EJERCICIO TEÓRICO. TESTING PROBLEMA 2.

TRABAJO TEÓRICO ISO II

PAREJA 3. DAVID DIONISIO LAUREANO Y SEBASTIÁN MEDINA SÁNCHEZ

1. **Escribir, al menos el pseudocódigo correspondiente al método o a los métodos identificados.**

```
public static boolean viajaDuranteElCurso(Cliente c){ //Quizás lo podemos quitar
        return c.viajaUnaVezAlMesDuranteCurso();
    }

    public static Tarifa calcularTarifa(Cliente c, Viaje v) {

        int edad = c.getEdad();
        int frecuencia = c.getFrecuenciaViajes();
        tipoBillete clase = c.getPreferenciaClase();
        String destino = c.getDestinoPreferido();
        int ingresos = c.getIngresos();
        boolean trabaja = c.getTrabaja();
        boolean independizado = c.getIndependizado();
        String paisDestino = v.getPaisDestino();

        // 1) Menor de edad → Pajarillo
        if (edad < 18) {
            if (frecuencia >= 6) {
                return new Tarifa("Pajarillo", 0.10);
            }
        }

        // 2) 18–25 años, estudiante en otra ciudad, viajando en turista
        if (edad >= 18 && edad <= 25 && !trabaja) {
            if (viajaDuranteElCurso(c)) {
                return new Tarifa("Gorrión", 0.15);
            }
        }

        // 3) 18–25, trabaja
        if (edad >= 18 && edad <= 25 && trabaja) {

            if (clase == tipoBillete.TURISTA && frecuencia >= 3) {
                if (!independizado) {
                    return new Tarifa("Viaja ahora que puedes", 0.05);
                } else {
```

```
                    return new Tarifa("Atreviéndose a saltar del Nido", 0.25);
                }
            }
        }

        // 4) >25 años, ingresos entre 20k y 35k, 6 viajes/año en turista dentro de Europa
        if (edad > 25 && ingresos > 20000 && ingresos < 35000) {
            if (clase == tipoBillete.TURISTA && frecuencia >= 6 &&
paisDestino.equalsIgnoreCase("europa")) {
                if (v.isConNinos()) {
                    return new Tarifa("Conoce Europa con tus peques", 0.10);
                } else {
                    return new Tarifa("Conoce Europa", 0.15);
                }
            }
        }

        // 5) >25 años, ingresos >35k, 6 viajes/año en business a Asia o América
        if (edad > 25 && ingresos > 35000) {
            if (clase == tipoBillete.BUSINESS && frecuencia >= 6 &&
                (paisDestino.equalsIgnoreCase("asia") ||
paisDestino.equalsIgnoreCase("america"))) {

                if (v.isConNinos()) {
                    return new Tarifa("Conoce el Mundo con tus peques", 0.10);
                } else {
                    return new Tarifa("Conoce el Mundo", 0.20);
                }
            }
        }

        // Si no cae en ninguno de los casos:
        return null;
    }
```

2. **Identificar las unidades de prueba.**

La unidad de prueba principal es la clase encargada de la lógica de negocio, en este caso la clase Tarifa, en concreto el método calcularTarifa(), ya que encapsula todas las reglas de decisión para asignar la tarifa y su descuento. Es el núcleo funcional que debe validarse.

Además, hay unidades auxiliares que contienen pequeña lógica como la clase Cliente, método viajaUnaVezAlMesDuranteCurso() que valida si un cliente cumple el patrón de viajes del curso. Otro sería en la clase Viaje, método parse_fecha() que agrupa la lógica de parseo y validación de fechas.

3. **Identificar las variables que se deben tener en cuenta para probar los métodos de interés incluidos en las unidades de prueba.**

Para probar correctamente los métodos identificados, debemos desglosar los objetos de entrada (Cliente y Viaje) en las **variables individuales** que realmente afectan al flujo de control (los if y else). A continuación, se identifican las variables clasificadas por su origen:

1. Variables provenientes del objeto Cliente

Estas variables determinan el perfil del usuario y son las que más ramificaciones crean en el código.

- **edad** (int):
    - *Origen:* c.getEdad()
    - *Importancia:* variable crítica principal. Divide el flujo en tres grandes bloques: Menores (<18), Jóvenes (18-25) y Adultos (>25).
- **frecuencia** (int):
    - *Origen:* c.getFrecuenciaViajes()
    - *Importancia:* usada en casi todas las ramas para filtrar si el cliente viaja lo suficiente para merecer descuento (umbrales de >= 6 o >= 3).
- **clase** (tipoBillete):
    - *Origen:* c.getPreferenciaClase()
    - *Importancia:* distingue entre tarifas de jóvenes trabajadores y tarifas de adultos (TURISTA vs BUSINESS).
- **ingresos** (int):
    - *Origen:* c.getIngresos()
    - *Importancia:* fundamental para la lógica de adultos (>25 años). Define los rangos de 20k-35k y >35k.
- **trabaja** (boolean):
    - *Origen:* c.getTrabaja()
    - *Importancia:* distingue entre la tarifa "Gorrión" (estudiante) y las de jóvenes trabajadores.
- **independizado** (boolean):
    - *Origen:* c.getIndependizado()
    - *Importancia:* decide el tipo de tarifa para jóvenes trabajadores ("Viaja ahora..." vs "Atreviéndose a saltar...").
- **viajaDuranteCurso** (boolean):
    - *Origen:* retorno del método viajaDuranteElCurso(c) (que a su vez llama a c.viajaUnaVezAlMesDuranteCurso()).
    - *Importancia:* única condición para aplicar la tarifa "Gorrión".

