ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Дисциплина:
“Математический анализ”

Отчёт по лабораторной работе

«Интеграл Римана»

Варианты: 11, 16, 28

Выполнили:

Морозова Полина Сергеевна, студент группы N3151
Пресняков Дмитрий Станиславович, студент группы N3148
Проскуряков Роман Владимирович, студент группы K3139

Проверил(а):

Санкт-Петербург
2024 г.

Аналитическая часть

Для доказательства существования интеграла Римана нужно доказать:

1. $f(x)$ — ограниченная функция
2. $\lim\limits_{\lambda(\tau)\to 0} \sigma_\tau(f,\xi) \in \mathbb{R}$

Вариант №11

Вариант 11

$f(x) = \sin x \quad [0, 2\pi] \qquad |f(x)| \le 1 \Rightarrow f(x)$ — огран

$\tau = \left\{\frac{2\pi}{n}k\right\}_{k=0}^{n} \qquad \xi_k = \frac{2\pi}{n}k$

$\lambda(\tau) = \frac{2\pi}{n}, \quad \lambda(\tau) \xrightarrow[n\to+\infty]{} 0$

$$\lim_{\lambda(\tau)\to 0} \sigma(\sin x, \tau, \xi) = \lim_{n\to+\infty} \sum_{k=0}^{n-1} \sin\left(\frac{2\pi}{n}k\right)\cdot\frac{2\pi}{n} =$$

$$= \lim_{n\to+\infty} \frac{2\pi}{n} \sum_{k=0}^{n-1} \sin\frac{2\pi}{n}k$$

$$\sin\frac{2\pi}{n}k = -\sin\left(-\frac{2\pi}{n}k\right) = -\sin\left(2\pi - \frac{2\pi}{n}k\right) =$$

$$= -\sin\left(\frac{2\pi}{n}(n-k)\right)$$

$$\sum_{k=0}^{n-1} \sin\frac{2\pi}{n}k = \sin\frac{2\pi 0}{n} + \sin\frac{2\pi 1}{n} + \ldots + \sin\frac{2\pi}{n}\left(\left[\frac{n-1}{2}\right]-1\right) +$$

$$+ \sin\frac{2\pi}{n}\left[\frac{n-1}{2}\right] + \sin\frac{2\pi}{n}\left(\left[\frac{n-1}{2}\right]+1\right) + \ldots + \sin\frac{2\pi}{n}(n-1) =$$

$$= \sin\frac{2\pi 0}{n} + \sin\frac{2\pi 1}{n} + \ldots + \sin\frac{2\pi}{n}\left(\left[\frac{n-1}{2}\right]-1\right) + \sin\frac{2\pi}{n}\left[\frac{n-1}{2}\right] -$$

$$- \sin\frac{2\pi}{n}\left(n - \left(\left[\frac{n-1}{2}\right]+1\right)\right) - \ldots - \sin\frac{2\pi}{n}\left(n-(n-1)\right) =$$

$$= \left[\begin{array}{l} n - \text{чёт}; \left[\frac{n-1}{2}\right] = \frac{n-2}{2} \\ n - \text{нечёт}; \left[\frac{n-1}{2}\right] = \frac{n-1}{2} \end{array}\right] = \begin{cases} \sin\left(\frac{2\pi}{n}\cdot 0\right) - \sin\left(\frac{2\pi}{n}\left(\frac{n}{2}\right)\right), & n - \text{чёт} \\ \sin\left(\frac{2\pi}{n}\cdot 0\right), & n - \text{нечёт} \end{cases} =$$

$$= \begin{cases} 0, & n - \text{чёт} \\ 0, & n - \text{нечёт} \end{cases} = 0. \qquad \lim_{n\to+\infty} \frac{2\pi}{n} \sum_{k=0}^{n-1} \sin\frac{2\pi}{n}k = 0\cdot 0 = 0$$

Значит интеграл Римана $\exists$ и равен 0.

