Taller Repaso General Ruby

- A continuación te presentamos una recopilación de ejercicios para repasar los conceptos aprendidos.
- Crea una carpeta para cada tipo de ejercicio(Arrays, Hashes, Objetos) y guarda cada archivo .rb con el número de la pregunta de manera local con **Sublime** o **Atom**.
- Luego guarda los cambios y súbelos a tu repositorio de Github.
- Luego de pusheados los últimos cambios, sube el link de Github en el desafío de la sección correspondiente en la plataforma.

Ejercicios de Arrays

Ejercicio 1: Array Básicos

Dado el array `[1,2,3,9,1,4,5,2,3,6,6]`

1. Mostrar en pantalla el primer elemento.
2. Mostrar en pantalla el último elemento.
3. Mostrar en pantalla todos los elementos.
4. Mostrar en pantalla todos los elementos junto con un índice.
5. Mostrar en pantalla todos los elementos que se encuentren en una posición par.
6. Determinar con el método aprendido en clases si un elemento ingresando pertenece al array o no, mostrar el resultado en pantalla

Ejercicio 2: Operaciones de push y pop en arrays

Dado el array `[1,2,3,9,1,4,5,2,3,6,6]`

1. Eliminar el último elemento.
2. Eliminar el primer elemento.
3. Eliminar el elemento que se encuentra en la posición media, si el arreglo tiene un número par de elementos entonces remover el que se encuentre en la mitad izquierda, ejemplo, en el arreglo [1,2,3,4] se removería el elemento 2.
4. Borrar el último elemento mientras ese número sea distinto a 1.
5. Utilizando un arreglo vacío auxiliar y operaciones de push and pop invertir el orden de los elementos en un arreglo.

Ejercicio 3: Array, operaciones numéricas y métodos.

Dado un array como el siguiente `[1,2,3,9,1,4,5,2,3,6,6]`:

1. Crear un método para eliminar todos los números pares del arreglo.
2. Crear un método para obtener la suma de todos los elementos del arreglo Utilizando .each
3. Crear un método para obtener el promedio de un arreglo.
4. Crear un método que incrementa todos los elementos en una unidad y devuelva un arreglo nuevo.

Ejercicio 4: Arrays y strings

Dado un arreglo con nombres como el siguiente ["Violeta", "Andino", "Clemente", "Javiera", "Paula", "Pia", "Ray"].

1. Extraer todos los elementos que excedan mas de 5 caracteres.
2. Crear un arreglo nuevo con todos los elementos en minúscula
3. Crear un método que devuelva un arreglo con la cantidad de caracteres que tiene cada nombre.

Ejercicio 5: Iteración en múltiples arrays

Dado los siguientes arreglos

```
a = [1,2,3]
b = [:azul, :rojo, :amarillo]
c = ["Tacos", "Quesadillas", "Hamburguesas"]
```

Se pide iterar sobre los arreglos para mostrar la información de la siguiente forma:

```
1 :azul, Tacos
2 :rojo, Quesadillas
3 :amarillo, Hamburguesas
```

Se pide iterar sobre los arreglos para mostrar la información de la siguiente forma:

```
1 :amarillo, Tacos
2 :rojo, Quesadillas
3 :azul, Hamburguesas
```

Iterando los arreglos anteriores crear un único arreglo final con:

```
[1, :azul, Tacos, 2, :rojo, Quesadillas, 3,
:amarillo, "Hamburguesas"]
```

Ejercicio 6: Manipulación de múltiples arrays

Se tienen dos arreglos

El primero es del tipo [1,2,3,0,1,2,2,1,2,1,2,0,3] y el segundo es del tipo [:azul, :verde, :amarillo]

Se pide cambiar todas las apariciones del número que aparece en el arreglo 1 por el elemento con el mismo índice del arreglo 2, en caso de no existir el elemento deberá ser remplazado por nil.

El resultado de este ejercicio debería quedar:

```
[:verde, :amarillo, nil, :azul, :verde, :amarillo ....]
```

Ejercicio 7: Operaciones de conjunto con 2 arrays

Dado los arrays:

```
a = [1,2,3,9,12,31, "domingo"]
b = ["lunes", "martes", "miércoles", "jueves", "viernes", "sábado", "domingo"]
```

1. Se pide obtener la concatenación de ambos

   ```
   [1,2,3,9,
   12,31, "domingo", "lunes",
   "martes", "miércoles", "jueves",
   "viernes", "sábado", "domingo"]
   ```

   En la concatenación los elementos que están en 1 y en 2 aparecen dos veces en los resultados.

2. Se pide obtener la unión de ambos, o sea:

   ```
   [1,2,3,9,
   12,31, "domingo", "lunes",
   "martes", "miércoles", "jueves",
   "viernes", "sábado"]
   ```

   En la unión de dos conjuntos los elementos que se repiten tanto en el 1º como en el segundo no se muestran dos veces en el resultado final.

3. Se pide obtener la intersección de ambos.

```
["domingo"]
```

   En la intersección solo resultan los elementos que se encuentran en ambos conjuntos.