2. Variables provenientes del objeto Viaje

Estas variables contextualizan el viaje específico para ver si aplica a la tarifa del perfil.

- **paisDestino** (String):
    - *Origen:* v.getPaisDestino()
    - *Importancia:* filtra geográficamente. Se comprueba si es "Europa", "Asia" o "America", por ejemplo (ignorando mayúsculas/minúsculas).
- **conNinos** (boolean):
    - *Origen:* v.isConNinos()
    - *Importancia:* determina la subtarifa familiar ("... con tus peques") en los casos de adultos.

3. Variables Identificadas pero irrelevantes

Es importante identificar variables que se extraen en el código, pero **no afectan a la prueba**.

- **destino** (String):
    - *Origen:* c.getDestinoPreferido()
    - *Razón:* en el código aparece la línea String destino = c.getDestinoPreferido();, pero esa variable destino **jamás se usa** dentro de las condiciones if. Por lo tanto, no es necesario crear casos de prueba variando este valor (es irrelevante/código muerto a efectos lógicos).

4. Identificar los valores de pruebas para cada una de las variables anteriores que conformarán los correspondientes casos de prueba usando las tres técnicas vistas en teoría (clases de equivalencia, teoría de los valores límites y conjetura de errores), especificando para cada valor obtenido cuál es la que ha sido usada.

Variable	Clases de Equivalencia	Valores Seleccionados	Límites	Conjetura de errores
edad	(-∞, 0], [1, 17], [18, 25], [26, +∞)	[-2,16 (menor),22 (joven),30 (adulto)]	{0,1,17,26,18,25}	{2*10^3,”e”,”,”,”“ }
frecuencia	(-∞, -1], [0, 2], [3, 5], [6, +∞]	[-5,2 (baja),4 (media),7 (alta)]	{2,3,5,6}	{2*10^3,”e”,”,”,”“ }
ingresos	(-∞, -1], [0, 20000], [20001, 34999], [35000],[35001, +∞]	[-3,19000, 25000,35000,40000]	{20000,20001,35000,35001}	{-2^25,2^25,”e”,”,”,”“}
clase	"TURISTA”, “BUSINESS”	[TURISTA, BUSINESS, “ “]	{TURISTA, BUSINESS}	{null,” “}
paisDestino	"Europa”, “Asia”, “America”, “Africa”	[Europa, Asia, America, Africa, “ ”]	{Europa, Asia, America, Africa}	{null,” “}
trabaja	True; false	[True; false]	{True; false}	
independizado	True; false	[True; false]	{True; false}	
conNinos	True; false	[True; false]	{True; false}	

viajaDuranteElCurso	True; false	[True; false]	{True; false}	

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

EACH_USE

Cada valor válido de cada variable debe ser utilizado en al menos un caso de prueba. El número mínimo de casos de prueba está determinado simplemente por la variable que tiene más valores. Las demás variables pueden "rellenar" estos casos repitiendo sus valores. Variable con más valores: edad (14 valores).

Resultado EACH-USE: 14 casos de prueba. *(Con 14 pruebas podemos probar todos los valores de edad, y como sobran filas para las demás variables, cubrimos todos los ingresos, frecuencias, etc.)*

PAIR_WISE

Cada par posible de valores de cualesquiera dos variables debe aparecer junto al menos una vez. El número mínimo de pruebas necesarias es, al menos, el producto de los números de valores de las dos variables con mayores dimensiones. Variable 1 (Mayor): edad (14 valores). Variable 2 (Segunda Mayor): ingresos (13 valores). Total = 14 * 13 = 182

Resultado PAIR-WISE: 182 casos de prueba (mínimo).

N-WISE

Se prueban todas las combinaciones posibles de todas las variables. En este caso, N = 9 (el número total de variables). Se deberán multiplicar el número de valores de todas las variables del programa. En nuestro caso, será de 14 * 13 * 11 * 6 * 4 * 2 * 2 * 2 * 2 = 768.768

Resultado N-WISE: 768.768 casos de prueba.

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

Para cumplir con el criterio Each-Use (1-Wise), debemos crear un conjunto mínimo de casos de prueba donde cada valor único identificado en tu tabla anterior aparezca al menos una vez. El número de casos de prueba lo determina la variable con más valores únicos, que en este caso es edad con 14 valores.