Проверим по формуле Ньютона-Лейбница

$$\int_0^{2\pi} \sin x\,dx = -\cos x\Big|_0^{2\pi} = -\cos 2\pi + \cos 0 = -1+1 = 0$$

Вариант №16

$f(x) = e^{-x}$ на $[0,2]$

1) На $[0,2]$ $f(x) \in (0,1] \Rightarrow e^{-x}$ – ограниченная ф-ия на отрезке

2) Положим разбиение $\tau = \left\{\frac{2}{n}\cdot k\right\}_{k=0}^{n}$ и оснащение $\xi_k = \frac{2}{n}\cdot k$, $k = 0 \ldots n-1$

Тогда $\lambda(\tau) = \frac{2}{n}$, т.е. $\lambda(\tau) \xrightarrow[n\to+\infty]{} 0$.

$$\lim_{\lambda(\tau)\to 0} \sigma_\tau(f,\xi) = \lim_{n\to+\infty} \sum_{k=0}^{n-1} e^{-\left(\frac{2}{n}\cdot k\right)} \cdot \frac{2}{n} = \lim_{n\to+\infty} \frac{2}{n} \cdot \frac{e^0\left(1 - e^{-\frac{2}{n}\cdot n}\right)}{1 - e^{-\frac{2}{n}}} =$$

$$= \lim_{n\to+\infty} \frac{2}{n} \cdot \frac{1-e^{-2}}{1-e^{-\frac{2}{n}}} = \left/ \begin{matrix} m = -\frac{2}{n} \\ m \to 0 \end{matrix} \right/ = \lim_{m\to 0}\left(1 - \frac{1}{e^2}\right)\frac{-m}{1-e^m} \sim -m =$$

$$= \lim_{m\to 0}\left(1 - \frac{1}{e^2}\right)\frac{\cancel{-m}}{\cancel{-m}} = \underline{1 - \frac{1}{e^2}} \in \mathbb{R}$$

Найдём интеграл по формуле Ньютона-Лейбница:

$$\int_0^2 e^{-x}\,dx = -e^{-x}\Big|_0^2 = -e^{-2} + e^0 = \underline{1 - \frac{1}{e^2}}$$

Вариант №28

$f(x) = \cos 2x$ на $[0,\pi]$

1) $|\cos 2x| \le 1 \Rightarrow \cos 2x$ ограниченная

2) Положим разбиение $\tau = \left\{\frac{\pi}{n}\cdot k\right\}_{k=0}^{n}$ и оснащение $\xi_k = \frac{\pi}{n}\cdot k$ $k = 0 \ldots n-1$. Тогда $\lambda(\tau) = \frac{\pi}{n}$, т.е. $\lambda(\tau) \xrightarrow[n\to+\infty]{} 0$.

$$\lim_{\lambda(\tau)\to 0} \sigma_\tau(f,\xi) = \lim_{n\to+\infty} \sum_{k=0}^{n-1} \cos\left(2\cdot\frac{\pi}{n}\cdot k\right)\cdot\frac{\pi}{n} = \lim_{n\to+\infty}\frac{\pi}{n}\cdot\sum_{k=0}^{n-1}\cos\left(\frac{2\pi k}{n}\right) =$$

$$= \lim_{n\to+\infty}\frac{\pi}{n}\cdot 0 = 0$$

$$\cdot \sum_{k=0}^{n-1}\cos\left(\frac{2\pi k}{n}\right) = \sum_{k=0}^{n-1}\left(\frac{e^{i\cdot\frac{2\pi k}{n}} + e^{-i\frac{2\pi k}{n}}}{2}\right) = \frac{1}{2}\sum_{k=0}^{n-1} e^{i\frac{2\pi k}{n}} + \frac{1}{2}\sum_{k=0}^{n-1} e^{i\frac{2\pi k}{n}} \ominus$$

