ИДЗ 2

Исполнитель: Чапурина Валерия Сергеевна

Группа: БПИ237

Вариант: 28

Условие задания

> Разработать программы на языке Ассемблера процесса RISC-V, с использованием команд арифметического сопроцессора, выполняемые в симуляторе RARS. Разработанные программы должны принимать числа в допустимом диапазоне. Например, нужно учитывать области определения и допустимых значений, если это связано с условием задачи.

Разработать программу численного интегрирования функции $y = a + b \cdot x^4$
(задаётся действительными числами a, b) в определённом
диапазоне целых (задаётся так же)
методом прямоугольников
с избытком
(точность вычислений = 0.0001).

Текст программы

(на Питоне)

```
def y(a, b, x) -> float:
    return a + b * x ** 4


def main():
    precision = 0.0001
    a = float(input("Введите действительное число a: "))
    b = float(input("Введите действительное число b: "))
    begin = float(input("Введите действительное число begin - начало отрезка: "))
```

```
    end = float(input(f"Введите действительное число end - конец отрезка (не меньше begin + {pre

    if begin + precision > end:
        print(f"begin + {precision} > end, неверно введённые данные")
        return

    current = begin
    summa = 0.0
    while current + precision <= end:
        summa += precision * max(y(a, b, current), y(a, b, current + precision))
        current += precision
    print(f"Интеграл равен: {summa}")
    return


main()
```

На ассемблере:

```
.include "macrolib.s"
.data
    sum:            .double    0.0      # Итоговая сумма интеграла
    precision:      .double    0.0001   # Точность
.text

# считывает вещественное число с клавиатуры и записывает его в регистр %x
.macro read_double(%x)
    li a7, 7          # Системный вызов для чтения double
        ecall
        fmv.d %x, fa0       # Копируем значение из f0 в регистр %x
.end_macro

# печатает число типа double из регистра %x
.macro print_double(%x)
    li a7, 3
        fmv.d fa0, %x      # Копируем значение из регистра %x в f12 для печати
        ecall
.end_macro

# сохраняет в регистр %z максимум из двух double'ов в регистрах %x и %y
.macro max(%x, %y, %z)
    flt.d t0, %x, %y      # Сравниваем %x и %y. t0 = 1, if %x < %y
        fmv.d %z, %x          # Если t0 = 0, тозаписываем %x в %z
        beqz t0, end         # Если %x >= %y (t0 = 0), пропускаем следующую инструкцию
        fmv.d %z, %y          # Если %x < %y, записываем %x в %z
```

```
end:                    # Метка конца макроса
.end_macro

# считает значение y = %a + %b * (%x)⁴ и записывает в %y (all are double's)
.macro calc_y(%a, %b, %x, %y)
        fmul.d ft0, %x, %x      # ft0 = %x * %x
        fmul.d ft1, ft0, ft0    # ft1 = (%x)² * (%x)² = (%x)⁴
        fmul.d ft2, %b, ft1     # ft2 = %b * (%x)⁴
        fadd.d %y, %a, ft2      # %y = %a + %b * (%x)⁴
.end_macro


main:
    print_str("Введите действительное число a: ")
    read_double(fs1)

    print_str("Введите действительное число b: ")
    read_double(fs2)

    print_str("Введите действительное число begin - начало отрезка: ")
    read_double(fs3)

    print_str("Введите действительное число end - конец отрезка (не меньше begin + 0.0001): ")
    read_double(fs4)

    # просто для красоты
    print_str("-------------------------------------------------------------------------------\n")

    # проверим, выполнено ли begin + 0.0001 <= end
    fld     fs5 precision t0    # Загружаем precision в fs5
    fadd.d   ft2 fs3 fs5    # ft2 = begin + precision
    fge.d    t1 fs4 ft2     # t1 = 1 if end >= begin + precision, else t0 = 0

    # Если end < begin + precision
    beqz t1 incorrect_segment

    fld fs10 sum t0          # Загружаем sum в fs10
    j loop

incorrect_segment:
    print_str("begin + 0.0001 > end, неверно введённые данные")
    exit

loop:
    # проверим, выполнено ли current + 0.0001 <= end
```

```
    fadd.d   ft2 fs3 fs5     # ft2 = current + precision
    fge.d    t1 fs4 ft2      # t1 = 1 if end >= current + precision, else t0 = 0

    # Если end < current + precision
    beqz t1 print_result

    # Считаем значения функции на концах отрезка прямоугольника
    calc_y(fs1, fs2, fs3, fs6) # fs6 = a + b * current⁴
    fadd.d   ft3 fs3 fs5     # ft3 = current + precision
    calc_y(fs1, fs2, ft3, fs7)  # fs7 = a + b * (current + precision)⁴

    # Площадь большего прямоугольника
    max(fs6, fs7, fs8)       # fs8 - наибольшое значение
    fmul.d fs9 fs8 fs5          # площадь fs9 = fs8 * precision

  # Обновление суммы
  fadd.d fs10, fs10, fs9
    j next_iter

next_iter:
    fadd.d   fs3 fs3 fs5        # fs3 = current + precision
    j loop

print_result:
    print_str("Интеграл равен: ")
    print_double(fs10)
    print_str("\n")
    j exit

exit:
    li a7, 10
    ecall
```