ID Caso	edad	frecuencia	ingresos	clase	paisDestino	trabaja	independiente	conNinos	viaja
CP01	-2	-5	-3	"TURISTA"	"Europa"	True	True	True	True
CP02	16	2	19000	"BUSINESS"	"Asia"	False	False	False	False
CP03	22	4	25000	" "	"America"	True	True	True	True
CP04	30	7	35000	null	"Africa"	False	False	False	False
CP05	0	3	40000	TURISTA	" "	True	True	True	True
CP06	1	5	20000	BUSINESS	null	False	False	False	False
CP07	17	6	20001	TURISTA	Europa	True	True	True	True

CP08	26	2000	35001	BUSINESS	Asia	False	False	False	False
CP09	18	"e"	-2^25	TURISTA	America	True	True	True	True
CP10	25	","	2^25	BUSINESS	Africa	False	False	False	False
CP11	2000	""	"e"	TURISTA	Europa	True	True	True	True
CP12	"e"	2	","	BUSINESS	Asia	False	False	False	False
CP13	","	4	""	TURISTA	America	True	True	True	True
CP14	""	7	25000	BUSINESS	Europa	False	False	False	False

7. Defina conjuntos de pruebas para alcanzar cobertura pairwaise usando el algoritmo explicado en clase. Se pueden comprobar los resultados con el programa PICT5

ID	edad	frecuencia	ingresos	clase	paisDestino	trabaja	independiente	conNinos	viajaDuranteElCurso
1	-2	-5	-3	TURISTA	Europa	True	True	True	True
2	-2	2	19000	BUSINESS	Asia	False	False	False	False
3	-2	4	25000		America	True	True	True	True
4	-2	7	35000	null	Africa	False	False	False	False
5	-2	3	40000	TURISTA		True	True	True	True
6	-2	5	20000	BUSINESS	null	False	False	False	False
7	-2	6	20001		Europa	True	True	True	True
8	-2	2000	35001	null	Asia	False	False	False	False
9	-2	e	-33554432	TURISTA	America	True	True	True	True
10	-2	,	2^25	BUSINESS	Africa	False	False	False	False
11	-2		e			True	True	True	True
12	-2	-5	,	null	null	False	False	False	False
13	-2	2		TURISTA	Europa	True	True	True	True
14	16	4	-3	BUSINESS	Asia	False	False	False	False
15	16	7	19000		America	True	True	True	True
16	16	3	25000	null	Africa	False	False	False	False
17	16	5	35000	TURISTA		True	True	True	True
18	16	6	40000	BUSINESS	null	False	False	False	False
19	16	2000	20000		Europa	True	True	True	True
20	16	e	20001	null	Asia	False	False	False	False
21	16	,	35001	TURISTA	America	True	True	True	True
22	16		-33554432	BUSINESS	Africa	False	False	False	False
23	16	-5	2^25			True	True	True	True
24	16	2	e	null	null	False	False	False	False

25	16	4	,	TURISTA	Europa	True	True	True	True
26	16	7		BUSINESS	Asia	False	False	False	False
27	22	3	-3		America	True	True	True	True
28	22	5	19000	null	Africa	False	False	False	False
29	22	6	25000	TURISTA		True	True	True	True
30	22	2000	35000	BUSINESS	null	False	False	False	False
31	22	e	40000		Europa	True	True	True	True
32	22	,	20000	null	Asia	False	False	False	False
33	22		20001	TURISTA	America	True	True	True	True
34	22	-5	35001	BUSINESS	Africa	False	False	False	False
35	22	2	-33554432			True	True	True	True
36	22	4	2^25	null	null	False	False	False	False
37	22	7	e	TURISTA	Europa	True	True	True	True
38	22	3	,	BUSINESS	Asia	False	False	False	False
39	22	5			America	True	True	True	True
40	30	6	-3	null	Africa	False	False	False	False
41	30	2000	19000	TURISTA		True	True	True	True
42	30	e	25000	BUSINESS	null	False	False	False	False
43	30	,	35000		Europa	True	True	True	True
44	30		40000	null	Asia	False	False	False	False
45	30	-5	20000	TURISTA	America	True	True	True	True
46	30	2	20001	BUSINESS	Africa	False	False	False	False
47	30	4	35001			True	True	True	True
48	30	7	-33554432	null	null	False	False	False	False
49	30	3	2^25	TURISTA	Europa	True	True	True	True
50	30	5	e	BUSINESS	Asia	False	False	False	False
51	30	6	,		America	True	True	True	True
52	30	2000		null	Africa	False	False	False	False
53	0	e	-3	TURISTA		True	True	True	True
54	0	,	19000	BUSINESS	null	False	False	False	False
55	0		25000		Europa	True	True	True	True
56	0	-5	35000	null	Asia	False	False	False	False
57	0	2	40000	TURISTA	America	True	True	True	True

