Programación en Lenguaje Ruby

Ejercicios

Delio Tolivia

Gijón crece 2016 / 2019
AYTO. DE GIJÓN / CCOO / FADE / UGT
Ayuntamiento de Gijón
Gijón impulsa Empleo
dicampus TALENTO FORMACIÓN

Indice

Ejercicios

Ejercicio 1

- Crear una calculadora que tenga un metodo para sumar dos numero y otro método para restarlos.

Ejercicio 2

- Definir un método que recibe un array de Strings y devuelve un array que contiene la longitud de esos Strings.

Ejercicio 3

- Dada una sentencia que contiene varias palabras, calcular la frecuencia de otra palabra dada en esa secuencia. El método por tanto recibirá dos Strings (sentencia, palabra). El método Array#count cuenta la frecuencia de un elemento en un array: [9,3,4,9,5].count(9) devolvera 2.

Ejercicio 4

- Crear un método que acepte un String reordene las palabras en orden ascendente por longitud (no habrá simbolos de puntuación y todas las palabras estarán separadas por espacios). Para ordenar un array se puede utilizar el método sort (buscar documentación). Para generar un string a partir de un array existe el método join.

Ejercicio 5

- Crear un método que dado un array de elementos y un numero retorne n elementos seleccionados aleatoriamente de ese array. Existe un método rand que dado un parámetro max da un numero aleatorio menor que ese max.

Ejercicio 6 (I)

- Queremos seleccionar candidatos para una empresa de programación. Crear una clase candidato que tenga como parámetros:
    - years_of_experience: Años de experiencia
    - github_points: puntos en git hub
    - languages_worked_with: array con Strings con el nombre de los lenguajes que conozca
    - applied_recently: booleano indicando si ha solicitado el trabajo recientemente
    - age: edad

- Una vez hecha la clase definir un método que nos indique si un candidato es válido basándose en:
    - Se buscan candidatos que sepan Ruby
    - Con más de dos años de experiencia
    - Si no tienen mas de dos años de experiencia pero tienen más de 500 puntos en GitHub también son validos
    - No pueden ser menores de 15 años
    - Se rechazaran los candidatos que hayan solicitado el trabajo recientemente.

Ejercicio 6 (II)

- Ejemplos:
  - Trabajador con conocimientos de Ruby, con 3 años de experiencia, 200 puntos en github, de 20 años y que no ha solicitado el trabajo recientemente ->Aceptado
  - Programador con conocimientos de Ruby, 1 año de experiencia, 600 puntos en GitHub, de 20 años y que no ha solicitado el trabajo recientemente -> Aceptado
  - Cualquier trabajador que no tenga Ruby entre sus lenguajes -> Rechazado
  - Cualquier trabajador que haya solicitado el trabajo recientemente -> Rechazado
  - Programador con conocimientos de Ruby, 1 año de experiencia, 300 puntos en Github, de 20 años y que no ha solicitado el trabajo recientemente -> Rechazado
  - Cualquier programador menor de 15 años -> Rechazado

- Probar nuestro método con todos estas opciones

Ejercicio 7

- Hacer un método que recibe una sentencia y devuelve true si es un palindromo (ignoramos espacios y mayusculas/minusculas) . El método reverse de cadenas puede ser util.

Ejercicio 8

- Hacer un método que recibiendo dos números calcule la suma de los cubos de los números del rango definido por los dos parámetros que recibe el método.

Ejercicio 9

- Definir un método que dado un array devuelva los elementos que están exactamente una vez en dicho array. Hay un método find_all que puede ser útil.

Ejercicio 10

- Definir un método que dado un array devuelva true si todos son de la clase Fixnums

Ejercicio 11

- Definir un método que indique si un numero es un Kaprekar (lo es si la suma de las cifras del número elevado a 2 es igual al número).
- Ejemplo:
- - 9^2 = 81 y 8+1=9
- - 297^2 = 88209 y 88+209 = 297 (genéricamente es que un numero de n cifras lo es si al hacer el cuadrado y sumamos los n dígitos de la derecha con los n o n-1 dígitos de la izquierda del resultado devuelve el número)

Ejercicio 12

- Para dos colores en RGB (R1,G1,B1) y (R2, G2, B2):
  - Índice de brillo: (299*R1 + 587*G1 + 114*B1)/1000
  - Diferencia de brillo: Diferencia absoluta entre los índices de brillo
  - Diferencia de matiz (Hue): |R1 –R2| + |G1 – G2| + |B1 – B2|

- Si la diferencia de brillo es mas de 125 y la diferencia de matiz es mayor de 500 dos colores tienen suficiente contraste.

- Hacer una clase Color con métodos que calculen esos datos y tenga un método que diga si dos colores dados tienen suficiente contraste

- Ejemplos:
  - Entre (42,21,58) y (240,41, 25) no hay suficiente contraste
  - Entre (42,42,42) y (210,210,210) hay suficiente contraste

Ejercicio 13

- Definir un método que recibe un código en forma de lambda y calcule el tiempo que tarda en ejecutarse. El tiempo actual se obtiene con Time.now. Probar por ejemplo a realizar una operación una vez y la misma operación en un bucle 10 veces.

Ejercicio 14

- Dado un número con 3 o 4 dígitos diferentes devolver un array ordenado con todos los números que se pueden hacer con esos dígitos. El método uniq sobre un array devuelve un array con los elementos únicos del primero. El método shuffle puede ser útil.

Ejercicio 15

- Hacer una clase Restaurante que reciba al construir un objeto, un menu en forma de Hash:

- Por ejemplo {:rice =>1, :noodles =>1}

- Tendrá un método coste que calculara el coste de una serie de pedidos en forma de Hash:

- Por ejemplo {:rice =>1, :noodles=>2} y {:rice=>3, :noodles=>2} devolverá un coste de 8 (para le menú anterior)

Ejercicio 16

- Hacer una clase MyArray que recibe en su constructor un array de números y tiene un método sum que recibe un parámetro que es un valor inicial al que sumar los valores del array y que puede recibir un bloque como parámetro. Ejemplos:

  my_array = MyArray.new([1, 2, 3])
  my_array.sum dará de resultado 6
  my_array.sum(10) dará de resultado 16
  my_array.sum(0) {\|n\| n ** 2 } dará de resultado 14 (x ** y es x^y)