TRABAJO TEÓRICO PARTE 1

Tercer Problema. Personas

Hemos realizado el testing del tercer problema, derivado del código de nuestros compañeros.

```java
package com.trabajoTeorico2.persona;

import java.time.LocalDate;

public class Persona {

    @SuppressWarnings("unused")
    private String nombre;
    @SuppressWarnings("unused")
    private String apellidos;
    private LocalDate fechaNacimiento;
    private String nacionalidad;
    private String titulacion;
    private boolean tieneCertificadoIngles;
    private String numeroTelefono;
    private String correoElectronico;

    public Persona() {
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }


    public void setApellidos(String apellidos) {
        this.apellidos = apellidos;
    }


    public void setFechaNacimiento(LocalDate fechaNacimiento) {
        this.fechaNacimiento = fechaNacimiento;
    }


    public void setNacionalidad(String nacionalidad) {
        this.nacionalidad = nacionalidad;
    }

    public void setTitulacion(String titulacion) {
        this.titulacion = titulacion;
    }

    public boolean isTieneCertificadoIngles() {
        return tieneCertificadoIngles;
    }

    public void setTieneCertificadoIngles(boolean tieneCertificadoIngles) {
        this.tieneCertificadoIngles = tieneCertificadoIngles;
    }
```

```java
    public void setNumeroTelefono(String numeroTelefono) {
        this.numeroTelefono = numeroTelefono;
    }


    public void setCorreoElectronico(String correoElectronico) {
        this.correoElectronico = correoElectronico;
    }

    public boolean esElegibleParaMatricularse() {
        if (LocalDate.now().getYear() - fechaNacimiento.getYear() < 18) {
            return false;
        }
        if (titulacion == null || !(titulacion.equalsIgnoreCase("Máster") || titulacion.equalsIgnoreCase("Doctorado"))) {
            return false;
        }

        if (!"España".equalsIgnoreCase(nacionalidad) && !"Francia".equalsIgnoreCase(nacionalidad) && !"Alemania".equalsIgnoreCase(nacionalidad)) {
            return false;
        }

        if (numeroTelefono == null || !numeroTelefono.matches("\\d{9}")) {
            return false;
        }

        if (correoElectronico == null || !correoElectronico.matches("^[a-zA-Z0-9_+&*-]+(?:\\.[a-zA-Z0-9_+&*-]+)*@(?:[a-zA-Z0-9-]+\\.)+[a-zA-Z]{2,7}$")) {
            return false;
        }

        return true;
    }
}
```

2) Identificar las variables que se deben tener en cuenta para probar el método de interés.

1. Edad:
    - Uso en el método: Se calcula con LocalDate.now().getYear() - fechaNacimiento.getYear() y se verifica si es menor de 18.
    - Variable relevante: fechaNacimiento.
2. Titulación:
    - Uso en el método: Se verifica si es "Máster" o "Doctorado".
    - Variable relevante: titulacion.
3. Nacionalidad:
    - Uso en el método: Se verifica si es "España", "Francia" o "Alemania".
    - Variable relevante: nacionalidad.
4. Número de Teléfono:
    - Uso en el método: Validado con una expresión regular \\d{9}.
    - Variable relevante: numeroTelefono.
5. Correo Electrónico:
    - Uso en el método: Validado con una expresión regular.
    - Variable relevante: correoElectronico.

3) Identificar los valores de pruebas para cada una de las variables anteriores usando las tres técnicas vistas en teoría, especificando para cada una cual es la que ha sido usada.

Edad:

Técnica aplicada: Valores límite.
Dominio de entrada: Edad calculada a partir de fechaNacimiento.
Valores seleccionados:
Menores de edad: 17 (límite inferior).
Justo en el límite: 18.
Adultos jóvenes: 25 (límite superior).
Más allá del límite: 26.
Valores finales: {17, 18, 25, 26}.

Titulación:

Técnica aplicada: Clases de Equivalencia.
Dominio de entrada: String.
Clases válidas:
"Máster".
"Doctorado".
Clases inválidas:
"Licenciatura" (otra titulación).
null (valor nulo).
"" (cadena vacía).
"master" (variación en minúsculas).
Valores finales: {"Máster", "Doctorado", "Licenciatura", null, "", "master"}.

Nacionalidad:

Técnica aplicada: Clases de Equivalencia y Conjetura de Errores.
Dominio de entrada: String.
Clases válidas:

"España".
"Francia".
"Alemania".

Clases inválidas:

"Italia" (otra nacionalidad).
null (valor nulo).
"" (cadena vacía).
"espana" (variación en minúsculas).