По формуле суммы геометрической прогрессии

$$\ominus \frac{1}{2}\cdot\frac{e^{i\cdot\frac{2\pi n}{n}} - e^{i\frac{2\pi\cdot 0}{n}}}{e^{i\frac{2\pi}{n}} - 1} + \frac{1}{2}\cdot\frac{e^{-i\frac{2\pi n}{n}} - e^{-i\frac{2\pi\cdot 0}{n}}}{e^{i\frac{2\pi}{n}} - 1} =$$

$$= \frac{1}{2}\,\frac{\cos(2\pi) + i\sin(2\pi) - e^0}{e^{i\frac{2\pi}{n}} - 1} + \frac{1}{2}\cdot\frac{\cos(-2\pi) + i\sin(-2\pi) - e^0}{e^{i\frac{2\pi}{n}} - 1} =$$

$$= \frac{1}{2}\cdot\frac{1+0-1}{e^{i\frac{2\pi}{n}} - 1} + \frac{1}{2}\cdot\frac{1+0-1}{e^{i\frac{2\pi}{n}} - 1} = 0 + 0 = 0$$ – при любом n

Найдём интеграл по формуле Ньютана-Лейбница

$$\int_0^\pi \cos 2x\,dx = \int_0^\pi \frac{1}{2}\cos 2x\,d(2x) = \frac{1}{2}\Big[\sin 2x\Big]_0^\pi = \frac{1}{2}(0-0) = 0$$

Результаты работы программы

После запуска программы нас просят ввести число разбиений и способ оснащения: левый, правый, средний или случайный.

