frac{n}{2} - 2 \cdot 0\right) = \frac{2\pi}{n} \cdot \frac{n}{2} = \boxed{\pi}$$

$$S_\tau(f) = \frac{2\pi}{n} \cdot 4 \cdot \left(\frac{n}{16} - \frac{1}{2}\sum_{i=1}^{\frac{n}{8}} \cos(32 \cdot \frac{i\pi}{n}) + \frac{n}{16} - \frac{1}{2}\sum_{\frac{n}{8}}^{\frac{n}{4}} \cos(32 \cdot \frac{(i-1)\pi}{n})\right) =$$

$$= \frac{2\pi}{n}\left(\frac{n}{2} - 2(\sum_{i=1}^{\frac{n}{8}} \cos(32 \cdot \frac{i\pi}{n}) + \sum_{\frac{n}{8}}^{\frac{n}{4}} \cos(32 \cdot \frac{(i-1)\pi}{n}))\right) = \frac{2\pi}{n}\left(\frac{n}{2} - 2 \cdot 0\right) = \frac{2\pi}{n} \cdot \frac{n}{2} = \boxed{\pi}$$

Проверить критерий Римана интегрируемости функции, сделать вывод. Как ещё можно доказать интегрируемость данной функции?

Критерий Римана интегрируемости функции:

$$\forall \epsilon > 0 \exists \tau : S_\tau - s_\tau < \epsilon$$

Проверим для данной в условии функции, учитывая найденные ранее s_τ и S_τ:

$$S_\tau - s_\tau = \pi - \pi = 0$$

Тривиально, что это будет меньше, чем $\forall \epsilon > 0$. А значит, получили, что критерий Римана выполняется для $f(x) = \sin^2(2x)$ на отрезке $[0; 2\pi]$.
Ещё способы доказать интегрируемость $f(x) = \sin^2(2x)$ на отрезке $[0; 2\pi]$:
Теорема об интегрируемости непрерывной функции:

$$f(x) = \sin^2(2x) \in C[0; 2\pi] \Rightarrow f(x) = \sin^2(2x) \in R[0; 2\pi]$$

Теорема о конечном числе точек разрыва:
Для доказательства интегрируемости функции $f(x) = \sin^2(2x)$ на отрезке $[0, 2\pi]$ достаточно показать, что она является ограниченной на данном отрезке и имеет только конечное число точек разрыва первого рода.

Ограниченность функции $f(x)$ на отрезке $[0, 2\pi]$ следует из того, что $0 \le \sin^2(2x) \le 1$ для любого $x \in [0, 2\pi]$. Таким образом, функция $f(x)$ ограничена на отрезке $[0, 2\pi]$ сверху и снизу числами 1 и 0.

Далее, функция $f(x)$ имеет только конечное число точек разрыва первого рода на отрезке $[0, 2\pi]$, поскольку $\sin(2x)$ имеет только конечное число точек разрыва первого рода на этом отрезке, а квадрат функции $\sin(2x)$ также имеет только конечное число таких точек.

Таким образом, функция $f(x)$ удовлетворяет условиям теоремы, и следовательно, является интегрируемой на отрезке $[0, 2\pi]$.

Найти пределы сумм Дарбу, сделать вывод о значении интеграла.

$$\lim_{\lambda(\tau)\to 0} S_\tau = \lim_{\lambda(\tau)\to 0} \frac{2\pi}{n}\frac{n}{2} = \lim_{\lambda(\tau)\to 0} \pi = \pi$$

$$\lim_{\lambda(\tau)\to 0} s_\tau = \lim_{\lambda(\tau)\to 0} \frac{2\pi}{n}\frac{n}{2} = \lim_{\lambda(\tau)\to 0} \pi = \pi$$

Логичный вывод, что $I = \pi$

Проверить результат с помощью формулы Ньютона — Лейбница.

$$\int_0^{2\pi} \sin^2(2x)dx = \frac{1}{2}\int_0^{2\pi} 1 - \cos(4x)dx = \frac{1}{2}\left(x - \frac{1}{4}\sin(4x)\right)\Bigg|_0^{2\pi} = \boxed{\pi}$$

Получили, что пределы сумм Дарбу совпадают с определенным интегралом, посчитанным по формуле Ньютона-Лейбница.