58	0	4	20000	BUSINESS	Africa	False	False	False	False
59	0	7	20001			True	True	True	True
60	0	3	35001	null	null	False	False	False	False
61	0	5	-33554432	TURISTA	Europa	True	True	True	True
62	0	6	2^25	BUSINESS	Asia	False	False	False	False
63	0	2000	e		America	True	True	True	True
64	0	e	,	null	Africa	False	False	False	False
65	0	,		TURISTA		True	True	True	True
66	1		-3	BUSINESS	null	False	False	False	False
67	1	-5	19000		Europa	True	True	True	True
68	1	2	25000	null	Asia	False	False	False	False
69	1	4	35000	TURISTA	America	True	True	True	True
70	1	7	40000	BUSINESS	Africa	False	False	False	False
71	1	3	20000			True	True	True	True
72	1	5	20001	null	null	False	False	False	False
73	1	6	35001	TURISTA	Europa	True	True	True	True
74	1	2000	-33554432	BUSINESS	Asia	False	False	False	False
75	1	e	2^25		America	True	True	True	True
76	1	,	e	null	Africa	False	False	False	False
77	1		,	TURISTA		True	True	True	True
78	1	-5		BUSINESS	null	False	False	False	False
79	17	2	-3		Europa	True	True	True	True
80	17	4	19000	null	Asia	False	False	False	False
81	17	7	25000	TURISTA	America	True	True	True	True
82	17	3	35000	BUSINESS	Africa	False	False	False	False
83	17	5	40000			True	True	True	True
84	17	6	20000	null	null	False	False	False	False
85	17	2000	20001	TURISTA	Europa	True	True	True	True
86	17	e	35001	BUSINESS	Asia	False	False	False	False
87	17	,	-33554432		America	True	True	True	True
88	17		2^25	null	Africa	False	False	False	False
89	17	-5	e	TURISTA		True	True	True	True
90	17	2	,	BUSINESS	null	False	False	False	False

91	17	4			Europa	True	True	True	True
92	26	7	-3	null	Asia	False	False	False	False
93	26	3	19000	TURISTA	America	True	True	True	True
94	26	5	25000	BUSINESS	Africa	False	False	False	False
95	26	6	35000			True	True	True	True
96	26	2000	40000	null	null	False	False	False	False
97	26	e	20000	TURISTA	Europa	True	True	True	True
98	26	,	20001	BUSINESS	Asia	False	False	False	False
99	26		35001		America	True	True	True	True
100	26	-5	-33554432	null	Africa	False	False	False	False
101	26	2	2^25	TURISTA		True	True	True	True
102	26	4	e	BUSINESS	null	False	False	False	False
103	26	7	,		Europa	True	True	True	True
104	26	3		null	Asia	False	False	False	False
105	18	5	-3	TURISTA	America	True	True	True	True
106	18	6	19000	BUSINESS	Africa	False	False	False	False
107	18	2000	25000			True	True	True	True
108	18	e	35000	null	null	False	False	False	False
109	18	,	40000	TURISTA	Europa	True	True	True	True
110	18		20000	BUSINESS	Asia	False	False	False	False
111	18	-5	20001		America	True	True	True	True
112	18	2	35001	null	Africa	False	False	False	False
113	18	4	-33554432	TURISTA		True	True	True	True
114	18	7	2^25	BUSINESS	null	False	False	False	False
115	18	3	e		Europa	True	True	True	True
116	18	5	,	null	Asia	False	False	False	False
117	18	6		TURISTA	America	True	True	True	True
118	25	2000	-3	BUSINESS	Africa	False	False	False	False

119	25	e	19000			True	True	True	True
120	25	,	25000	null	null	False	False	False	False
121	25		35000	TURISTA	Europa	True	True	True	True
122	25	-5	40000	BUSINESS	Asia	False	False	False	False
123	25	2	20000		America	True	True	True	True
124	25	4	20001	null	Africa	False	False	False	False
125	25	7	35001	TURISTA		True	True	True	True
126	25	3	-33554432	BUSINESS	null	False	False	False	False
127	25	5	2^25		Europa	True	True	True	True
128	25	6	e	null	Asia	False	False	False	False
129	25	2000	,	TURISTA	America	True	True	True	True
130	25	e		BUSINESS	Africa	False	False	False	False
131	2000	,	-3			True	True	True	True
132	2000		19000	null	null	False	False	False	False
133	2000	-5	25000	TURISTA	Europa	True	True	True	True
134	2000	2	35000	BUSINESS	Asia	False	False	False	False
135	2000	4	40000		America	True	True	True	True
136	2000	7	20000	null	Africa	False	False	False	False
137	2000	3	20001	TURISTA		True	True	True	True
138	2000	5	35001	BUSINESS	null	False	False	False	False
139	2000	6	-33554432		Europa	True	True	True	True
140	2000	2000	2^25	null	Asia	False	False	False	False
141	2000	e	e	TURISTA	America	True	True	True	True
142	2000	,	,	BUSINESS	Africa	False	False	False	False
143	2000					True	True	True	True
144	e	-5	-3	null	null	False	False	False	False