Вариант №11

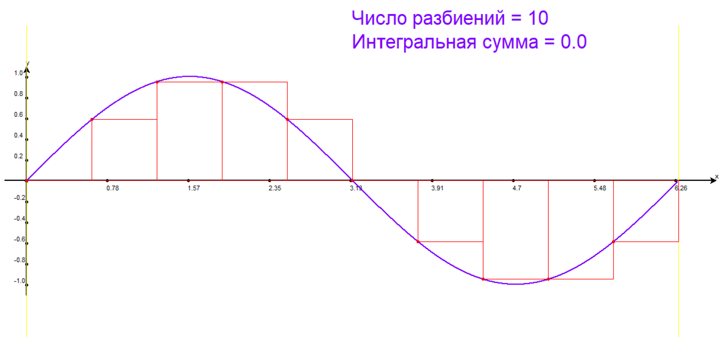

Рис. 4. Левое оснащение

Рис. 5. Среднее оснащение

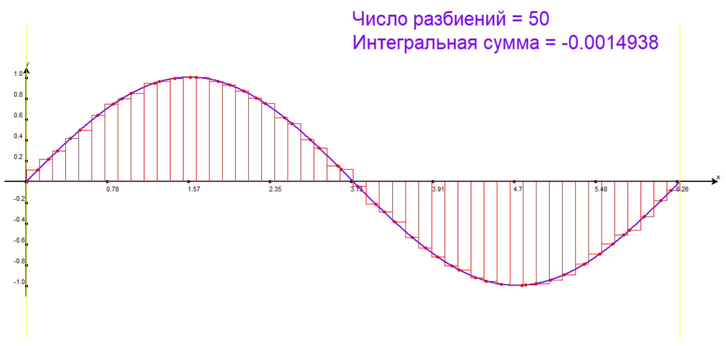

Рис. 6. Случайное оснащение

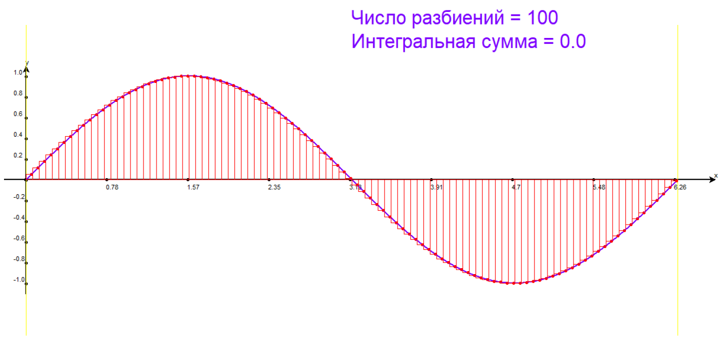

Рис. 7. Правое оснащение

Вариант №16

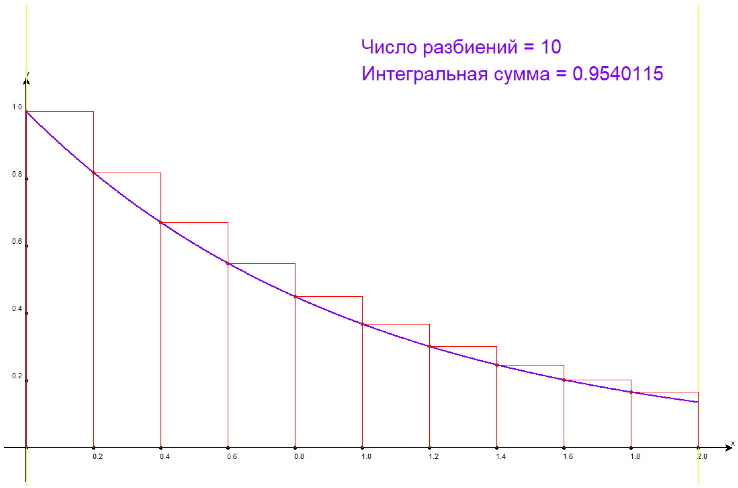

Рис. 8. Левое оснащение

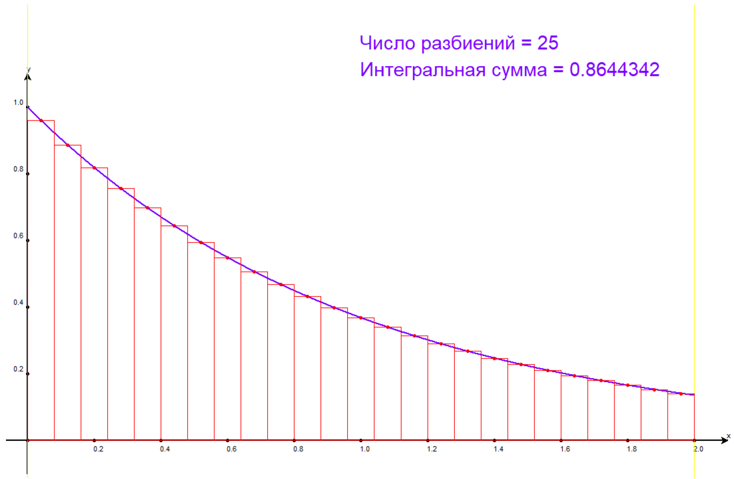

Рис. 9. Среднее оснащение

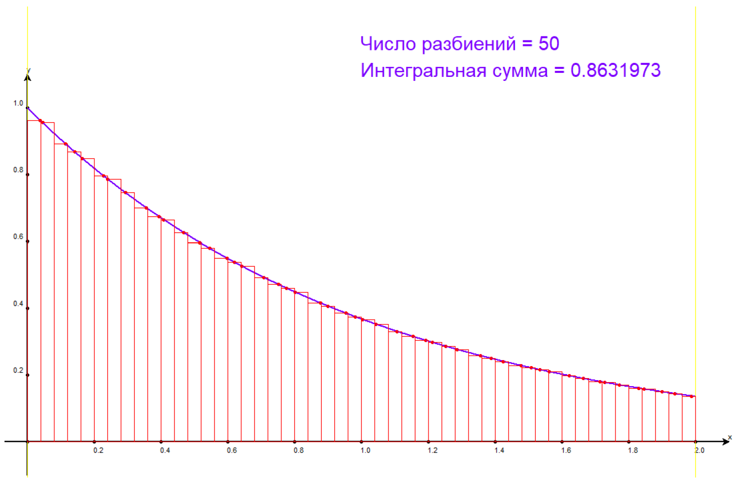

Рис. 10. Случайное оснащение

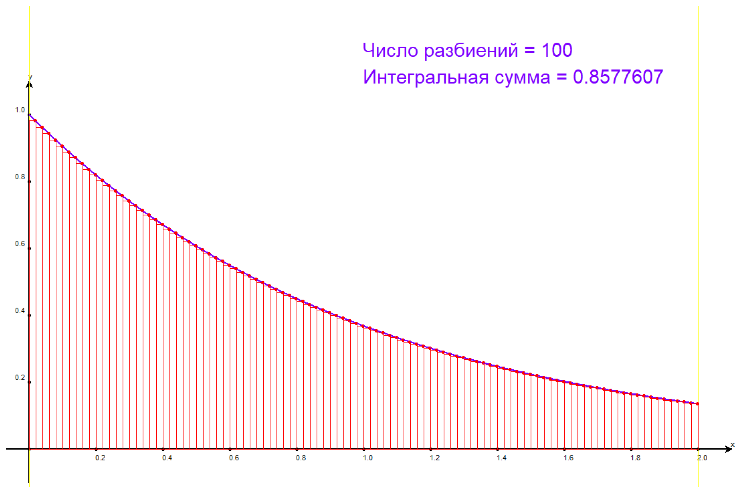

Рис. 11. Правое оснащение

Вариант №28

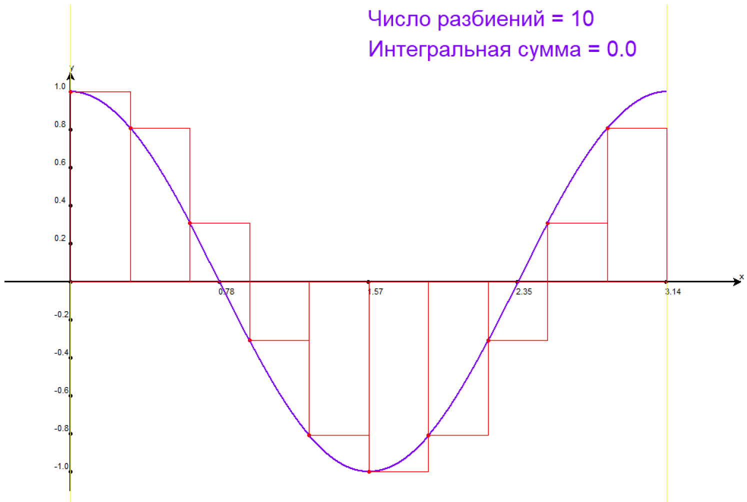

Рис. 12. Левое оснащение

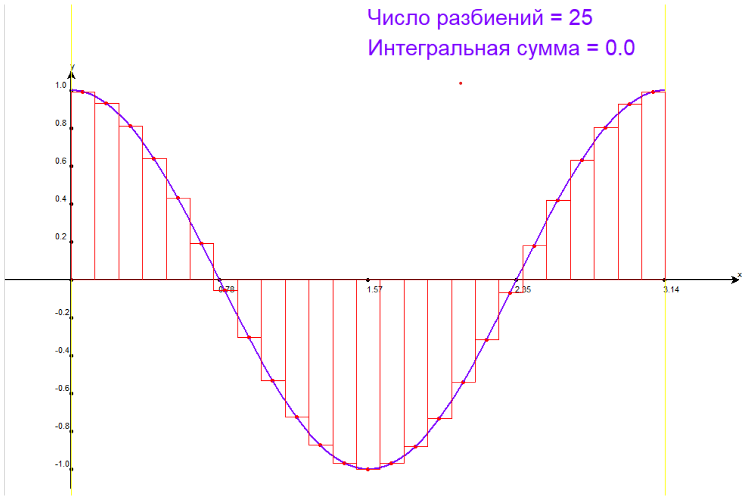

Рис. 13. Случайное оснащение

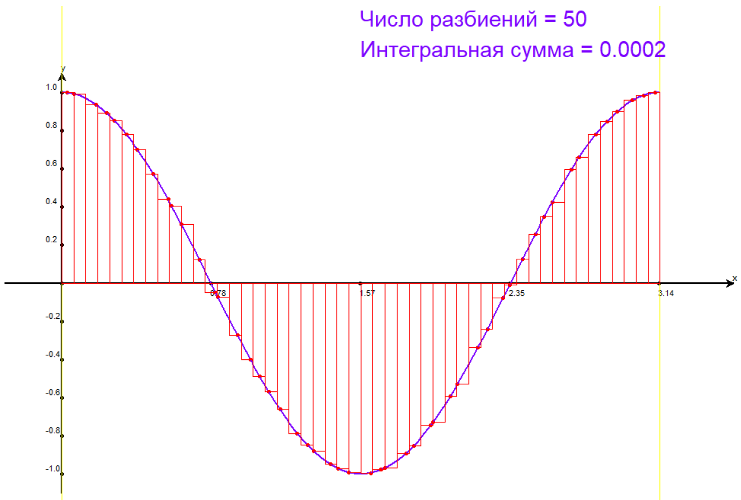

Рис. 14. Случайное оснащение

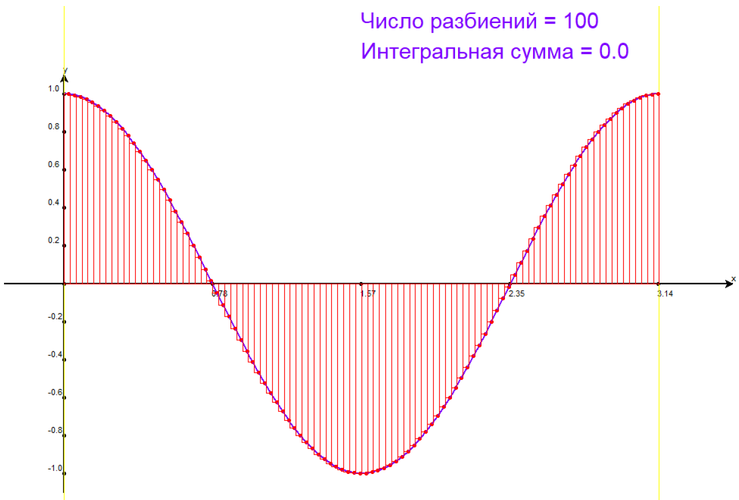

Рис. 15. Правое оснащение

Текст программы

```
import turtle
import random
from math import *

def sign(n):
  return 1 if n >= 0 else -1

def myRound(f, n = 0):
  return int(f * 10**n + 0.5 * sign(f)) / 10**n

def readInputData():
  global divisions
  global getIndexRectHeight
  divisions = int(input("Введите число точек разбиения\n"))

  global rectWidth
  rectWidth = (finX - startX) / divisions   # ширина одного разбиения

  while True:
    c = input('''Выберите способ оснащения:\n\t
    L – левый\n\tM – средний\n\tR – правый\n\tA - случайный\n''')