Численный метод

Таблица

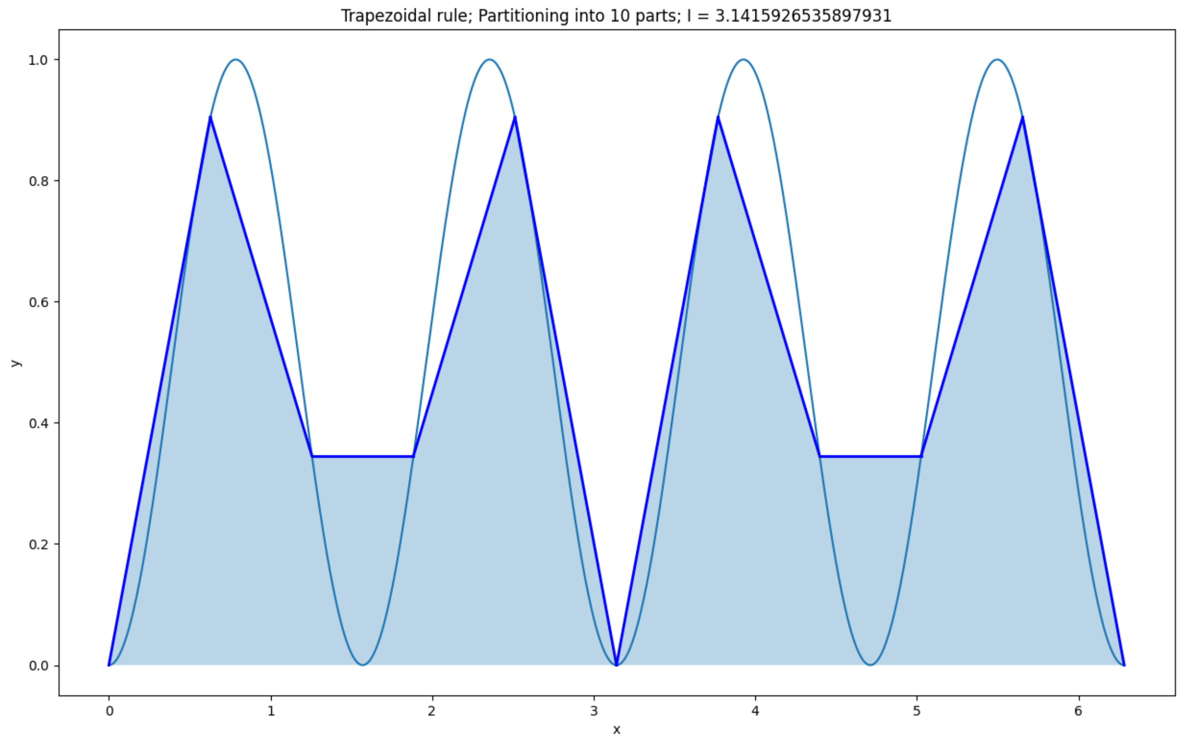
	Левое	Среднее	Правое	Случайное	Трапециальное
n = 10	3.1415926535897936	3.1415926535897931	3.1415926535897931	3.0947247353953662	3.1415926535897931
n = 25	3.1415926535897936	3.1415926535897931	3.1415926535897936	3.1835945210431440	3.1415926535897931
n = 50	3.1415926535897931	3.1415926535897936	3.1415926535897931	3.1655557417203481	3.1415926535897931
n = 100	3.1415926535897936	3.1415926535897936	3.1415926535897931	3.1596776645776359	3.1415926535897931
n = 1000	3.1415926535897931	3.1415926535897931	3.1415926535897931	3.1410972015278227	3.1415926535897936

