Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | ***«Московский государственный технический университет  имени Н.Э. Баумана»***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Компьютерные системы и сети

**Отчет**

**по Лабораторной работе № 6**

**Вариант № 16**

**Дисциплина: Языки интернет программирования**

**Название домашней работы: Анализ линейной электрической цепи постоянного тока**

Студент гр. ИУ6-34 **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** Лавренов И.С.

(Подпись, дата) (И.О. Фамилия)

Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О. Фамилия)

Москва, 2017

# Задание

Часть 1

Организовать итерацинный цикл. Вычислить значение непрерывной дроби 1+1/(1+1/(1+1/1+…)) с точностью 0.001, 0.0001

Часть 2

Решить предыдущее задание с помощью enumerator.

Часть 3

Найти минимальный положительный корень уравнения с точностью до 0.0001

Уравнения:

x^2+sin(x/2)=0

Arctg(x)+x=1

# Часть 1

Код interface.rb

*require\_relative* **'./main.rb'**print **'Input precision 1e-3 or 1e-4: '***precision\_s* = gets  
*precision* = *precision\_s*.to\_f  
print **'Input radius: '***radius\_s* = gets  
*radius* = *radius\_s*.to\_f  
**if** [1e-3, 1e-4].include? *precision* print calc(*precision*,*radius*)  
**else** print **'Choose one of given numbers'  
end**

Код main.rb

**def** *side*(*previous\_side*,*r*)  
 ***Math***.sqrt(2\**r*\*\*2 - 2\**r*\****Math***.sqrt(*r*\*\*2 - *previous\_side*\*\*2/4))  
**end  
  
def** *calc*(*precision*,*rad*)  
 *current\_side* = *rad  
 sides\_num* = 6  
 **while** ((*sides\_num*\*2)\*side(*current\_side*,*rad*) - *sides\_num*\**current\_side*) > *precision* / 2  
 *current\_side* = side(*current\_side*,*rad*)   
 *sides\_num* \*= 2  
 **end** *current\_side*\**sides\_num***end**

Код test.rb

*require* **'minitest/autorun'***require* **'./main.rb'  
  
class *Test*** < ***MiniTest***::***Test* def** *setup* **end  
   
 def** *teardown* **end  
   
 def** *test\_ok* (3..100).each **do** |*r*|  
 assert\_in\_delta 2\**r*\****Math***::***PI***, calc(1e-3, *r*.to\_f), 1e-3  
 assert\_in\_delta 2\**r*\****Math***::***PI***, calc(1e-4, *r*.to\_f), 1e-4  
 **end  
 end  
   
end**

Скриншоты

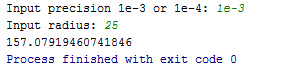


Рис.1. Выполнение программы



Рис.2. Выполнение теста

# Часть 2

Код interface.rb

*require\_relative* **'./main.rb'**print **'Input precision 1e-3 or 1e-4: '***precision\_s* = gets  
*precision* = *precision\_s*.to\_f  
print **'Input radius: '***radius\_s* = gets  
*radius* = *radius\_s*.to\_f  
**if** [1e-3, 1e-4].include? *precision* print calc(*precision*,*radius*)  
**else** print **'Choose one of given numbers'  
end**

Код main.rb

**def** *side*(*previous\_side*,*r*)  
 ***Math***.sqrt(2\**r*\*\*2 - 2\**r*\****Math***.sqrt(*r*\*\*2 - *previous\_side*\*\*2/4))  
**end  
  
def** *calc*(*precision*,*rad*)  
 *sides\_num* = 6.0  
 *current\_side* = *rad  
 list* = ***Enumerator***.*new* **do** |*calculating*|  
 loop **do** *calculating*.yield *current\_side  
 current\_side* = side(*current\_side*,*rad*)  
 *sides\_num* \*= 2  
 **end  
 end** *list*.take\_while {|*current\_side*| ((*sides\_num*\*2)\*side(*current\_side*,*rad*) - *sides\_num*\**current\_side*) > *precision*/2 }  
 *current\_side*\**sides\_num***end**