    match c:
      case 'l' | 'L':
          getIndexRectHeight = lambda : 0
          break
      case 'm' | 'M':
          getIndexRectHeight = lambda : int(rectWidth * zoom) // 2
          break
      case 'r' | 'R':
          getIndexRectHeight = lambda : int(rectWidth * zoom) - 1
          break
      case 'a' | 'A':
          getIndexRectHeight = lambda : random.randint(0, int(rectWidth * zoom) - 1)
          break
      case _:
          print("Не найденно соответствие!\n==============\n

def F(x):                   # наша прекрасная функция
  return sin(x)             # вар 11
  #return e**-x             # вар 16
  #return cos(2 * x)        # вар 28

singleSegmentX = pi / 4     # единичный отрезок по X
singleSegmentY = 0.2        # единичный отрезок по Y
startX = 0                  # начало ООФ
finX = 2 * pi               # конец ООФ вар 11
#finX = 2                   # конец ООФ вар 16
#finX = pi                  # конец ООФ вар 28
startY = -1                 # начало ОДЗ
finY = 1                    # конец ОДЗ

zoom = 175                  # увеличить изображение в zoom раз
```

```python
cordOriginX = -600          # сместить график в окне по X
cordOriginY = 0             # сместить график в окне по Y
colorChart = "#7f00ff"      # цвет графика
colorRect = "#ff0000"       # цвет прямоугольничков
