Tarea 2 del RA5
Expresiones regulares

Crea un programa con un menú que valide varias expresiones regulares utilizando la clase **Pattern** y **Matcher**.

Las opciones del menú a validar serán:

1. Validar sólo números. (Se valida si la cadena introducida sólo tiene números).

```
//Regex
Pattern pattern = Pattern.compile( regex: "\\d+");
```

2. Validar sólo letras. (Se valida si la cadena introducida sólo tiene letras).

```
//Regex
Pattern pattern = Pattern.compile( regex: "[a-zA-Z]+");
```

3. Validar si es un nombre de teléfono español. (Con la forma +34 y a continuación 9 dígitos).

```
//Regex
Pattern pattern = Pattern
        .compile( regex: "\\+34\\d{9}");
```

4. Validar un email correcto.

```
//Regex
Pattern pattern = Pattern
        .compile( regex: "[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\\.[A-Z|a-z]{2,}");
```

5. Validar si la cadena tiene espacios.

```
//Regex
Pattern pattern = Pattern.compile( regex: "\\s");
//Input para comprobar
Matcher matcher = pattern.matcher(input);
//Devolvemos si hay coincidencia
if (matcher.find()) {
    return true;
} else {
    return false;
}
```

6. Validar un DNI español. (8 dígitos y a continuación una letra que puede ser minúsculas o mayúsculas). Hay que tener en cuenta que las letras I, Ñ, O, U no existen el DNI español.

7. Validar un *password*. Se debe comprobar lo siguiente:
    a. Al menos tiene un número.
    b. Tiene una letra mayúscula.
    c. Una letra minúscula.
    d. Tiene un carácter especial.
    e. Longitud mayor o igual que 10.

    Si cumple todas estas restricciones, entonces es un password válido.
8. Salir.

**** Añade las siguientes validaciones:

- Número estándar de teléfono con código de país.
- Dirección IPv4.
- El *password* no puede tener más de 20 dígitos.

El programa deberá tener las siguientes clases:

- App
  - Se encargará de mostrar un menú al usuario con todas las opciones y pedir al usuario las cadenas a validar.
- Banner
  - Clase con dos métodos estáticos que devolverán un String con el mensaje de bienvenida o el mensaje de despedida.
    - public static String getBannerInit(String name): devuelve un mensaje de bienvenida de la aplicación con el nombre del usuario.
    - public static String getBannerClose(): devuelve el mensaje de despedida.
- InputValidator
  - Clase con métodos estáticos que validan las cadenas pasadas.
    - public static boolean checkIfNumber(String input)
    - public static boolean checkIfLetters(String input)
    - public static boolean checkIfPhone(String input)
    - public static boolean checkIfEmail(String input)
    - public static boolean checkIfSpace(String input)
    - public static boolean checkSpanishDNI(String input)
    - public static boolean checkPassword(String input)

Ejemplo de salida:

```
##########################################
               Welcome to 'Input Checker'
               by Joaquín Franco
##########################################


Choose one of the next input validation checkers
or press 'q' to close the program:
        1) Validate only numbers.
        2) Validate only letters.
        3) Validate is a Spanish phone number.
        4) Validate is an e-mail.
        5) Validate it contains spaces.
        6) Validate it is a Spanish DNI.
        7) Check password security.
        q) Close program.


1
Please, introduce a string with numbers only:
adfa4
The input wasn't only numbers!!
```

```
Choose one of the next input validation checkers
or press 'q' to close the program:
        1) Validate only numbers.
        2) Validate only letters.
        3) Validate is a Spanish phone number.
        4) Validate is an e-mail.
        5) Validate it contains spaces.
        6) Validate it is a Spanish DNI.
        7) Check password security.
        q) Close program.


1
Please, introduce a string with numbers only:
234566
¡Great, it has only numbers!
```