Код test.rb

*require* **'minitest/autorun'***require* **'./main.rb'  
  
class *Test*** < ***MiniTest***::***Test* def** *setup* **end  
   
 def** *teardown* **end  
   
 def** *test\_ok* (3..100).each **do** |*r*|  
 assert\_in\_delta 2\**r*\****Math***::***PI***, calc(1e-3, *r*.to\_f), 1e-3  
 assert\_in\_delta 2\**r*\****Math***::***PI***, calc(1e-4, *r*.to\_f), 1e-4  
 **end  
 end  
end**

Скриншоты

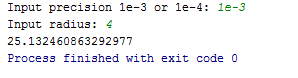


Рис.3. Выполнение программы



Рис.4. Выполнение теста

# Часть 3

Код interface.rb

*require\_relative* **'./main.rb'**puts **"Lambda: "**p *root*(0, 300, 0, ->(*x*) {*x*\*\*2 + ***Math***.sin(*x*/2)})  
p *root*(0, 300, 1, ->(*x*) {***Math***.atan(*x*) + *x*})  
puts **"Block: "**p *root*(0, 300, 0) {|*x*| *x*\*\*2 + ***Math***.sin(*x*/2)}  
p *root*(0, 300, 1) {|*x*| ***Math***.atan(*x*) + *x*}

Код main.rb

**def** *root*(*a*, *b*, *answer*=0, *equation* = **nil**)  
 **while** (*b*-*a* > 0.0001)  
 *c* = (*a*+*b*)/2.0  
 **if** block\_given?  
 **if** (**yield**(*b*)\***yield**(*c*) < *answer* )  
 *a* = *c* **else** *b* = *c* **end  
 else  
 if** (*equation*.call(*b*) \* *equation*.call(*c*) < *answer*)  
 *a* = *c* **else** *b* = *c* **end  
 end  
 end** (*a*+*b*)/2  
**end**

Код test.rb

*require* **'minitest/autorun'***require\_relative* **'./main.rb'  
  
class *Test*** < ***MiniTest***::***Test* def** *setup* **end  
   
 def** *teardown* **end  
   
 def** *test\_ok* assert\_in\_delta 3.576e-05, *root*(0, 300, 0, ->(*x*) {*x*\*\*2 + ***Math***.sin(*x*/2)}), 1e-3  
 assert\_in\_delta 0.513, *root*(0, 300, 1) {|*x*| ***Math***.atan(*x*) + *x*}, 1e-3  
 **end  
   
end**

Скриншоты

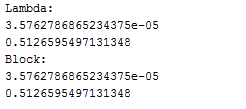


Рис.5. Выполнение программы

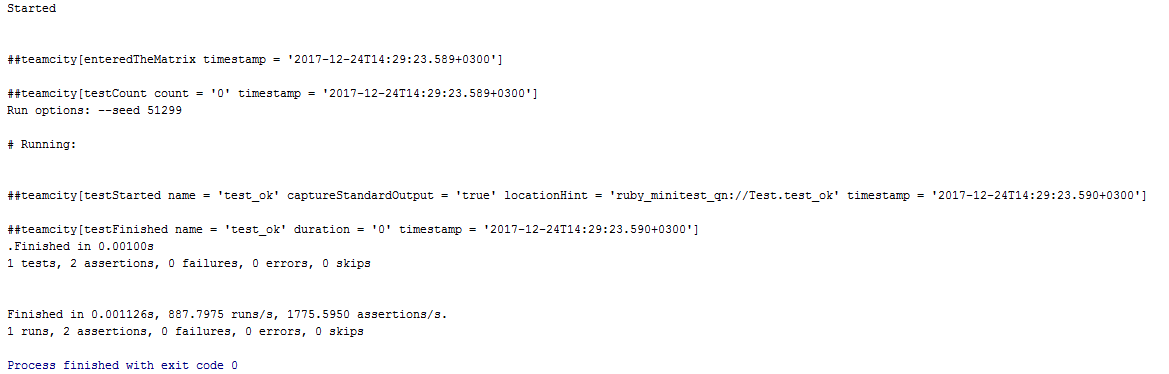


Рис.6. Выполнение теста

# Вывод

Было создано 3 консольных ruby приложений по выданному условию. Все программы имеют тесты. Приложения протестированы и работают верно.