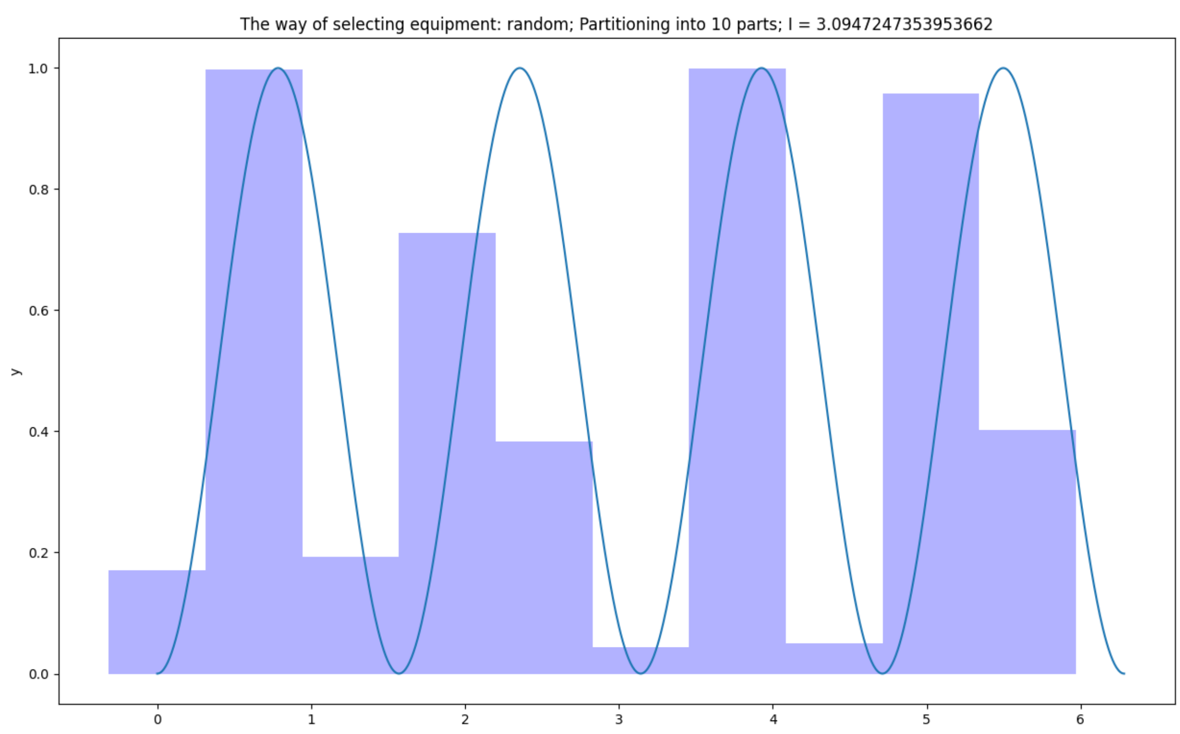
Анализ результатов таблицы

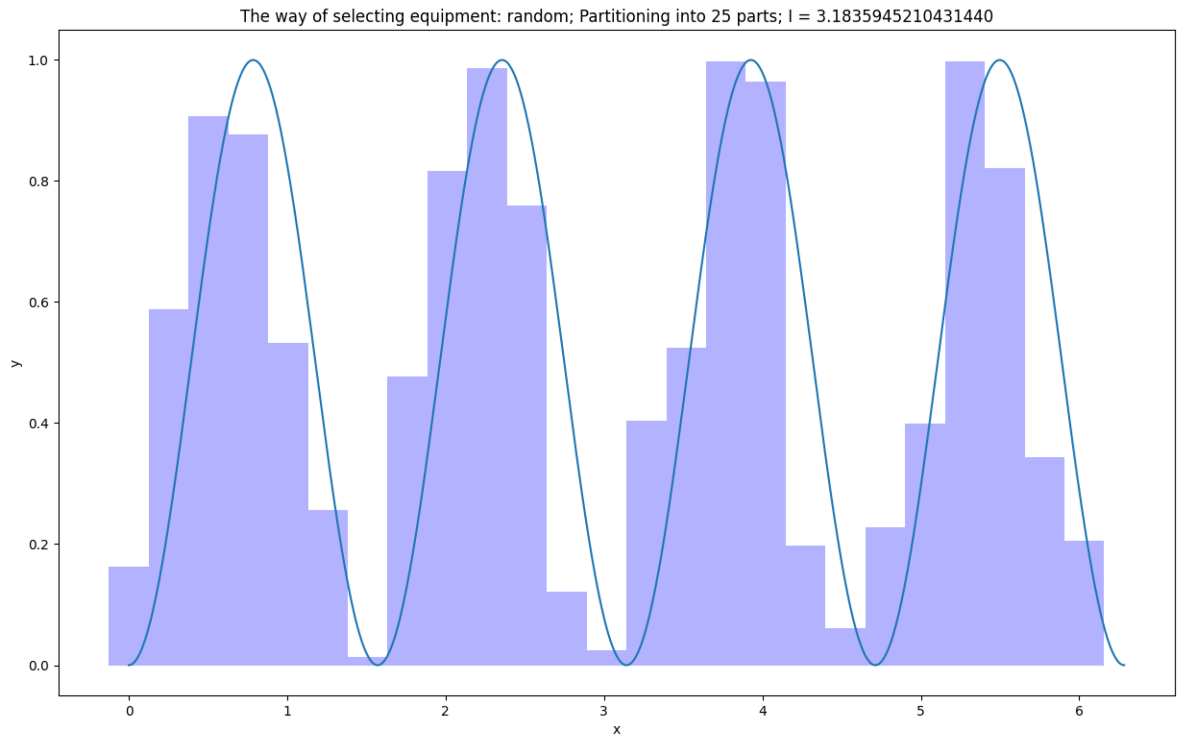
В аналитической части получили, что определенный интеграл $\int\limits_0^{2\pi} \sin^2(2x)dx = \pi \approx 3.14159265358979311600$

Погрешность измерения для всех видов выбора $\leq 10^{-15}$, кроме случайного. Для случайного погрешность $\leq 5 \cdot 10^{-2}$.
По таблице заметно, что такие варианты выбора оснащения, как левое, среднее и правое дают нам практически точное значение. При выборе случайного оснащения значение интегральной суммы приближается к числу π при измельчении разбиения. Приближенное вычисление интеграла методом трапеций даёт нам значения приближенные к числу π.