145	e	2	19000	TURISTA	Europa	True	True	True	True
146	e	4	25000	BUSINESS	Asia	False	False	False	False
147	e	7	35000		America	True	True	True	True
148	e	3	40000	null	Africa	False	False	False	False
149	e	5	20000	TURISTA		True	True	True	True
150	e	6	20001	BUSINESS	null	False	False	False	False
151	e	2000	35001		Europa	True	True	True	True
152	e	e	-33554432	null	Asia	False	False	False	False
153	e	,	2^25	TURISTA	America	True	True	True	True
154	e			e	BUSINESS	Africa	False	False	False
155	e	-5	,			True	True	True	True
156	e	2		null	null	False	False	False	False
157	,	4	-3	TURISTA	Europa	True	True	True	True
158	,	7	19000	BUSINESS	Asia	False	False	False	False
159	,	3	25000		America	True	True	True	True
160	,	5	35000	null	Africa	False	False	False	False
161	,	6	40000	TURISTA		True	True	True	True
162	,	2000	20000	BUSINESS	null	False	False	False	False
163	,	e	20001		Europa	True	True	True	True
164	,	,	35001	null	Asia	False	False	False	False
165	,		-33554432	TURISTA	America	True	True	True	True
166	,	-5	2^25	BUSINESS	Africa	False	False	False	False
167	,	2	e			True	True	True	True
168	,	4	,	null	null	False	False	False	False
169	,	7		TURISTA	Europa	True	True	True	True
170		3	-3	BUSINESS	Asia	False	False	False	False

171		5	19000		America	True	True	True	True
172		6	25000	null	Africa	False	False	False	False
173		2000	35000	TURISTA		True	True	True	True
174		e	40000	BUSINESS	null	False	False	False	False
175		,	20000		Europa	True	True	True	True
176			20001	null	Asia	False	False	False	False
177		-5	35001	TURISTA	America	True	True	True	True
178		2	-33554432	BUSINESS	Africa	False	False	False	False
179		4	2^25			True	True	True	True
180		7	e	null	null	False	False	False	False
181		3	,	TURISTA	Europa	True	True	True	True
182		5		BUSINESS	Asia	False	False	False	False

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

El objetivo es asegurar que cada decisión lógica (if) del código tome al menos una vez el valor Verdadero y una vez el valor Falso.

BLOQUE 1: MENOR DE EDAD (“PAJARILLO”)

DECISIÓN 1.1 (Principal)

Código: if (edad < 18)

- A: edad < 18
- Decisión: A

A	Decisión
V	V
F	F

Selección de Casos de Prueba: usando un caso que haga que la decisión sea V y otro que la haga F

Caso	A (Edad)	Decisión
C. Verdadero	17	V (Entra a evaluar 1.2)
C. Falso	22	F (Salta al Bloque 2)

DECISIÓN 1.2 (Anidada)

Código: if (frecuencia >= 6)

- Dependencia: requiere Decisión 1.1 = V (Edad < 18)
- A: frecuencia >= 6
- Decisión: A

A	Decisión
V	V
F	F

Selección de Casos de Prueba: usando un caso que haga que la decisión sea V y otro que la haga F

Caso	A (Frecuencia)	Decisión
C. Verdadero	7	V (Return Pajarillo)
C. Falso	2	F (Sale del bloque)

BLOQUE 2: JOVEN ESTUDIANTE ("Gorrión")

DECISIÓN 2.1 (Principal)

Código: if (edad >= 18 && edad <= 25 && !trabaja)

- A: edad >= 18
- B: edad <= 25
- C: !trabaja
- Decisión: A ∧ B ∧ ¬C

A (>=18)	B (<=25)	¬ C (!trabaja)	Decisión
V	V	F	F
V	V	V	F
V	F	F	F
V	F	V	F
F	V	F	F
F	V	V	F
F	F	F	V
F	F	V	F

Selección de Casos de Prueba: usando un caso que haga que la decisión sea V y otro que la haga F

Caso	Valores (Edad, Trabaja)	Decisión
C. Verdadero	Edad: 20, Trabaja: False	V (Entra a evaluar anidada 2.2)
C. Falso	Edad: 20, Trabaja: True	F (Salta al Bloque 3)

DECISIÓN 2.2 (Anidada)

Código: if (viajaDuranteElCurso(c))

- Dependencia: Requiere Decisión 2.1 = V
- A: viajaDuranteElCurso(c)

- Decisión: A

A	Decisión
V	V
F	F

Selección de Casos de Prueba: usando un caso que haga que la decisión sea V y otro que la haga F

Caso	Valor (viajaEnCurso)	Decisión
C. Verdadero	True	V (Return Gorrión)
C. Falso	False	F (Sale del bloque)

BLOQUE 3: JOVEN TRABAJADOR

DECISIÓN 3.1 (Principal)

Código: if (edad >= 18 && edad <= 25 && trabaja)

- A: edad >= 18
- B: edad <= 25
- C: trabaja
- Decisión: A ∧ B ∧ C

A	B	C	Decisión
V	V	V	V
V	V	F	F
V	F	V	F
V	F	F	F
F	V	V	F
F	V	F	F
F	F	V	F
F	F	F	F

Selección de Casos de Prueba: usando un caso que haga que la decisión sea V y otro que la haga F