Valores finales: {"España", "Francia", "Alemania", "Italia", null, "", "espana"}.

Número de Teléfono:

Técnica aplicada: Clases de Equivalencia y Conjetura de Errores.
Dominio de entrada: String.
Clases válidas:

Cadena de 9 dígitos: "123456789".

Clases inválidas:

null (valor nulo).
"" (cadena vacía).
"12345" (longitud corta).
"1234567890" (longitud larga).
"abcdefghij" (caracteres no numéricos).

Valores finales: {"123456789", null, "", "12345", "1234567890", "abcdefghij"}.

Correo Electrónico:

Técnica aplicada: Clases de Equivalencia y Conjetura de Errores.
Dominio de entrada: String.
Clases válidas:

Formato estándar: "test@example.com".
Variaciones válidas: "user.name+tag@domain.co".

Clases inválidas:

null (valor nulo).
"" (cadena vacía).
"invalid-email@" (sin dominio).
"@example.com" (sin usuario).
"user@.com" (sin nombre de dominio).

Valores finales: {"test@example.com", "user.name+tag@domain.co", null, "", "invalid-email@", "@example.com", "user@.com"}.

Variable	Técnica Aplicada	Valores de Prueba Seleccionados
Edad	Valores Frontera	{17, 18, 25, 26}
Titulación	Clases de Equivalencia	{"Máster", "Doctorado", "Licenciatura", null, "", "master"}
Nacionalidad	Clases de Equivalencia	{"España", "Francia", "Alemania", "Italia", null, "", "espana"}

Número Teléfono	Conjetura de Errores	{"123456789", null, "", "12345", "1234567890", "abcdefghij"}
Correo Electrónico	Conjetura de Errores	{"test@example.com", "user.name+tag@domain.co", null, "", "invalid-email@", "@example.com", "user@.com"}

4) Calcular el número máximo posible de casos de pruebas que se podrían generar a partir de los valores de pruebas (combinatoria).

Variable	Valores de Prueba	Cantidad
Edad	{17, 18, 25, 26}	4
Titulación	{"Máster", "Doctorado", "Licenciatura", null, "", "master"}	6
Nacionalidad	{"España", "Francia", "Alemania", "Italia", null, "", "espana"}	7
Número Teléfono	{"123456789", null, "", "12345", "1234567890", "abcdefghij"}	6
Correo Electrónico	{"test@example.com", "user.name+tag@domain.co", null, "", "invalid-email@", "@example.com", "user@.com"}	7

Cálculo del número máximo de casos de prueba (Combinatoria)

Totales = 4×6×7×6×7 = 7056

5) Defina un conjunto de casos de pruebas para cumplir con each use (cada valor una vez)

Valores seleccionados por variable:

Variable	Valores de Prueba
Edad	{17, 18, 25, 26}
Titulación	{"Máster", "Doctorado", "Licenciatura", null, "", "master"}
Nacionalidad	{"España", "Francia", "Alemania", "Italia", null, "", "espana"}
Número Teléfono	{"123456789", null, "", "12345", "1234567890", "abcdefghij"}
Correo Electrónico	{"test@example.com", "user.name+tag@domain.co", null, "", "invalid-email@", "@example.com", "user@.com"}

Conjunto de casos de prueba para *each use*

Cada valor se incluye al menos una vez en los siguientes casos de prueba:

Caso de Prueba	Edad	Titulación	Nacionalidad	Número Teléfono	Correo Electrónico
Caso 1	17	Máster	España	123456789	test@example.com
Caso 2	18	Doctorado	Francia	null	user.name+tag@domain.co
Caso 3	25	Licenciatura	Alemania	""	null
Caso 4	26	null	Italia	12345	""
Caso 5	18	""	null	1234567890	invalid-email@
Caso 6	17	master	espana	abcdefghij	@example.com
Caso 7	25	Máster	Francia	12345	user@.com

6) Defina conjuntos de pruebas para alcanzar cobertura pairwaise usando el algoritmo explicado en clase. Se pueden comprobar los resultados con el programa PICT2

Caso	Edad	Titulación	Nacionalidad	Número Teléfono	Correo Electrónico
1	17	Máster	España	123456789	test@example.com
2	18	Doctorado	Francia	null	user.name+tag@domain.co
3	25	Licenciatura	Alemania	""	invalid-email@
4	26	null	Italia	12345	@example.com
5	17	""	null	1234567890	user@.com
6	18	master	espana	abcdefghij	null
7	25	Máster	Francia	""	""
8	26	Doctorado	España	null	invalid-email@
9	18	null	Alemania	12345	test@example.com
10	17	master	Italia	abcdefghij	user.name+tag@domain.co
11	26	Máster	null	""	@example.com
12	25	Doctorado	Francia	null	test@example.com