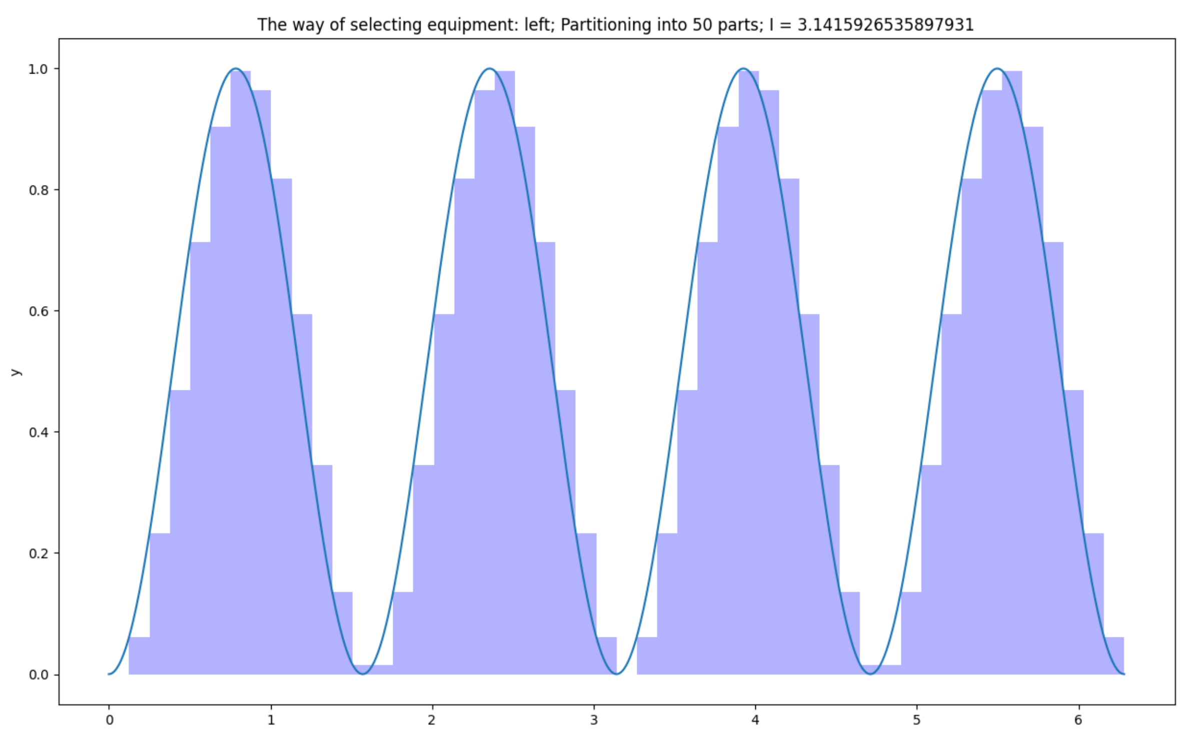
Графики


Trapezoidal rule; Partitioning into 10 parts; I = 3.1415926535897931
y
x
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
3
4
5
6

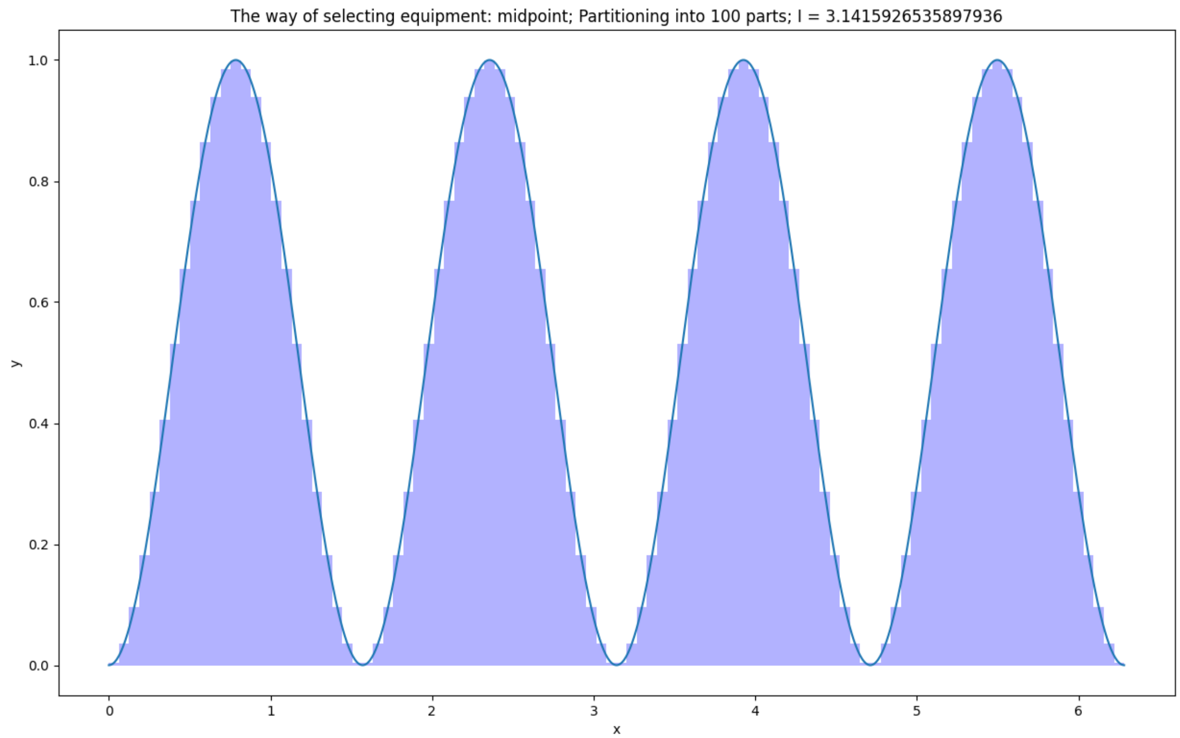

The way of selecting equipment: random; Partitioning into 10 parts; I = 3.0947247353953662
y
x
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
3
4
5
6