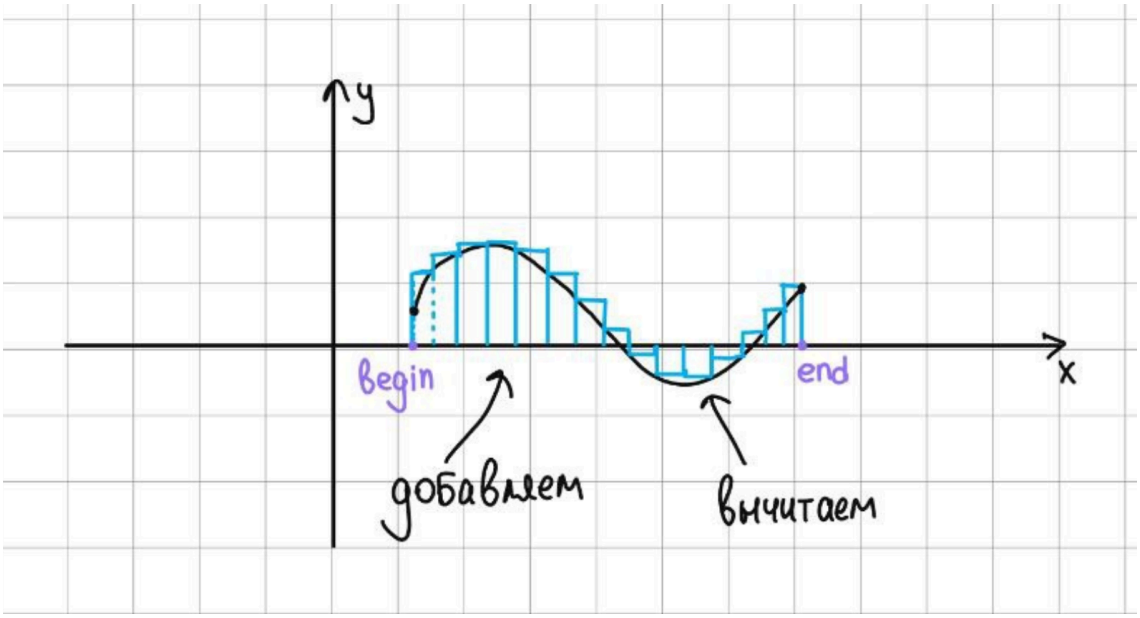
Некоторые пояснения к программе

Идея решения

→ теоретический материал можно посмотреть тут (4 слайд)

так как нам нужна точность `precision = 0.0001`, то разобьём (начиная с начала отрезка `begin`) изначальный отрезок на подотрезки длины precision и будем рассматривать прямоугольники на них. Посчитаем значения функции в концах подоотрезков и выберем наибольшее из них (так как

просили считать с избытком). Найдём площади таких прямоугольников и просуммируем — это и будет определённым интегралом



Неочевидные моменты

- если пользователь ввёл отрезок [begin, end] такой, что begin + precision < end, то считаем, что он ошибся
- при решении поставленной задачи я подумала, что точность вычислений - насколько может отличаться полученное значение интеграла метод прямоугольников с избытком от настоящего значения интеграла. Поэтому сравнивала ответ программы в RARS со значением интеграла
- возможно, задание подразумевало такое решение: самостоятельно проинтегрировать функцию ($ax + \frac{b}{5}x^5$), найти значение определённого интеграла на отрезке, а потом при помощи бин поиска искать количество подоотрезков, на которое будем разбивать наш рассматриваемый отрезок, считать интеграл методом прямоугольников с избытком и проверять, сходится ли полученное значение с ответом (с погрешностью). Но я о таком подумала уже после написания кода к своей интерпретации
- ещё не очень поняла, что означает “в определённом диапазоне целых (задаётся так же)” - концы отрезка являются целыми числами или действительными.. Решила, что действительными, поскольку если ссылаться на “задаётся так же” то придём в “(задаётся действительными числами a, b)” — тут про действительные говорили. И если разрешить пользователю вводить действительные числа в качестве концов рассматриваемого отрезка, программа будет более функциональной

Переменные

- `a` для задания функции → регистр `fs1`

- `b` для задания функции → регистр `fs2`
- `begin` — начало рассматриваемого отрезка → регистр `fs3` (используем только в начале, потом превращается в `current`)
- `current` — текущая точка на отрезке → регистр `fs3` (изначально равен `begin`)
- `end` — конец рассматриваемого отрезка → регистр `fs4`
- `precision` — точность вычислений → регистр `fs5`
- `sum` — текущая сумма площадей прямоугольников → регистр `fs10`

Тестирование

Как будет проходить тестирование

буду искать правильный ответ при помощи программы на Python (где заранее посчитана формула для нахождения определённого интеграла) и сверять его с ответом в эмуляторе RARS

