Факультет компьютерных технологий и прикладной математики

Кафедра вычислительных технологий

02.03.02

Паттерны программирования

Лабораторная работа № 1. Введение

Вопросы:

1. Что такое аргумент программы? Для чего он может быть полезен?
2. Как перейти в irb режим? Продемонстрировать.
3. Как получить методы класса и методы объекта? В чем разница? Покажите на примере.

#### Методы класса

Методы класса относятся ко всему классу и могут быть вызваны без создания экземпляра класса. Они разрабатываются внутри блока classили модуля.

Методы объекта являются частью экземпляра класса и могут вызываться только в экземплярах этого класса.

1. Описать 5 методов объекта класса строка.
   * + 1. Верхний регистр: upcase()

str = "hello world"

puts str.upcase # Выведет: HELLO WORLD

* + - 1. Нижний регистр: downcase()

str = "HELLO WORLD"

puts str.downcase # Выведет: hello world

* + - 1. Заглавные буквы:capitalize()

str = "hello world"

puts str.capitalize # Выведет: Hello World

* + - 1. Обратный: reverse()

str = "hello"

puts str.reverse # Выведет: olleh

* + - 1. swapcase () используется для замены всех символов строки, которые находятся в верхнем регистре, на символы в нижнем регистре и наоборот.
      2. Разделить: Метод split()разделения показывает текст массива слов по указанному разделителю (по умолчанию - пробел).

str = "apple banana cherry"

fruits = str.split

puts fruits # Выведет: ["apple", "banana", "cherry"]

* + - 1. Дополнительные методы:

Length()

include?(substring): впоследствии выяснилось ли подстрока в строке.

strip(): удалить лишние пробелы в начале и конце строки.

concat(string): добавить текст к концу текущей строки.

1. Расскажите, что такое форматирование строки? Приведите пример.

Форматирование строки — это процесс создания строки, которая содержит как статический текст, так и активные элементы, такие как переменные, выражения и другие данные. Основная цель формирования строки – обеспечить простое и понятное представление информации.

Примеры:

* + 1. name = "Иван"

age = 30

puts "Меня зовут #{name} и мне #{age} лет."

* + 1. name = "Иван"

age = 30

puts "%s, возраст: %d" % [name, age]

1. Покажите 4 способа реализации условного оператора в ruby.
   1. If c==0 then 2. Unless x>5 then

Puts “hello” puts “7”

Else else

Puts ”no” puts”10”

End end

3.x=if d>7 then n 4. X=unless if d<7 then n

Else c end else c end

1. Как выполнить команду языка, хранящуюся в строке?
   * 1. Eval()-используется для выполнения Ruby-кода, хранящегося в строке. возвращает результат выполнения кода.
     2. Exec()-используется для выполнения командной обработки и замены текущего процесса. возвращает статус выполнения команды.
2. Как выполнить команду ОС, хранящуюся в строке?
   * + 1. Использование системы:

command = "ls -l"

status = system(command)

puts "Команда выполнена успешно" if status

Этот метод выполняет команду и возвращает статус выполнения.

* + - 1. Использование исполнителя

command = "echo Hello from another process"

exec(command)

Этот метод заменяет текущий процесс новым и завершает его после выполнения команды.

* + - 1. Использование обратных ковычек

output = `ls -l`

puts output

Этот метод возвращает результат выполнения команды как текст.

1. Как описывать и вызывать функции в ruby? Что будет возвращать описываемая Вами функция по умолчанию? Почему это возможно?

def имя\_функции(параметры)

# Тело функции

End

имя\_функции(аргументы)

Если не указано, метод в Ruby по умолчанию возвращает nil.

Возможно, это потому, что Ruby использует «последние выражения» для определения значения, метод которого должен вернуть. Если ни одно выражение в теле методе не извлекает значения явно, этот метод автоматически возвращает nil

1. Опишите класс Array, основные методы объекта класса, описать их принадлежность – модули или суперклассы.

* Метод at получает ссылку на элемент массива

x = a.at(2)

* Метод values\_at возвращает подмассив по списку индексов

x = a.values\_at(2..5, 7, 9)

* first и last – получение первого и последнего элементов
* size и length – длина массива
* nitems – длина без учета nil-элементов

• Диапазоны можно задавать двумя способами

• Две точки: n..m – диапазон [n, m]

• Три точки: n…m – диапазон [n, m-1]

1. Опишите известные Вам способы задать массив.

a = Array.[](1,2,3)

b = Array[1,2,3]

c = [1,2,3]

d = Array.new # пустой массив

e = Array(3) # [nil,nil,nil]

f = Array(3, "Test") # ["Test", "Test", "Test"]

h = ["Дверь", "запили", [1000, "Очень мало"]]

1. Приведите 5 способов добавить элемент в Массив.
   * 1. <<
     2. Push()
     3. Unshift()- добавляет в начало

array.unshift(\*[0, -1]) # Развернет массив [0, -1] перед добавление

# Результат: [-1, 0, 1, 2, 3]

* + 1. insert()позволяет вставить элементы в конкретное место массива
    2. Оператор + конкатенация

array = [1, 2, 3]

array = array + [4] # Создаст новый массив, не изменяя оригинальный

# Результат: [1, 2, 3, 4]

1. Опишите особенности работы с индексами в массиве на ruby.

### Особенности работы с индексами

1. Индексы отрицательные:
   * -1указывает на последний элемент массива.
   * -2, -3и т.д. указываются на предпоследний, третий с конца и так далее.
2. Доступ к различным элементам:
   * Можно использовать диапазоны для получения нескольких элементов.
   * Например, array[1..3]вернет [2, 3].
3. Методы работы с индексами:
   * index(): найди первое вхождение значения.
   * rindex(): найди последнее вхождение значения.
   * find\_index(): найдите индекс первого вхождения значени

### Ключевые

* Ruby использует нумерацию от 0 для индексов массивов.
* Аксессор []используется как для чтения, так и для записей.
* Попытка получить незаменимый элемент для волос.
* Индексы отрицательные указывают на элементы с конца массива.
* Методы index(), rindex()и find\_index()полезные для поиска элементов в массиве.