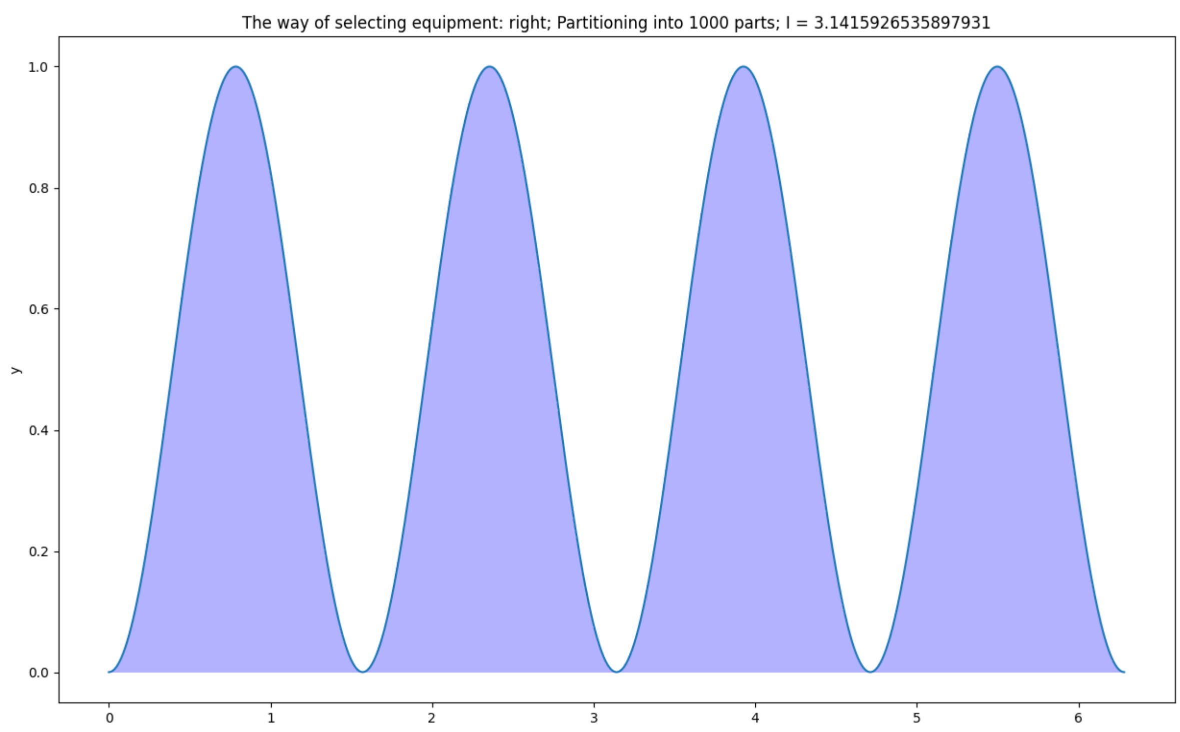
The way of selecting equipment: random; Partitioning into 25 parts; I = 3.1835945210431440
y
x
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
3
4
5
6



The way of selecting equipment: left; Partitioning into 50 parts; I = 3.1415926535897931
y
x
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
3
4
5
6



The way of selecting equipment: midpoint; Partitioning into 100 parts; I = 3.1415926535897936
y
x



The way of selecting equipment: right; Partitioning into 1000 parts; I = 3.1415926535897931
y
x

Описание программы и функций

Описание

В начале программы прописаны данные по умолчанию (f, a, b, n), которые можно изменять.
f: функция, которую нужно интегрировать
a: левая граница отрезка
b: правая граница отрезка
n: число разбиений
При запуске по умолчанию происходит выполнение функции *main*.