colorStopLine = "#ffff00"   # цвет линий, ограничивающий ООФ

def teleport(T, x, y):
  T.pu()
  T.goto(cordOriginX + x, cordOriginY + y)
  T.pd()

def drawAxes(x_lt, y_lt, x_rb, y_rb, x_step = 1, y_step = 1):

  def drawAxe(T, length, step, fromNumber, shift):
    T.fd(-step // 2)
    T.fd(step // 2)
    for i in range(0, length // step + 1):
      T.dot()
      if(abs(fromNumber + i * step / zoom) > 0.01):
        T.pu()
        T.left(90)
        T.fd(shift)
        T.pd()
        T.write(myRound(fromNumber + i * step / zoom, 2))
        T.pu()
        T.fd(-shift)
        T.right(90)
        T.pd()
      if(i == length // step):
        T.fd(step // 2)
      else:
        T.fd(step)

  X = turtle.Turtle()
  X.speed(0)
  X.pensize(2)
  teleport(X, x_lt, 0)
  drawAxe(X, x_rb - x_lt, x_step, startX, -20)
  X.write("x")

  Y = turtle.Turtle()
  Y.speed(0)
  Y.pensize(2)
  teleport(Y, 0, y_rb)
  Y.left(90)
  drawAxe(Y, y_lt - y_rb, y_step, startY, 20)
  Y.write("y")

def drawRect(x_lt, y_lt, x_rb, y_rb):
  T = turtle.Turtle()
  T.hideturtle()
  T.speed(0)
  T.color("#ff0000")
  teleport(T, x_lt, y_lt)
  for i in range(2):
    T.forward(x_rb - x_lt)
```

```
    T.right(90)
    T.forward(y_lt - y_rb)
    T.right(90)

readInputData()

drawAxes(int(startX * zoom), int(finY * zoom), int(finX * zoom), int(startY * zoom),
int(singleSegmentX * zoom), int(singleSegmentY * zoom))

