**Trabajo práctico nro. 12**

|  |  |  |
| --- | --- | --- |
| Logo Departamento 2.jpg | **Asignatura: Laboratorio I** | |
|  | |
| **Cursado:**Primer Trimestre | **Horas** **semanales**: |
|  | **Horas semestrales:**  *Cantidad estimada de horas semestrales/anuales.* |
| **Carrera**: *Tecnicatura Universitaria en Programación* | **Nivel (Año):** |
| **Ciclo Lectivo: 2023** |

1. **Tema:**

**Expresiones Regulares**

1. **Enunciados:**

**Empezaremos por unos ejercicios básicos de programas Java con estructura secuencial, es decir, en estos programas no hay instrucciones condicionales ni repetitivas. En la mayoría de ellos las operaciones a realizar son: lectura de datos por teclado, realizar alguna operación con esos datos y mostrar resultados por pantalla.**

**Recomendaciones: Leer la teoría, investigar, consultar si existen dudas y realizar el trabajo en clases. La entrega de este trabajo practico es individual.**

1. **Investigación de Expresiones Regulares**:
   * Investiga y comprende los conceptos básicos de las expresiones regulares, incluyendo caracteres especiales, cuantificadores y grupos.
   * Explora ejemplos de patrones comunes de expresiones regulares utilizados en la validación de datos.
2. **Creación de un Conjunto de Datos**:
   * Crea un conjunto de datos ficticio que contenga información variada, como direcciones de correo electrónico, números de teléfono, fechas, códigos postales, etc.
3. **Validación de Datos**:
   * Diseña un programa en un lenguaje de programación (Java) que utilice expresiones regulares para validar y verificar los datos en tu conjunto de datos. Por ejemplo, puedes verificar si una dirección de correo electrónico es válida o si un número de teléfono sigue un formato específico.
   * Validar entrada de datos en un arraylist y aplicarlo.
4. **Informe de Investigación**:
   * Prepara un informe que desarrolle el tema sobre las expresiones regulares y cómo se aplicaron en la validación de datos.
   * Incluye ejemplos de patrones de expresiones regulares utilizados en tu programa y cómo ayudaron en la validación de los datos.
   * Comprensión de los conceptos de expresiones regulares.
   * Diseño y desarrollo del programa de validación de datos.
   * Claridad y organización del informe de investigación.
5. **Ejercicios**:

* Comprobar si el String *cadena* no acaba con un dígito.

public static void main(String[] args) {  
 System.*out*.println("String cadena no acaba con un dígito.");  
  
 System.*out*.println("Paula: "+*validar*("Paula"));  
  
 System.*out*.println("Payaso\*: "+*validar*("Payaso\*"));  
  
 System.*out*.println("1234: "+*validar*("1234"));  
  
 System.*out*.println("Colon325: "+*validar*("Colon325"));  
 }  
 public static boolean validar(String palabra){  
 return !palabra.matches("^.\*\\s\*\\d$");  
}

* Comprobar si el String *cadena* no empieza por un dígito.

public static void main(String[] args) {  
 System.*out*.println("El String cadena no empieza por un dígito.");  
  
 System.*out*.println("Paula Geier: "+*validar*("Paula"));  
  
 System.*out*.println("Payaso\*: "+*validar*("Payaso\*"));  
  
 System.*out*.println("-Serena: "+*validar*("-Serena"));  
  
 System.*out*.println("1234: "+*validar*("1234"));  
  
 System.*out*.println("Colon325: "+*validar*("Colon325"));  
  
 System.*out*.println("5 Naranjas: "+*validar*("5 Naranjas"));  
 }  
 public static boolean validar(String palabra){  
 return palabra.matches("^[^0-9].\*$");  
}

* Comprobar si el String *cadena* está formado por un mínimo de 5 letras mayúsculas o minúsculas y un máximo de 10.

public static void main(String[] args) {  
 System.*out*.println("Formado por un mínimo de 5 letras mayúsculas o minúsculas y un máximo de 10:");  
  
 System.*out*.println("Paula GEIER: "+*validar*("Paula"));  
  
 System.*out*.println("PERRO :"+*validar*("PERRO"));  
  
 System.*out*.println("verano :"+*validar*("verano"));  
  
 System.*out*.println("sol :"+*validar*("sol"));  
  
 }  
 public static boolean validar(String palabra){  
 return palabra.matches("^([a-zA-Z\\s]){5,10}$");  
}

* Validar el ingreso de un DNI (xx-xxx-xxx).

public static void main(String[] args) {  
 System.*out*.println("Validar el ingreso de un DNI (xx-xxx-xxx).:");  
  
 System.*out*.println("44-593-085: "+*validar*("44-593-085"));  
  
 System.*out*.println("44-593-05: "+*validar*("44-593-05"));  
  
 System.*out*.println("44593085: "+*validar*("44593085"));  
  
 }  
 public static boolean validar(String palabra){  
 return palabra.matches("^\\d{2}[-]\\d{3}[-]\\d{3}$");  
}

* Guardar en un arraylist una serie de mails, pero antes de ser guardados deben

cumplir con las condiciones de las expresiones regulares.

import java.util.ArrayList;  
import java.util.Arrays;  
  
public class validar {  
 public static void main(String[] args) {  
 System.*out*.println("Validar mail:");  
 ArrayList<String> numeros = new ArrayList<String>(Arrays.*asList*("paulageier@mail.com","rodrigozapata@mail.com","thomasmuñoz@mail.com","vacalola@mail.","lospalmeras@mail.com.ar"));  
 for (int i=0;i<numeros.size();i++){  
 System.*out*.println(numeros.get(i)+": "+*validarMail*(numeros.get(i)));  
 }  
 }  
 public static boolean validarMail(String palabra){  
  
 return palabra.matches("^.+@{1}.\*\\.com{1}.?[a-z]\*$");  
}  
}

* Comprobar si el String *cadena* contiene “abc”.

public static void main(String[] args) {  
 System.*out*.println("Contenga abc:");  
  
 System.*out*.println("banco: " + *validar*("banco"));  
  
 System.*out*.println("bencida: " + *validar*("bencida"));  
  
 System.*out*.println("vaso: " + *validar*("vaso"));  
  
 System.*out*.println("carton: " + *validar*("carton"));  
  
}  
  
public static boolean validar(String palabra) {  
  
 return palabra.matches("^(?=.\*a)(?=.\*b)(?=.\*c).\*$");  
  
}