CRIMEA DIGITAL GROUP ACADEMY

ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ

ЛЗ #1

Теория

Задание

Теория

Задание

SOFTWARE ENGINEERING

В целом о лабах

Мы будем изучать базовые основы языка программирования Ruby, напишем свой HTTP сервер и начнем писать тесты.

SOFTWARE ENGINEERING

Что надо знать о Ruby

👉 Он очень популярен и прост в изучении. Сама философия языка ставит во главу угла понятность и простоту написания кода.

👉 Язык зрелый и ему больше 20 лет.

👉 Ему предрекают закат уже много лет, но что-то ничего не меняется, а в рейтингах популярности языков он особо не падает. Разве что на пару позиций.

Синтаксис

irb

Это интерактивная консоль для выполнения Ruby кода. В ней можно тестировать свои куски кода и проверять как работает ваш функционал.

```
➜  ~ irb
irb(main):001:0> 1 + 2
=> 3
irb(main):002:0> puts "hey there"
hey there
=> nil
irb(main):003:1* def print_hello
irb(main):004:1*   puts "hello"
irb(main):005:0> end
=> :print_hello
irb(main):006:0> print_hello
hello
=> nil
irb(main):007:0>
```

хеловорлд?

В качестве STDOUT в руби выступает команда **puts**

helloworld.rb

```
#!/usr/bin/ruby
puts "hello world"
```

```
➜  ~ ruby helloworld.rb
hello world
➜  ~
```

переменные

В руби есть локальные переменные (1), константы (2), глобальные переменные (3) и переменные экземпляра или инстанс переменные (4)

Как и везде, их можно присваивать, складывать, вычитать и проделывать с ними различные операции и преобразования

```ruby
# 1
a = 1
b = 2
c = a + b

# 2
CONST_A = 1
CONST_B = 2

# 3
$a = 1
$b = 2

# 4
@a = 1
@b = 2
```

Условия

В условиях нет необходимости ставить какие-либо скобочки или что-то подобное, а самое объявление операторов контроля потока ничем не отличается от других языков

```ruby
flag = true

if flag
  puts "Hello world"
end
# => Hello world

flag = false

if flag
  puts "Hello world"
else
  puts "Goodbye world"
end
# => Goodbye world
```

Условия

Кроме пары интересных моментов :)

```ruby
flag = false
another_flag = true

if flag
  puts "Hello world"
elsif another_flag
  puts "Hello another world"
end
# => Hello another world

flag = false

unless flag
  puts "Hello world"
else
  puts "Goodbye world"
end
# => Hello world
```

Условия

И еще пары :))

```ruby
flag = true
puts "Hello world" if flag
# => "Hello world"

puts "Hello world" unless flag
# => nil

flag ? "Hello world" : "Goodbye world"
# => Hello world
```

Методы

Методы - это исполняемые блоки кода, которые могут принимать аргументы

```ruby
def say_hello
  puts "Say hello
end

say_hello
# => Say hello
```

Методы

Методы - это исполняемые блоки кода, которые могут принимать аргументы

```ruby
def greeting(username)
  puts "Hello #{username}"
end

name = gets

greeting(name)

# type Alex in prompt
# => Hello Alex
```

Массивы

Массивы - это коллекции элементов, с которыми можно всячески взаимодействовать

```
[1, 2, 3, 4, 5]
# => [1, 2, 3, 4, 5]

[1 , false, 2, "bar", 3, true]
# => [1, false, 2, "bar", 3, true]

arr = []
arr << "bar"
arr.push(1)
# => ["bar", 1]

arr.delete("bar")
arr.delete(1)
# => []
```

Массивы

```ruby
arr[0]
# => "bar"

arr[1]
# => :foo

arr[0, 1]
# => ["bar", :foo]

arr.slice(1)
# => :foo

arr.slice(0, 7)
# => ["bar", :foo]
```

```ruby
arr.empty?
# => false

arr.size
arr.count
# => 2

arr.empty?
# => true

arr.each do |item|
  puts item
end
# bar
# foo
```

Циклы

Циклов много разных. Но основных 2. Один проходит по массиву не изменяя его. Второй проходит и создает новый массив, с результатами преобразований изначального массива.

```ruby
[1, 2, 3].each { |value| puts value }
# 1
# 2
# 3
# => [1, 2, 3]

[1, 2, 3].map { |value| value + 1 }
# => [2, 3, 4]
```

Циклы

Есть также **do/while, for/in, until, loop** и несколько других типов циклов, но они почти никогда не используются.

```ruby
planets = [
  "Mercury", "Venus", "Earth",
  "Mars", "Jupiter", "Saturn",
  "Uranus", "Neptune"
]

for planet in planets
  puts planet
end

# ===========

sum = 0

loop do
  input = gets.to_i

  next if input == 0
  break if input == -1

  sum += input
end
```

Циклы

Когда действие нужно повторить определенное количество раз, удобно использовать цикл **times**

```ruby
i = 10

3.times do |value|
  puts i
  i = i * 10
end

# 10
# 100
# 1000
```

Ввод с клавиатуры

Для того, чтобы остановить исполнение программы и ввести что-то с клавиатуры существует команда **gets**

```
input = nil

loop do
  print "Enter a number (-1 quit) > "
  input = gets.to_i

  puts "You've entered #{input}"

  break if input == -1
end

puts "All done"
```

```
➜  ~ ruby snippet_01.rb
Enter a number (-1 quit) > 1
You've entered 1
Enter a number (-1 quit) > 2
You've entered 2
Enter a number (-1 quit) > 2356
You've entered 2356
Enter a number (-1 quit) > p23o
You've entered 0
Enter a number (-1 quit) > 982739857236
You've entered 982739857236
Enter a number (-1 quit) > -1
You've entered -1
All done
```

Теория

Задание

SOFTWARE ENGINEERING

Задание

👉 Написать скрипт, который будет запрашивать имя, фамилию и возраст, а затем выдавать приветствие в одном из двух вариантов, в зависимости от возраста:

- *Привет, {имя} {фамилия}. Тебе меньше 18 лет, но начать учиться программировать никогда не рано*
- *Привет, {имя} {фамилия}. Самое время заняться делом!*

👉 Написать скрипт, который принимает пару чисел, если хотя бы одно равно 20 - возвращает его, в противном случае выводит сумму этих чисел.

* Старайтесь организовывать код в методы

КОНТАКТЫ

Для быстрого входа
на сайт

courses@crimeadigital.ru

tel: 8 (800) 551 44 86

https://crimeadigital.ru