дополнительно буду смотреть на картинку в Geogebra и полученную ей площадь

Код программы на Python:

```
def y(a, b, x):
    return a * x + b * (x ** 5) / 5


def integral(func, a, b, begin, end) -> float:
    return func(a, b, end) - func(a, b, begin)


def main():
    precision = 0.0001
    a = float(input("Введите действительное число a: "))
    b = float(input("Введите действительное число b: "))
    begin = float(input("Введите действительное число begin - начало отрезка: "))
    end = float(input(f"Введите действительное число end - конец отрезка (не меньше begin + {pre
        print("-" * 100)

    if begin + precision > end:
        print(f"begin + {precision} > end, неверно введённые данные")
        return

    print(f"Интеграл равен: {integral(y, a, b, begin, end)}")
    return
```

```
main()
```

Страница в Geogebra: https://www.geogebra.org/graphing/travcvm7

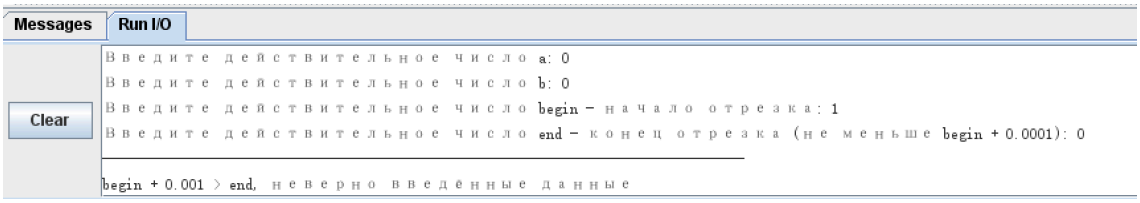
Тест 1 - begin > end, числа целые

Ввод: a = 0, b = 0, begin = 1, end = 0

Предполагаемый вывод: *begin + 0.0001 > end, неверно введённые данные*

Вывод:

```
Введите действительное число a: 0
Введите действительное число b: 0
Введите действительное число begin - начало отрезка: 1
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 0
--------------------------------------------------------------------------------
begin + 0.0001 > end, неверно введённые данные
```



Комментарий к выводу программы

Тут и ещё в нескольких тестах ниже программа в RARS выводит “begin + 0.001 > end, неверно введённые данные” ⟶ тут точность неправильно напечатана, 0.0001 должно быть, это исправлено

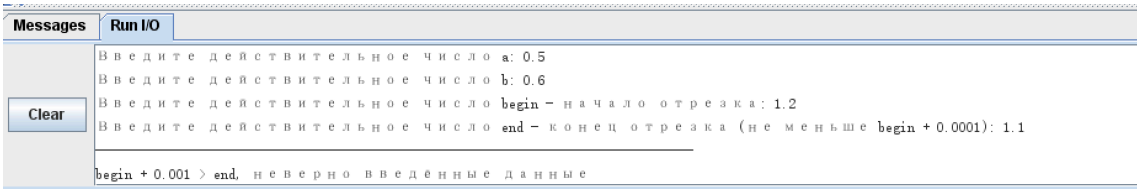
Тест 2 - begin > end, числа дробные

Ввод: a = 0.5, b = 0.6, begin = 1.2, end = 1.1

Предполагаемый вывод: *begin + 0.0001 > end, неверно введённые данные*

Вывод:

```
Введите действительное число a: 0.5
Введите действительное число b: 0.6
Введите действительное число begin - начало отрезка: 1.2
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 1.1
-------------------------------------------------------------------------------------------------
begin + 0.0001 > end, неверно введённые данные
```



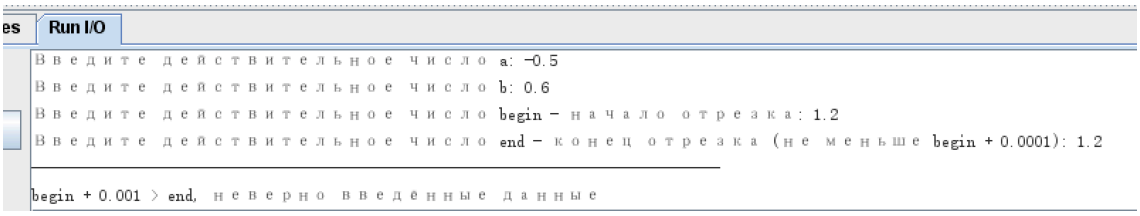
Тест 3 - begin = end, числа дробные

Ввод: a = -0.5, b = 0.6, begin = 1.2, end = 1.2

Предполагаемый вывод: *begin + 0.0001 > end, неверно введённые данные*

Вывод:

```
Введите действительное число a: -0.5
Введите действительное число b: 0.6
Введите действительное число begin - начало отрезка: 1.2
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 1.2
-------------------------------------------------------------------------------------------------
begin + 0.0001 > end, неверно введённые данные
```



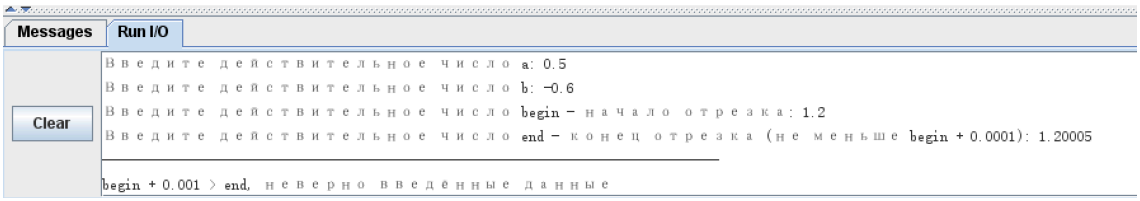
Тест 4 - begin + 0.00005 = end, числа дробные

