dicampus
TALENTO FORMACIÓN

Ejercicios Ruby

Manual de ejercicios

Delio Tolivia Cadrecha

JUEVES 22 DE FEBRERO DE 2018

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

Índice

1. Reloj Digital

El usuario deberá interactuar con el programa colocando un horario X

El mismo pedir la HORA, MINUTOS y SEGUNDOS, luego de ingresar los datos el reloj se imprimirá y empezara su funcionamiento adecuado.

```
## entrar datos
hora = gets.chomp.to_i
mint = gets.chomp.to_i
segs = gets.chomp.to_i

## contar 24 horas
if( hora < 24 && mint < 60 && segs << 60)
    while(true)
        for hora in hora...24
            for minuto in mint...60
                for segundo in segs...60
                    puts "#{hora}:#{minuto}:#{segundo}"
                    sleep(1)
                end
                segs = 0
            end
            mint= 0
        end
        hora = 0
    end
end
```

2. Numeros Divisibles

Dado un intervalo de números enteros positivos, indicado por el usuario, nos muestre cuantos números del intervalo son divisibles entre otro, también indicado por el usuario

```
## leer la entrada del usuario
puts "Escriba el divisor"; divisor = gets.chomp.to_i

## entrada del rango
puts "Escriba el inicio del rango"; rango_inicio = gets.chomp.to_i

## final del rango
puts "Escriba el final del rango"; rango_final  = gets.chomp.to_i

i = 0
for numero in rango_inicio..rango_final do
    if ( numero.modulo(divisor) == 0)
        p numero
        i = i +1
    end
end


puts "En el rango de #{rango_inicio} a #{rango_final} hay #{i} divisores de #{divisor}"
```

3. Suma de Pares e Impares

Crear una clase que contenga un método al cual se le pase un número y que este retorne la suma de los numeros pares y de los numeros impartes. El numero que se le pasa al método es limitador a los numeros que hay que sumar.

Por ejemplo, si le pasamos un 5 tendría que devolver lo siguiente:

Suma de Impares: 9

Suma de pares: 6

Primero devuelve 9 ya que 1 3 5 son 9 y

después devuelve 6 ya que 2 + 4 son 6.

```
class SumParesImpares
  def initialize(num)
    @sumPares = 0
    @sumImpares = 0
    @num = num
  end
  def sumar_pares
    1.upto @num do |p|
      if p % 2 == 0
        @sumPares += p
      end
    end
    @sumPares
  end
  def sumar_impares
    1.upto @num do |i|
      if i % 2 != 0
        @sumImpares += i
      end
    end
    @sumImpares
  end
end

print "Ingrese el numero maximo a sumar: "
num = gets().to_i

operacion = SumParesImpares.new(num)
puts "La suma de los numero pares es: #{operacion.sumar_pares}"
puts "La suma de los numero impares es: #{operacion.sumar_impares}"
```

4. Código Morse

El usuario introduce por pantalla una palabra y el programa la traduce a código morse.

```
print "Ingrese una frase, sin vocales con tildes: "
gets.chomp!.each_char do |c|
  print ( { A: ".-", B: ".---", C: "-.-.", D: "-..",\
            E: ".", F: "..-.", G: "--.", H: "....",\
            I: "..", J: ".---", K: "-.-", L: ".-..",\
            M: "--", N: "-.", Ñ: "--.--", O: "---",\
            P: ".--.", Q: "--.-", R: ".-.", S: "...",\
            T: "-", U: "..-", V: "...-", W: ".--", X: "-..-",\
            Y: "-.--", Z: "--..", 0 => "-----", 1 => ".----",\
            2 =>  "..---", 3 =>  "...--", 4 =>  "....-", 5 =>  ".....",\
            6 =>  "-....", 7 =>  "--...", 8 =>  "---..", 9 =>  "----."\
  }[c.upcase.to_sym] ), " "
end
puts
```

5. Clave Murcielago

Tenemos la palabra "Murcielago" reemplazamos cada letra con un número del 0 al 9 así:

M U R C I E L A G O

0 1 2 3 4 5 6 7 8 9

Tomando la conversión el programa debe traducir las palabras reemplazando las letras por números. Ejm:

miguel esta durmiendo = 048156 5st7 d12045nd9

Como añadido, el programa debe poder hacer lo inverso:

048156 5st7 d12045nd9 = miguel esta durmiendo

```
def encriptar entrada
    cripto = Array.new
    entrada.each_char do |letra|
        if $clave.include? letra
            cripto.push $clave.index letra
        else
            cripto.push letra
        end
    end
    cripto.join
end

def desencriptar entrada
    indices = entrada.scan(/\d/)
    indices.each { |indice| entrada[ entrada.index indice] = $clave[indice.to_i]}
    entrada
end

$clave = 'murcielago'[0...9]; entrada = gets.chomp.to_s
puts 'Palabra    encriptada : ' + encriptar(entrada), 'Palabra desencriptada : '
+desencriptar(encriptar(entrada))
```

6. Números Capicua

Recibe por teclado dos numeros enteros, Para cada par a, b imprime en una linea distinta cuantos n´umeros capicua hay en el intervalo [a, b]. Por ejemplo recibe 20 40, imprime 1, ya que el unico numero capicua en ese intervalo es 33.

```
def capicua?( number )
    return number.to_s==number.to_s.reverse
end

def count_capicuas_from_range( range )
    return range.select{ |number| capicua? number }.size
end


if $0==$__FILE__
    print "Ingresá el 1er número: "; a=gets.chomp
    print "Ingresá el 2do número: "; b=gets.chomp
    puts "#{count_capicuas_from_range( a..b )} capicuas encontrada entre #{a} y #{b}"
end
```