13	18	Licenciatura	España	1234567890	user@.com
14	17	""	espana	abcdefghij	""
15	26	master	Alemania	12345	null

PAIR-WASE = 15

7) Para los trozos de código que incluyan decisiones, proponga conjunto de casos de prueba para alcanzar cobertura de decisiones

if (LocalDate.now().getYear() - fechaNacimiento.getYear() < 18)

Condiciones	Decisión	Condición Dominante
True (edad < 18)	True	A
False	False	A

if (titulacion == null || !(titulacion.equalsIgnoreCase("Máster") || titulacion.equalsIgnoreCase("Doctorado")))

Condiciones	Decisión	Condición Dominante
True (A: titulacion == null)	True	A
False & True (B: !Máster)	True	B
False & False (C: !Doctorado)	True	C
False & False & False	False	A, B, C

if (!"España".equalsIgnoreCase(nacionalidad) && !"Francia".equalsIgnoreCase(nacionalidad) && !"Alemania".equalsIgnoreCase(nacionalidad))

Condiciones	Decisión	Condición Dominante
True (no España) & True (no Francia) & True (no Alemania)	True	A, B, C
True (no España) & True (no Francia) & False (es Alemania)	False	C
True (no España) & False (es Francia) & -	False	B
False (es España) & - & -	False	A

if (numeroTelefono == null || !numeroTelefono.matches("\\d{9}"))

Condiciones	Decisión	Condición Dominante
True (A: numeroTelefono == null)	True	A
Talse & True (B: no cumple regex)	True	B
Talse & False	False	A, B

if (correoElectronico == null || !correoElectronico.matches("regex"))

Condiciones	Decisión	Condición Dominante
True (A: correoElectronico == null)	True	A
False & True (B: no cumple regex)	True	B
False & False	False	A, B

8) Para los trozos de código que incluyan decisiones, proponga conjunto de casos de prueba para alcanzar cobertura MC/DC.

if (LocalDate.now().getYear() - fechaNacimiento.getYear() < 18)

Condición (A: Edad < 18)	Decisión	Comentarios
True	True	Edad menor a 18
False	False	Edad mayor o igual a 18

if (titulacion == null || !(titulacion.equalsIgnoreCase("Máster") ||
titulacion.equalsIgnoreCase("Doctorado")))

Condición	**A**: titulacion == null	**B**: !titulacion.equals ("Máster")	**C**: !titulacion.equals ("Doctorado")	Decisión	Condición Dominante
Caso 1	True	-	-	True	
Caso 2	False	True	-	True	A

Caso 3	False	False	True	True	B
Caso 4	False	False	False	False	A, B

if (!"España".equalsIgnoreCase(nacionalidad) && !"Francia".equalsIgnoreCase(nacionalidad) && !"Alemania".equalsIgnoreCase(nacionalidad))

Condición	**A**: no España	**B**: no Francia	**C**: no Alemania	Decisión	Condición Dominante
Caso 1	True	True	True	True	A, B, C
Caso 2	True	True	False	False	C
Caso 3	True	False	-	False	B
Caso 4	False	-	-	False	A

if (numeroTelefono == null || !numeroTelefono.matches("\\d{9}"))

Condición	**A**: numeroTelefono == null	**B**: !numeroTelefono.matches("\d{9}")	Decisión	Condición Dominante
Caso 1	True	-	True	A
Caso 2	False	True	True	B
Caso 3	False	False	False	A, B

if (correoElectronico == null || !correoElectronico.matches("regex"))

Condición	**A**: correoElectronico == null	**B**: !correoElectronico.matches ("regex")	Decisión	Condición Dominante
Caso 1	True	-	True	A
Caso 2	False	True	True	B
Caso 3	False	False	False	A, B

9) Comente los resultados del número de los casos de pruebas conseguidos en los apartados 4, 5 y 6 ¿qué podría decirse de la cobertura alcanzada?

La combinatoria completa (7056 casos) ofrece la cobertura más exhaustiva pero es impráctica en términos de tiempo y recursos.

La técnica each use (8 casos) es eficiente pero limitada, ya que no cubre interacciones entre valores.

La pairwise (19 casos) es la opción más eficiente y práctica, equilibrando cobertura e inversión de recursos al asegurar las combinaciones de pares de valores.

En este contexto, la técnica pairwise proporciona la mejor cobertura posible con un número razonable de casos de prueba, detectando defectos en combinaciones clave sin incurrir en la sobrecarga de la combinatoria completa.