Caso	Valores (Edad, Trabaja)	Decisión
C. Verdadero	Edad: 22, Trabaja: True	V (Entra a evaluar anidada 3.2)
C. Falso	Edad: 30, Trabaja: True	F (Salta al Bloque 4)

DECISIÓN 3.2 (Anidada)

Código: if (clase == tipoBillete.TURISTA && frecuencia >= 3)

- Dependencia: Requiere Decisión 3.1 = V
- A: clase == TURISTA
- B: frecuencia >= 3
- Decisión: A ∧ B

A	B	Decisión
V	V	V
V	F	F

F	V	F
F	F	F

Selección de Casos de Prueba: usando un caso que haga que la decisión sea V y otro que la haga F

Caso	Valores (Clase, Frecuencia)	Decisión
C. Verdadero	TURISTA, 4	V (Entra a evaluar anidada 3.3)
C. Falso	BUSINESS, 4	F (Sale del bloque)

DECISIÓN 3.3 (Anidada – If/Else)

Código: if (!independizado)

- Dependencia: Requiere Decisión 3.2 = V
- A: !independizado
- Decisión: ¬A

-A (!independizado)	Decisión
V	V
F	F

Selección de Casos de Prueba: usando un caso que haga que la decisión sea V y otro que la haga F

Caso	Valor (Independizado)	Decisión
C. Verdadero	False (No independizado)	V (Return "Viaja ahora...")
C. Falso	True (Independizado)	F (Return "Atreviéndose a...")

BLOQUE 4: ADULTO INGRESOS MEDIOS (Europa)

DECISIÓN 4.1 (Principal)

Código: if (edad > 25 && ingresos > 20000 && ingresos < 35000)

- A: edad > 25
- B: ingresos > 20000
- C: ingresos < 35000
- Decisión: A ∧ B ∧ C

A	B	C	Decisión
V	V	V	V
V	V	F	F
V	F	V	F
V	F	F	F
F	V	V	F
F	V	F	F
F	F	V	F
F	F	F	F

Selección de Casos de Prueba: usando un caso que haga que la decisión sea V y otro que la haga F

Caso	Valores (Edad, Ingresos)	Decisión
C. Verdadero	Edad: 30, Ingresos: 25000	V (Entra a evaluar anidada 4.2)

C. Falso	Edad: 30, Ingresos: 15000	F (Salta al Bloque 5)

DECISIÓN 4.2 (Anidada)

Código: if (clase == TURISTA && frecuencia >= 6 && paisDestino...("europa"))

- Dependencia: Requiere Decisión 4.1 = V
- A: clase == TURISTA
- B: frecuencia >= 6
- C: pais == "Europa"
- Decisión: A ∧ B ∧ C

A	B	C	Decisión
V	V	V	V
V	V	F	F
V	F	V	F
V	F	F	F
F	V	V	F
F	V	F	F
F	F	V	F
F	F	F	F

Selección de Casos de Prueba: usando un caso que haga que la decisión sea V y otro que la haga F

Caso	Valores (Clase, Destino)	Decisión
C. Verdadero	TURISTA, "Europa"	V (Entra a evaluar anidada 4.3)
C. Falso	TURISTA, "Asia"	F (Sale del bloque)

DECISIÓN 4.3 (Anidada – If/Else)

Código: if (v.isConNinos())

- Dependencia: Requiere Decisión 4.2 = V
- A: isConNinos()
- Decisión: A

A	Decisión
V	V
F	F

Selección de Casos de Prueba: usando un caso que haga que la decisión sea V y otro que la haga F

Caso	Valor (conNinos)	Decisión
C. Verdadero	True	V (Return "Europa con peques")
C. Falso	False	F (Return "Conoce Europa")

BLOQUE 5: ADULTO INGRESOS ALTOS (Mundo)

DECISIÓN 5.1 (Principal)

Código: if (edad > 25 && ingresos > 35000)

- A: edad > 25
- B: ingresos > 35000
- Decisión: A ∧ B

A	B	Decisión
V	V	V
V	F	F
F	V	F
F	F	F

Selección de Casos de Prueba: usando un caso que haga que la decisión sea V y otro que la haga F

Caso	Valores (Edad, Ingresos)	Decisión
C. Verdadero	Edad: 30, Ingresos: 40000	V (Entra a evaluar anidada 5.2)
C. Falso	Edad: 30, Ingresos: 30000	F (Sale, retorna null)

DECISIÓN 5.2 (Anidada)

Código: if (clase == BUSINESS && frecuencia >= 6 && (pais == "Asia" || "America"))

- Dependencia: Requiere Decisión 5.1 = V
- A: clase == BUSINESS
- B: frecuencia >= 6
- C: pais == "Asia" || "America"
- Decisión: A ∧ B ∧ C

A	B	C	Decisión
V	V	V	V
V	V	F	F
V	F	V	F
V	F	F	F
F	V	V	F
F	V	F	F
F	F	V	F
F	F	F	F

Selección de Casos de Prueba: usando un caso que haga que la decisión sea V y otro que la haga F