Функция plot_integral

"""
@*param* f: функция, которую нужно интегрировать
@*param* a: левая граница отрезка
@*param* b: правая граница отрезка
@*param* n: число разбиений
@*param* *method*: метод, который нужно использовать
@return I: значение интегральной суммы
Вычисляет интегральную сумму функции f на отрезке $[a; b]$ при заданном числе n разбиений с помощью одного из методов $(left, right, midpoint, random, trapezoidal)$. Строит график функции f на отрезке $[a; b]$ и добавляет на этот график прямоугольники, которые соответствуют выбранной интегральной сумме.
"""

Функция main

"""
Используется при запуске программы для вывода подсчётов реализованными методами при помощи ранее реализованной функции *plot_integral*.
"""

Исходный код программы

```
#                                                              Python(3.11)

import math
import numpy as np
import matplotlib.pyplot as plt

f = lambda x: np.sin(2 * x) ** 2
a, b = [0, 2 * math.pi]
n = 1000


MODE = "SHOW"
MODE = "SAVE"

def plot_integral(f, a, b, n, method='left'):
    """
    @param f: function to be integrated
    @param a: left endpoint of interval
    @param b: right endpoint of interval
    @param n: number of partitions
    @param method: method to be used
    @return I: value of the integral sum
    Calculates the integral sum of the function f on the interval [a, b]
```

```
    with a specified number of subdivisions n
    using one of the methods (left, right, midpoint, random, trapezoidal).
    Plots the graph of function f on the interval [a, b] and adds rectangles
    corresponding to the selected integral sum to this graph.
    """
    # adding function graphic
    x = np.linspace(a, b, 1000)
    y = f(x)
    plt.plot(x, y)
    plt.gcf().set_size_inches(15, 9)
    length_of_segment = (b - a) / n
    x_points = np.linspace(a, b, n + 1)
    if method == 'left':
        x_points = x_points[:-1]
        y_points = f(x_points)
        for i in range(n):
            rect = plt.Rectangle((x_points[i], 0), length_of_segment, y_points[i],
                                 facecolor='blue', alpha=0.3)
            plt.gca().add_patch(rect)
    elif method == 'right':
        x_points = x_points[1:]
        y_points = f(x_points)
        for i in range(n):
            rect = plt.Rectangle((x_points[i] - length_of_segment, 0),
                                   length_of_segment, y_points[i],
                                     facecolor='blue', alpha=0.3)
            plt.gca().add_patch(rect)
    elif method == 'midpoint':
        x_points = x_points[:-1] + length_of_segment / 2
        y_points = f(x_points)
        for i in range(n):
            rect = plt.Rectangle((x_points[i] - length_of_segment / 2, 0),
                                   length_of_segment, y_points[i],
                                     facecolor='blue', alpha=0.3)
            plt.gca().add_patch(rect)
    elif method == 'random':
        xx_points = x_points[:-1] + np.random.uniform(0, length_of_segment, size=n)
        x_points = x_points[:-1] + length_of_segment / 2
        y_points = f(xx_points)
        for i in range(n):
            rect = plt.Rectangle((x_points[i] - length_of_segment, 0),
                                   length_of_segment, y_points[i],
                                     facecolor='blue', alpha=0.3)
            plt.gca().add_patch(rect)
    elif method == 'trapezoidal':
        x_points = np.linspace(a, b, n + 1)
        y_points = f(x_points)
        for i in range(n):
            plt.plot([x_points[i], x_points[i + 1]], [y_points[i],
                                   y_points[i + 1]], 'b-', lw=2)
        plt.fill_between(x_points, y_points, alpha=0.3)
    else:
        raise ValueError('Unknown method: {}'.format(method))
    if method != 'trapezoidal':
        I = np.sum(y_points) * length_of_segment
        plt.title(f'The way of selecting equipment: {method}; ' +
```

```python
                f'Partitioning into {n} parts; I = {I:.6f}')
    else:
        I = np.sum((y_points[1:] + y_points[:-1]) * length_of_segment / 2)
        plt.title(f'Trapezoidal rule; Partitioning into {n} parts; I = {I:.6f}')
    plt.xlabel('x')
    plt.ylabel('y')
    if MODE == "SAVE":
        plt.savefig(f"{n}_{method}.png")
    if MODE == "SHOW":
        plt.show()
    plt.close()
    return I


def main():
    print(f"Integral of (sin(2x))^2 from {a} to {b} with {n} "
          + " segments using different methods:")
    for method in ['left', 'right', 'midpoint', 'random', 'trapezoidal']:
        I = plot_integral(f, a, b, n, method=method)
        print(f'{method}: {I:.6f}')


if __name__ == '__main__':
    main()
```