7. Cuadrado de un número

El cuadrado de un número "n" se obtiene sumando los "n" primeros números impares. Ejemplo: 3 al cuadrado es igual a 135=9. Hallar el cuadrado del número

```
def number_square( number )
  return (2*number).times.select { |n| n%2==1 }.reduce( :+ )
end
```

8. Triangulo de Floyd

Dado un N, mostrar en el terminal el Triangulo de Floyd correspondiente

el N determina la cantidad de filas y de columnas que determinan el tamaño del triangulo.

ejemplo:

FloydTriangle(5)

1

2 3

4 5 6

7 8 9 10

11 12 13 14 15

```
def triangulo(n)
    l = []
    ((n**2 + n)/2).times {|i| l << i + 1 }

    x = 0
    f = 0
    for i in 1..n
        x = f + i
        print l[f,i].to_s + "\n"
        f = x
    end
end

triangulo(5)
```

9. Números Bonitos

Crear una función que retorne "si" o "no" dependiendo si el numero ingresado es bonito o no.

Un numero es bonito si la suma de sus dígitos es igual a la suma de los dígitos de (3n + 11)

Ej: 8 es bonito ya que 8*3 + 11 = 35 y la suma de los dígitos de 35 es 8

17. 17*3 + 11 = 62

```
def es_bonito numero
    res = 0
    mod = 0
    resultado = ((numero*3)+11).to_i
    res = resultado

    while(res > 0)
        mod = mod + (res % 10)
        res = res / 10
    end

    if numero == mod
        puts "!Felicidades! el número #{numero} es bonito :)"
    else
        puts "Ups! Los siento! el #{numero} no es bonito :("
    end
end
```

10. Piedra-tijera-papel-spock

En la serie de televisión The Big Bang Theory , el popular personaje Sheldom Cooper, propone una extensión del popular juego piedra papel tijera. Con el fin de hacerlo más divertido añade dos variables nuevas al juego: lagarto y spock

Si las dos elecciones son las mismas es un empate

- Las tijeras cortan el papel
- El papel cubre a la piedra
- La piedra aplasta al lagarto
- El lagarto envenena a Spock
- Spock destroza las tijeras
- Las tijeras decapitan al lagarto
- El lagarto se come el papel
- El papel refuta a Spock
- Spock vaporiza la piedra
- La piedra aplasta las tijeras.

```
class LagartoSpock
  WIN, DRAW, LOSE = 1, 0, 2
  RULES = {
    tijeras: [:lagarto, :papel],
    lagarto: [:spock, :papel],
    spock:   [:piedra, :tijeras],
    piedra:  [:lagarto, :tijeras],
    papel:   [:piedra, :spock]
  }

  def self.fight(move_one, move_two)
    return DRAW if move_one == move_two
    RULES[move_one].include?(move_two) ? WIN : LOSE
  end
end
```

11. Cifrado Cesar

En criptografía, el cifrado César, también conocido como cifrado por desplazamiento, código de César o desplazamiento de César, es una de las técnicas de codificación más simples y más usadas. Es un tipo de cifrado por sustitución en el que una letra en el texto original es reemplazada por otra letra que se encuentra un número fijo de posiciones más adelante en el alfabeto. Por ejemplo, con un desplazamiento de 3, la A sería sustituida por la D (situada 3 lugares a la derecha de la A), la B sería reemplazada por la E, etc.

Se pide:

- Una función que realice el cifrado de un texto para un desplazamiento n dado.
- Una función que realice el descifrado del texto para un n dado. (es lo mismo que codificar con -n)

```
def codificar(texto,n)
    texto = texto.downcase! if texto != texto.downcase
    result = ''
    alphabet = ("a".."z").to_a
    texto.each_char do |letter|
          indixe = alphabet.index{|x| x.match /#{letter}/}
          result << alphabet[indixe + n]
    end
    puts result
end
```

12. Fecha Pascuas

Diseñar una función que retorne la fecha de pascuas.

Actualmente se utiliza el algoritmo Computus (Buscar información sobre el algoritmo) que calcula la fecha de cuando sería pascuas para un determinado año.

Para usar el tipo de dato Date hay que "importarlo" se hace mediante el comando: require 'date'

Por ejemplo:

Para el Año	La Fecha de Pascuas sería
2016	27/03/2016
2017	16/04/2017
2018	01/04/2017

```
require 'date'

# Utilizando el algoritmo de Butcher
def easter_date_for_year( year )
  A = year%19; B = year/100; C = year%100
  D = B / 4; E = B % 4; F = ( 8+B ) / 25
  G = ( B - F + 1 ) / 3; H = ( 19*A + B - D - G + 15 ) / 30
  I = C / 4; K = C % 4; L = (32 + 2E + 2I - H - K) % 7;
  M = (A + 11H + 22L) / 451; N = H + L - 7M + 114;
  return Date.new( year, N / 31, 1 + N % 31 );
end

if __FILE__==$0 then
  puts "Ingrese el año del cual quiere la fecha de pascuas"
  puts "Domingo de resurrección sería en la fecha ", easter_date_for_year( gets.to_i
).strftime( "%d.%B.%Y" );
end
```

Todos los problemas han sido obtenidos de la web

http://www.solveet.com/exercises