Caso	Valores (Clase, Destino)	Decisión
C. Verdadero	BUSINESS, "Asia"	V (Entra a evaluar anidada 5.3)
C. Falso	TURISTA, "Asia"	F (Sale del bloque)

DECISIÓN 5.3 (Anidada – If/Else)

Código: if (v.isConNinos())

- Dependencia: Requiere Decisión 5.2 = V

- A: isConNinos()
- Decisión: A

A	Decisión
V	V
F	F

Selección de Casos de Prueba: usando un caso que haga que la decisión sea V y otro que la haga F

Caso	Valor (conNinos)	Decisión
C. Verdadero	True	V (Return "Mundo con peques")
C. Falso	False	F (Return "Conoce el Mundo")

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

DECISIÓN 1: JOVEN ESTUDIANTE

Código: if (edad >= 18 && edad <= 25 && !trabaja)

- A: edad >= 18
- B: edad <= 25
- C: trabaja (La lógica niega esta variable)
- Lógica: A ∧ B ∧ ¬C

A (>=18)	B (<=25)	¬ C (!trabaja)	Resultado	Cond. dominante	Valores
V	V	V	V	A, B, C	A=20, B=20, C=False
F	V	V	F	A	A=10, B=10, C=False
V	F	V	F	B	A=30, B=30, C=False
V	V	F	F	C	A=20, B=20, C=True

DECISIÓN 2: JOVEN TRABAJADOR

Código: if (edad >= 18 && edad <= 25 && trabaja)

- A: edad >= 18
- B: edad <= 25
- C: trabaja
- Lógica: A ∧ B ∧ C

A (>=18)	B (<=25)	C (trabaja)	Resultado	Cond. dominante	Valores
V	V	V	V	A, B, C	A=20, B=20, C=True
F	V	V	F	A	A=10, B=10, C=True
V	F	V	F	B	A=30, B=30, C=True
V	V	F	F	C	A=20, B=20, C=False

DECISIÓN 3: REQUISITOS TRABAJADOR

Código: if (clase == tipoBillete.TURISTA && frecuencia >= 3)

- A: clase == TURISTA

- B: frecuencia >= 3
- Lógica: A ∧ B

A (Turista)	B (Frec>=3)	Resultado	Cond. dominante	Valores
V	V	V	A, B	A=Turista, B=4
F	V	F	A	A=Business, B=4
V	F	F	B	A=Turista, B=2

DECISIÓN 4: ADULTO INGRESOS MEDIOS

Código: if (edad > 25 && ingresos > 20000 && ingresos < 35000)

- A: edad > 25
- B: ingresos > 20000
- C: ingresos < 35000
- Lógica: A ∧ B ∧ C

A (>25)	B (>20k)	C (<35k)	Resultado	Cond. dominante	Valores
V	V	V	V	A, B, C	A=30, B=25000, C=25000
F	V	V	F	A	A=20, B=25000, C=25000
V	F	V	F	B	A=30, B=15000, C=15000
V	V	F	F	C	A=30, B=40000, C=40000

DECISIÓN 5: REQUISITOS EUROPA

Código: if (clase == TURISTA && frecuencia >= 6 && paisDestino...("europa"))

- A: clase == TURISTA
- B: frecuencia >= 6
- C: pais == "Europa"
- Lógica: A ∧ B ∧ C

A (Turista)	B (Frec>=6)	C (Europa)	Resultado	Cond. dominante	Valores
V	V	V	V	A, B, C	A=Turista, B=7, C=Eur
F	V	V	F	A	A=Bus, B=7, C=Eur
V	F	V	F	B	A=Turista, B=2, C=Eur
V	V	F	F	C	A=Turista, B=7, C=Asia

DECISIÓN 6: ADULTO INGRESOS ALTOS

Código: if (edad > 25 && ingresos > 35000)

- A: edad > 25
- B: ingresos > 35000
- Lógica: A ∧ B

A (>25)	B (>35k)	Resultado	Dependencia	Valores
V	V	V	A, B	A=30, B=40000
F	V	F	A	A=20, B=40000
V	F	F	B	A=30, B=20000

DECISIÓN 7: REQUISITOS MUNDO

Código: if (clase == BUSINESS && frecuencia >= 6 && (pais...("asia") || pais...("america")))

- A: clase == BUSINESS
- B: frecuencia >= 6
- C: pais == "Asia"
- D: pais == "America"
- Lógica: A ∧ B ∧ (C V D)

Para satisfacer MC/DC en una expresión con OR (C V D), debemos demostrar que C afecta el resultado (manteniendo D falso) y que D afecta el resultado (manteniendo C falso).

A (Bus)	B (>=6)	C (Asia)	D (Amer)	Resultado	Cond. dominante	Valores
V	V	V	F	V	A, B, C	A=Bus, B=7, P=Asia
V	V	F	V	V	D	A=Bus, B=7, P=Amer
F	V	V	F	F	A	A=Tur, B=7, P=Asia
V	F	V	F	F	B	A=Bus, B=2, P=Asia
V	V	F	F	F	C, D	A=Bus, B=7, P=África

10. 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? ¿en qué medida la implementación del programa influye en el diseño e implementación de los casos de pruebas?