4. Se pide agrupar en pares los elementos de cada conjunto.

```
[[1, "lunes"], [2, "martes"], [3, "miércoles"],
[9, "jueves"], [12, "viernes"], [31, "sábado"],
["domingo", "domingo"]]
```

Pregunta 8

Se tiene un ejercicio con una cantidad par de elementos en un arreglo, se pide agruparlos de a dos.

Ejemplo: [1,2,3,4,5,6,7,8] Resultado: [[1,2],[3,4],[5,6],[7,8]]

Hint: `each_slice`

Pregunta 9

Se tiene un string del tipo "1,2,7,1,3,5,3.4,9,1"

Se pide:

- Calcular el promedio
- Calcular la moda (o sea mostrar el número que más se repite)

Guía de ejercicios hashes

Ejercicio 1: Introducción a Hashes

Dado el hash = {"x": 1, "y":2}

- Agregar el texto z con el valor 3
- Cambiar el valor de x por 5
- Eliminar la clave y
- Si el hash tiene una clave llamada z mostrar en pantalla "yeeah"
- Invertir el diccionario de forma que los valores sean las llaves y las llaves los valores

- ejemplo: x = {"a":"hola" } se transforme en {"hola": "a"}

Ejercicio 2: Operaciones típicas sobre un hash

Escribir un hash con el menu de un restaurant, la llave es el nombre del plato y el valor es el precio de este.

```
restaurant_menu = { "Ramen" => 3, "Dal Makhani" => 4, "Coffee" => 2 }
```

1. Obtener el plato mas caro
2. Obtener el plato mas barato
3. Sacar el promedio del valor de los platos
4. Crear un arreglo con solo los nombres de los platos
5. Crear un arreglo con solo los valores de los platos
6. Modificar el hash y agregar el IVA a los valores de los platos (multiplicar por 1.19).
7. Dar descuento del 20% para los platos que tengan un nombre de mas 1 una palabra.

Ejercicio 3: Ejercicio completo con un Hash

Se tiene un hash con el inventario de un negocio de computadores.

```
inventario = {"Notebooks": 4, "PC Escritorio": 6, "Routers": 10, "Impresoras": 6}
```

Se pide:

- Crear un menú de 4 opciones, o sea el usuario puede ingresar, 1, 2, 3 o 4 y según eso el programa realizará distintas funciones.
- Si el usuario ingresa 1, podrá ingresar un item y su stock en un solo string y agregarlo al hash, para separar el nombre del stock el usuario debe utilizar una coma.
    - Ejemplo del input "Pendrives, 100"
    - Después de ingresar el valor
- Si el usuario igresa 2, podrá ver el stock total (suma del stock de cada item) que hay en el negocio
- Si el usuario ingresa 3 podrá ver el ítem que tiene la mayor cantidad de stock
- Si el usuario ingresa 4 podrá ingresar preguntarle al sistema si un item existe en el inventario o no, por ejemplo el usuario ingresará "Notebooks" y el programa responderá "si"
- El programa debe repertirse hasta que el usuario ingrese 5

Ejercicio 4: Array y Hashes

Se tienen dos arrays uno con el nombre de personas y otro con las edades, se pide generar un hash con el nombre y edad de cada persona (se asume que no existen dos personas con el mismo nombre)

```
personas = ["Carolina", "Alejandro", "Maria Jesús", "Valentín"]
edad = [32, 28, 41, 19]
```

- Se pide generar un hash con la información

  ```
  personas_hash = {"Carolina": 32, "Alejandro":28,
  "María Jesús":41, "Valentín":19}
  ```

- Crear un método que reciba el hash y devuelva la edad del hash pasado como argumento.

Ejercicio 5: Array de hashes

Crear un arreglo de hashes, cada hash contiene un array con datos de persona, estos datos son:

- Nombre
- País
- Continente
- Género

1. Crear el array de hashes y pobarlo con al menos 8 personas:
2. Contar la cantidad de personas de la lista.
3. Generar un array con cada continente, eliminar repeticiones, considerar que pueden haber nombres escritos con mayúscula y minúscula.
4. Generar una lista con todas las personas llamadas pedro
5. Hacer dos listas de personas, una por cada género6. Armar un hash, donde cada clave sea un continente y el value un array con los países de cada continente.

Ejercicios de Objetos

Ejercicio 01

Se tiene el siguiente código que no funciona, para arreglarlo se pide que el método1 funcione como método de instancia.

```
class T
    def method1()
    end
end

T.method1
```

Ejercicio 02

Se tiene el siguiente código que no funciona, para arreglarlo se pide que método1 funcione como método de clase.

```
class T
    def method1()
    end
end

T.method1
```

Ejercicio 04

Se tienen las siguientes clases:

```
class T
    def method1
    end
end

class Q
end
```

Se pide que:

- Q herede de T
- Q al hacer inicializado llame a method1
- method1 debe devolver un número al azar
- El método devuelto debe ser guardado en una variable de instancia de Q

Ejercicio 05

Dado el siguiente código

```
class Car
    @@t_instances = 0
    @@q_instances = 0
end

class T
end

class Q
end
```

Modificar T, Q y Car para que Car pueda contar las instancias respectivas de T y Q, además crear métodos dentro de car para obtener el número de instancias de cada uno.

Para probar Crear 20 instancias de T y 25 de Q.