Ввод: a = 0.5, b = -0.6, begin = 1.2, end = 1.20005

Предполагаемый вывод: *begin + 0.0001 > end, неверно введённые данные*

Вывод:

```
Введите действительное число a: 0.5
Введите действительное число b: -0.6
Введите действительное число begin - начало отрезка: 1.2
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 1.20005
-----------------------------------------------------------------------------------------------------
begin + 0.0001 > end, неверно введённые данные
```



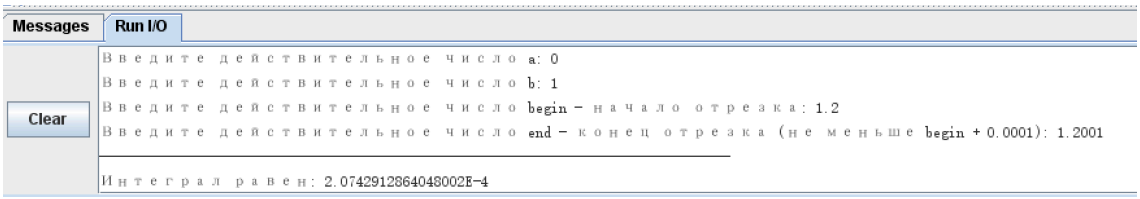
Тест 5 - begin + 0.0001 = end

Ввод: a = 0, b = 1, begin = 1.2, end = 1.2001

Предполагаемый вывод: интеграл посчитался, ответ выводит Python

Вывод:

```
Введите действительное число a: 0
Введите действительное число b: 1
Введите действительное число begin - начало отрезка: 1.2
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 1.2001
-----------------------------------------------------------------------------------------------------
Интеграл равен: 0.00020739456288010327
```



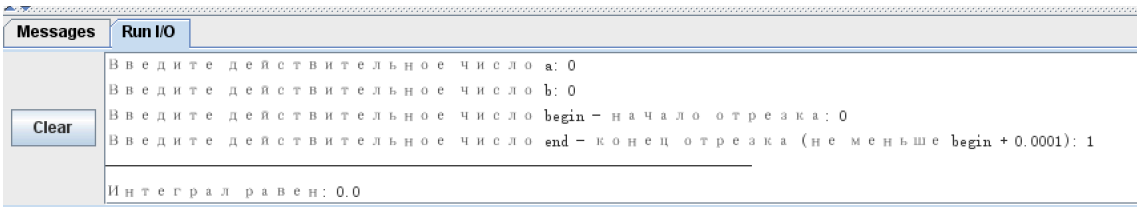
Тест 6 - функция y = 0

Ввод: a = 0, b = 0, begin = 0, end = 1

Предполагаемый вывод: 0

Вывод:

```
Введите действительное число a: 0
Введите действительное число b: 0
Введите действительное число begin - начало отрезка: 0
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 1
-----------------------------------------------------------------------------------------------
Интеграл равен: 0.0
```



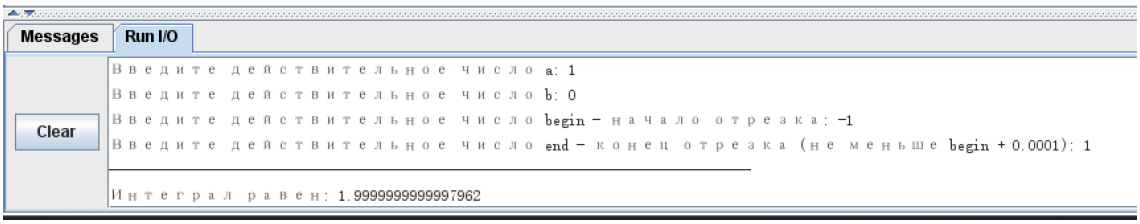
Тест 7 - функция вида $y = a$

Ввод: a = 1, b = 0, begin = -1, end = 1

Предполагаемый вывод: 2

Вывод:

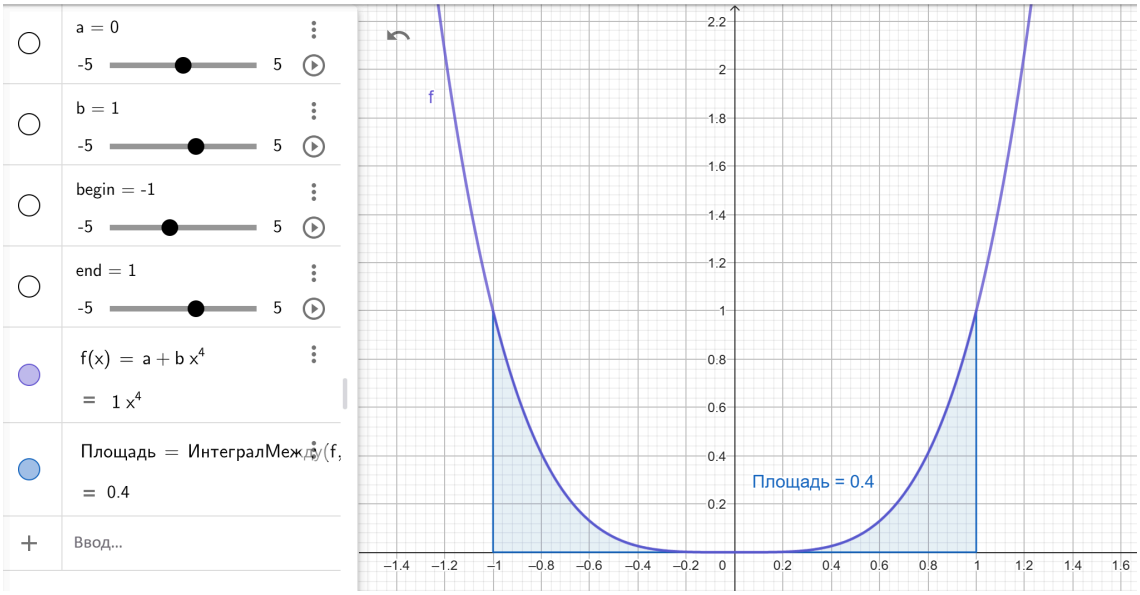
```
Введите действительное число a: 1
Введите действительное число b: 0
Введите действительное число begin - начало отрезка: -1
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 1
-----------------------------------------------------------------------------------------------
Интеграл равен: 2.0
```



Тест 8 - функция вида $y = bx^4$

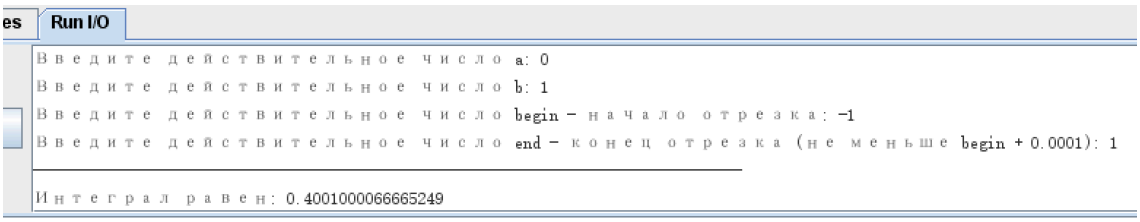
Ввод: a = 0, b = 1, begin = -1, end = 1

Предполагаемый вывод: 0.4



Вывод:

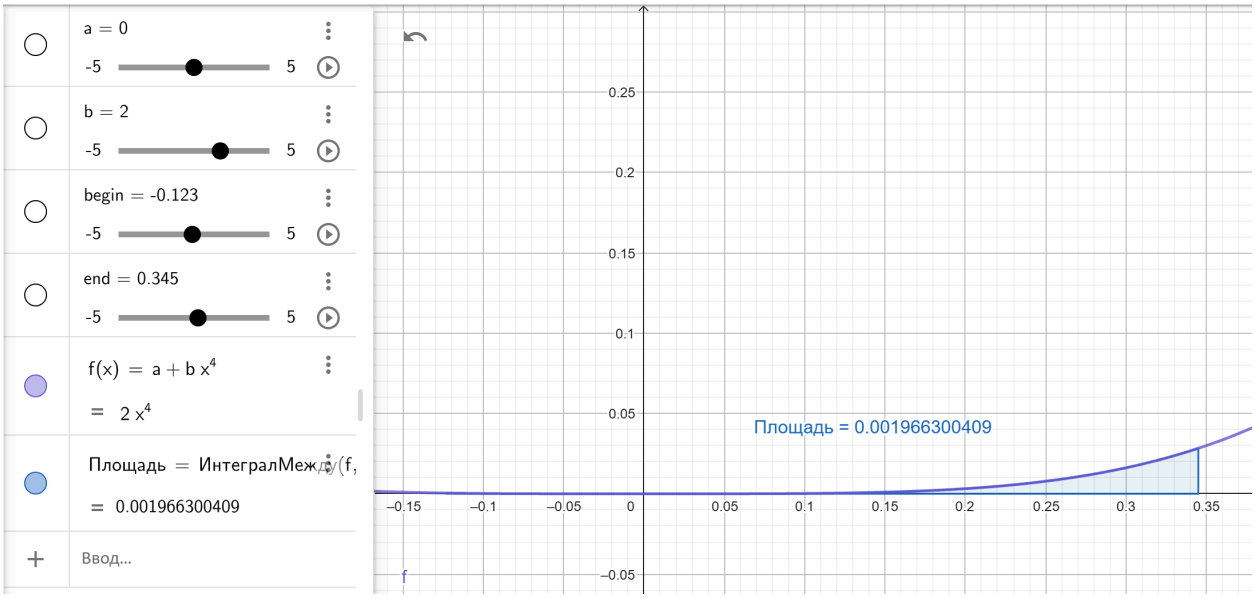
```
Введите действительное число a: 0
Введите действительное число b: 1
Введите действительное число begin - начало отрезка: -1
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 1
--------------------------------------------------------------------------------------------------------
Интеграл равен: 0.4
```