En el apartado cuatro, combinatoria (each-use / pair-wise / n-wise)

- Each-use (1-wise) con 14 casos da una cobertura muy baja en términos de interacción ya que garantiza que cada valor individual aparece al menos una vez, pero no garantiza que aparezcan combinaciones relevantes.
- Pair-wise (2-wise) con aproximadamente 182 casos mejora mucho porque cubre las interacciones por pares, lo cual suele capturar muchos fallos típicos (valores incompatibles, comparaciones mal puestas, etc). Aun así no garantiza cubrir decisiones que dependen de 3 o más condiciones simultáneas.
- N-wise (todas las combinaciones) con 768.768 casos es una "cobertura combinatoria total", pero normalmente inviable (tiempo, mantenimiento, coste). Además, muchas combinaciones son irreales o redundantes.

Por lo que en términos generales podríamos decir que:

- 1-wise → cubre *valores*, pero **no cubre reglas**.
- 2-wise → cubre muchas interacciones y suele ser "coste/beneficio" razonable.
- N-wise → cobertura total, pero **impracticable**.

Con el apartado cinco, aunque "matemáticamente" cumpla each-use de su tabla, no garantiza cobertura efectiva del código porque mezcla valores que no pertenecen al dominio tipado del método. En términos de lógica de negocio, la cobertura conseguida sería peor de lo que parece

En el apartado seis el conjunto de cobertura de decisiones es el más parecido al código real: diseñas casos para que cada if sea True y False al menos una vez. Eso implica que se cubren todas las ramas principales

(entrar/saltar bloques 1-5) y se cubren las anidadas Pero hay que tener en cuenta que la cobertura de decisiones no equivale a la cobertura de condiciones cuando hay expresiones con múltiples condiciones, por ejemplo, si (edad >= 18 && edad <= 25 && !trabaja) con decisión coverage basta con un caso True y uno False, pero no obliga a demostrar que cada sub-condición (A, B, C) puede afectar por sí sola al resultado.

Decision coverage es muy útil aquí, porque la lógica es una cascada de if con returns. Aun así, para expresiones compuestas, puede dejar huecos (sería mejor si lo complementas con valores límite y, si os lo piden, MC/DC)

De la cobertura total alcanzada podría decírse que:

- Each-use (por sí solo): cobertura superficial; asegura presencia de valores, pero no asegura ejercicio de reglas.
- Pair-wise: cobertura intermedia y práctica; detecta muchos fallos de interacción, aunque no garantiza activar reglas que dependen de 3+ condiciones.
- Cobertura de decisiones: cobertura directamente alineada con el código, muy adecuada para esta función porque cada regla termina en return. Es la que más sentido tiene para asegurar que todas las tarifas pueden salir y que también existe un caso que retorna null.

En resumen, lo que más "calidad" te da para este programa es Decision Coverage, y Pair-Wise como complemento para interacciones. Each-use queda como una verificación básica, pero no como evidencia fuerte.

La medida en que la implementación influye en el diseño y la propia implementación de los casos de prueba es alta, por varias razones:

1. Estructura "cascada" con returns tempranos
   El orden de los if importa: cuando una regla aplica, las siguientes ya no se evalúan.
   - Esto obliga a diseñar pruebas que eviten activar reglas anteriores si quieres alcanzar una regla posterior.
   - Ejemplo: para probar Bloque 4, tienes que asegurarte de no caer en Bloque 1–3 (edad>25 ya lo garantiza, bien). Para Bloque 5, además debes evitar que Bloque 4 aplique (ingresos > 35000 lo evita).
2. Condiciones compuestas (&&, ||)
   Como las decisiones dependen de varias variables, no basta con variar una sola.
   - Por eso pair-wise y decision coverage funcionan mejor que each-use.
3. Dependencia de métodos auxiliares (ej. viajaDuranteElCurso)
   La rama "Gorrión" depende completamente del valor devuelto por c.viajaUnaVezAlMesDuranteCurso(). Eso te fuerza a:
   - mockear/stubear Cliente (si hacéis unit testing puro), o
   - construir un Cliente real que devuelva True/False de forma controlada.
     La implementación concreta (si calcula con fechas, si depende de historial, etc.) cambia totalmente el esfuerzo del test.
4. Tipos y validación
   La implementación en Java con tipos fuertes hace que muchos valores de "conjetura de errores" (como "e" para int) no sean casos de prueba del método, sino casos de prueba del "input parsing/validación" *en otro sitio*.
   - Si el programa no valida y asume int correcto, esos "errores" no existen dentro de calcularTarifa().

5. Detalles concretos del código: comparación de strings paisDestino.equalsIgnoreCase("europa") implica que hay casos relevantes como "EUROPA", "Europa", etc., y también casos frontera como paisDestino = null que provocarían excepción si no se controla (aunque tú sí contemplas null en valores). Es decir: cómo está implementada la comparación te marca casos concretos.