T = turtle.Turtle()
T.speed(0)
T.hideturtle()

T.color(colorStopLine)
teleport(T, startX * zoom, startY * zoom - zoom / 2)
T.goto(startX * zoom + cordOriginX, finY * zoom + cordOriginY + zoom / 2)
teleport(T, finX * zoom, startY * zoom - zoom / 2)
T.goto(finX * zoom + cordOriginX, finY * zoom + cordOriginY + zoom / 2)

T.color(colorChart)
T.pensize(2)
teleport(T, startX * zoom, F(startX) * zoom)

integralSum = 0.0

rectHeight = 0
for n in range(0, divisions):
  indexRectHeight = getIndexRectHeight()
  rectStart = startX + rectWidth * n
  for i in range(0, int((rectWidth) * zoom)):
    x = rectStart + i / zoom
    f = F(x)
    T.goto(x  * zoom + cordOriginX, f * zoom + cordOriginY)
    if(i == indexRectHeight):
      rectHeight = f
      T.color(colorRect)
      T.dot()
      T.color(colorChart)
      drawRect(rectStart * zoom, rectHeight * zoom, (rectStart + rectWidth) * zoom, 0)
      integralSum += rectHeight * rectWidth

teleport(T, (startX + finX) / 2 * zoom, finY * zoom + 40)
T.write("Интегральная сумма = " + str(myRound(integralSum, 7)), font = ('Arial', 25, 'normal'))
teleport(T, (startX + finX) / 2 * zoom, finY * zoom + 70)
T.write("Число разбиений = " + str(divisions), font = ('Arial', 25, 'normal'))

print("\nИнтегральная сумма =", myRound(integralSum, 7), "\n")

turtle.exitonclick()
#turtle.Screen().mainloop()
```