Тест 9 - функция вида $y = bx^4$

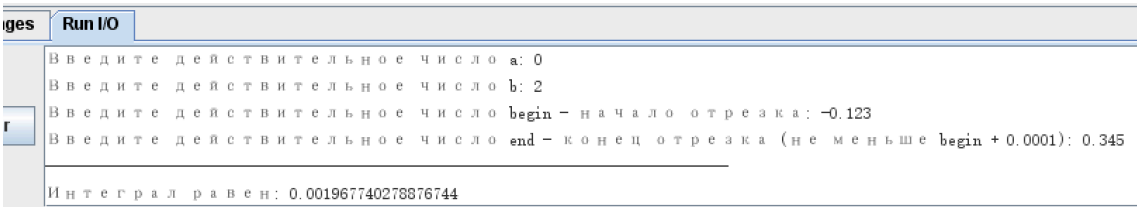
Ввод: a = 0, b = 2, begin = -0.123, end = 0.345

Предполагаемый вывод:



Вывод:

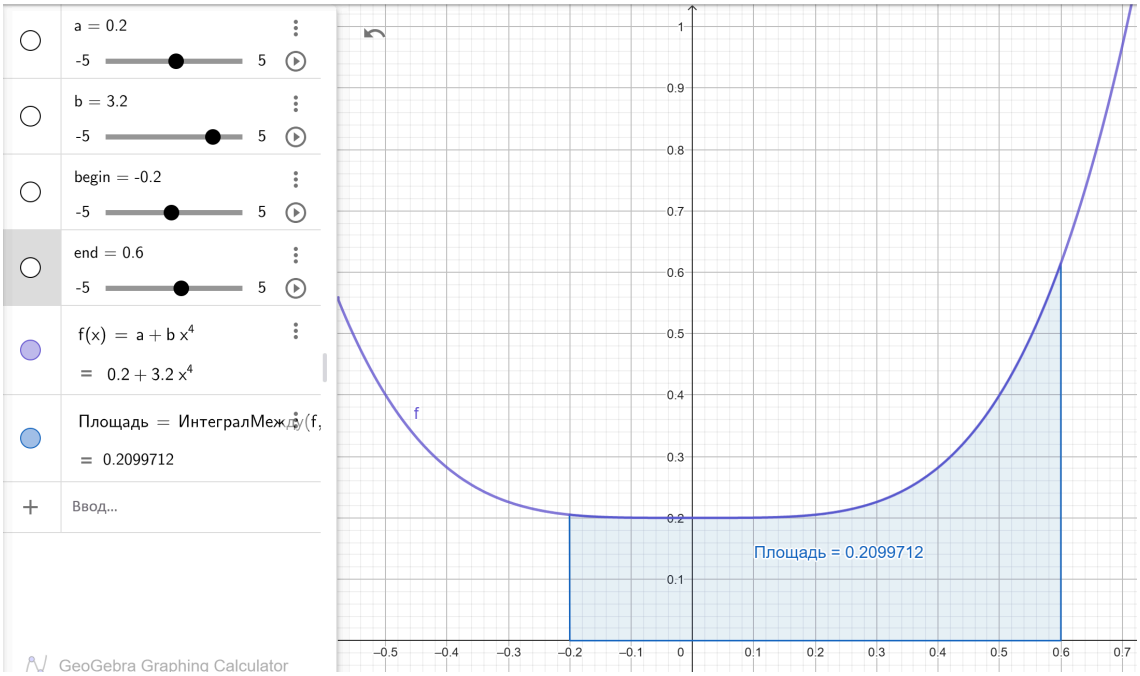
```
Введите действительное число a: 0
Введите действительное число b: 2
Введите действительное число begin - начало отрезка: -0.123
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 0.345
--------------------------------------------------------------------------------
Интеграл равен: 0.0019663004089871994
```



Тест 10 - $y = a + bx^4$

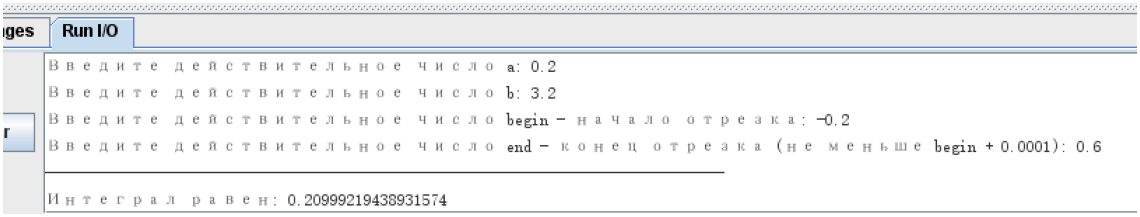
Ввод: a = 0.2, b = 3.2, begin = -0.2, end = 0.6

Предполагаемый вывод:



Вывод:

```
Введите действительное число a: 0.2
Введите действительное число b: 3.2
Введите действительное число begin - начало отрезка: -0.2
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 0.6
--------------------------------------------------------------------------------
Интеграл равен: 0.2099712
```



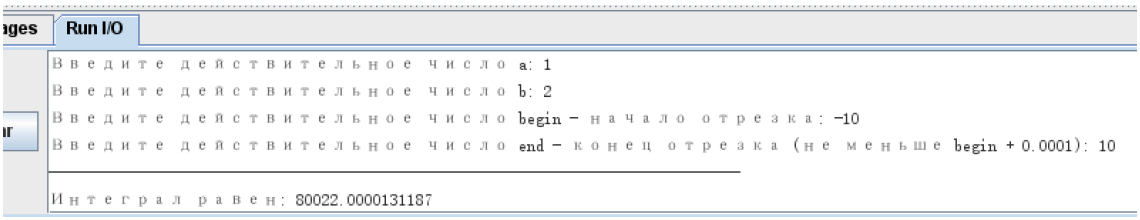
Тест 11 - большой отрезок

Ввод: a = 1, b = 2, begin = -10, end = 10

Предполагаемый вывод:

Вывод:

```
Введите действительное число a: 1
Введите действительное число b: 2
Введите действительное число begin - начало отрезка: -10
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 10
--------------------------------------------------------------------------------
Интеграл равен: 80020.0
```



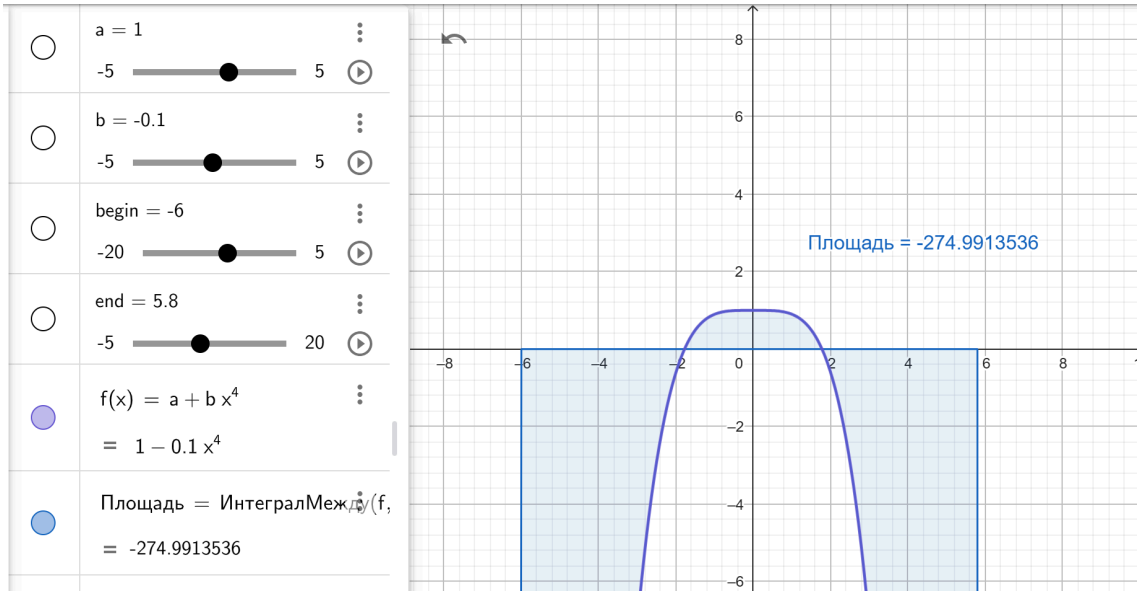
Избыток уже весомый получился, проверим аналогичной программой на питоне, что так и должно быть:

```
Введите действительное число a: 1
Введите действительное число b: 2
Введите действительное число begin - начало отрезка: -10
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 10
Интеграл равен: 80022.0000131187
```

Тест 11 - отрицательный интеграл

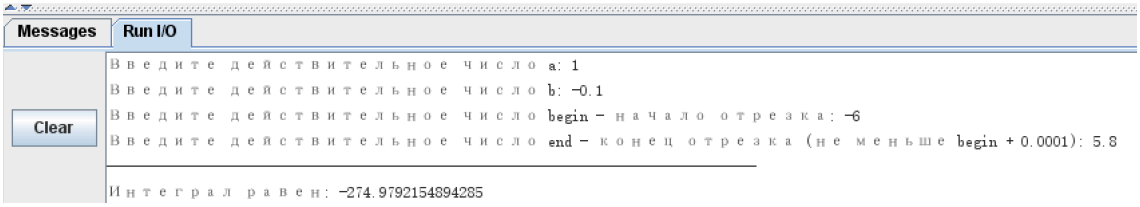
Ввод: a = 1, b = -0.1, begin = -6, end = 5.8

Предполагаемый вывод:



Вывод:

```
Введите действительное число a: 1
Введите действительное число b: -0.1
Введите действительное число begin - начало отрезка: -6
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 5.8
--------------------------------------------------------------------------------
Интеграл равен: -274.9913536
```



И что выводит аналогичная программа на питоне:

```
Введите действительное число a: 1
Введите действительное число b: -0.1
Введите действительное число begin - начало отрезка: -6
Введите действительное число end - конец отрезка (не меньше begin + 0.0001): 5.8
Интеграл равен: -274.97921548942844
```

Соответствие требованием на оценку: 7-9

Сформированный отчет должен содержать:

- фамилию, имя, отчество студента;
- номер группы;
- номер варианта задания;
- условие задачи;
- описание метода решения задачи;
- ссылки источник или источники информации с описанием метода решения задачи;
- описание тестовых прогонов с представлением информации о результатах тестирования.

При невыполнении хотя бы одного из требований оценка снижается.

Вот это всё сделано (разве что комментарии на русском написаны всё-таки, но тут можно код скопировать и вставить, и всё работать будет)

4–5 баллов

- Приведено решение задачи на ассемблере. Ввод данных осуществляется с клавиатуры. Вывод данных осуществляется на дисплей.
- В программе должны присутствовать комментарии, поясняющие выполняемые действия.

- Допускается использование требуемых подпрограмм без параметров и локальных переменных.
- В отчете должно быть представлено полное тестовое покрытие. Приведены результаты тестовых прогонов. Например, с использованием скриншотов.

- Решение задачи на ассемблере приведено, ввод данных совершается с клавиатуры, на дисплей осуществляется вывод значения интеграла, посчитанного метод прямоугольников с избытком

- В программе присутствуют комментарии
- Решение задачи разбито на подпрограммы, какие-то из них не используют локальные переменные и параметры, какие-то используют их (параметры - значения в сохранённых регистрах). Так же в решении были использованы макро из `macrolib` для создания более “лаконичного” и понятного решения и самостоятельно написанные макро: `read_double`, `print_double`, `max` и `calc_y` (описания можно почитать в комментариях, все макро работают с числами типа double)

6–7 баллов

При разработке программы на данную оценку необходимо учитывать все требования, предъявляемые на предшествующую оценку.

- В программе необходимо использовать подпрограммы с передачей аргументов через параметры, что должно обеспечивать их повторное использование с различными входными аргументами, включая, например, применение в других программах. При нехватке регистров, используемых для передачи параметров, оставшиеся параметры передавать через стек.

- Внутри подпрограмм необходимо использовать локальные переменные или свободные регистры. То есть, подпрограмма должна быть полностью отделена от вызывающего ее кода. При нехватке временных регистров обеспечить сохранение данных на стеке в соответствии с соглашениями, принятыми для процессора.

- в подпрограммы передаются аргументы через соответствующие регистры. Через регистры типа t передаются аргументы, значения в которых мы не боимся потерять (и дальше не используем в программе), через регистры типа s передаём значения, которые нам будут нужны далее в программе
- используются макро, которые мы можем использовать с различными входными аргументами и можем применять в других программах. Регистров хватало, поэтому стек не использовался
- свободные регистры использовались в подпрограммах и макро, при этом макро полностью отделены от вызывающего кода (то есть не используют переменные из вызывающей программы, пользуются переданными в качестве аргументов регистрами и регистрами типа t, чтобы “не испортить” никакие данные из основной программы)

- В местах вызова функции добавить комментарии, описывающие передачу фактических параметров и местоположение возвращаемого результата. При этом необходимо отметить, какая переменная или результат какого выражения соответствует тому или иному фактическому параметру.
- Информацию о проведенных изменениях отобразить в отчете наряду с информацией, необходимой на предыдущую оценку.

- перед макросами описано, как принимаются параметры и возвращается результат; в остальных местах понятно из комментариев

8 баллов

При разработке программы на данную оценку необходимо учитывать все требования, предъявляемые на предшествующие оценки.

- Разработанные подпрограммы должны поддерживать многократное использование с различными наборами исходных данных, включая возможность обработки в качестве параметров различных исходных данных.

- макросы этому соответствуют

- Реализовать автоматизированное тестирование за счет создания **дополнительной тестовой программы**, осуществляющей прогон подпрограмм, осуществляющих вычисления для различных тестовых данных (вместо их ввода). Осуществить прогон тестов обеспечивающих покрытие различных ситуаций. В том случае, если исходные данные напрямую не прописаны, а точность в условии зафиксирована, использовать организацию вычислений с различной точностью.

- вот этого у меня нет :(

- Для дополнительной проверки корректности вычислений осуществить аналогичные тестовые прогоны с использованием существующих библиотек и одного из языков программирования высокого уровня по выбору: C, C++, Python.
- Добавить информацию о проведенных изменениях в отчет.

- тестовые прогоны осуществлены с использованием программы на языке программирования Python (и дополнительно на Geogebra)

9 баллов

При разработке программы на данную оценку необходимо учитывать все требования, предъявляемые на предшествующие оценки.

- Добавить в программу использование макросов для реализации ввода и вывода данных. Добавить свои макросы, обертывающие подпрограммы обработки данных разработанные для решения основной задачи. Макросы должны поддерживать повторное использование с различными входными и выходными параметрами.

При невыполнении хотя бы одного из требований оценка снижается.

- макросы для реализации ввода и вывода данных добавлены: read_double и print_double
- изначально не увидела слово “обёртывающие” и думала, что нужны макросы для обработки данных, то есть вспомогательные функции, у меня это `max` и `calc_y`. Макросы-обёртки нужны для декомпозиции кода видимо, пишутся они вот так:

```
.macro input
    jal _input  # в исходном коде надо разделить main на _input и _check
.end_macro

.macro print
    jal print_result
.end_macro
```