UNIVERSIDAD DE CASTILLA-LA MANCHA
ESCUELA SUPERIOR DE INFORMÁTICA

TRABAJO TEÓRICO
EJERCICIO 2

María Victoria Alcázar Clemente
Alejandro Paniagua Rodríguez
Jesús Calzado González
Enrique Rubio Gonzalo
Clara Sacedón Ortega
Diego Alba Ruiz

Asignatura: Ingeniería del Software II

Grupo de Titulación (21/22): BC.02

Titulación: Grado en Ingeniería Informática

Fecha: 23-12-2021

ÍNDICE

PROBLEMA 1 4
PROBLEMA 2 22
PROBLEMA 3 34

FICHA DEL TRABAJO

Código:		Fecha:	23-12-2021
Título:	TRABAJO TEÓRICO – EJERCICIO 2		

Equipo Nº: BC.02		
Apellidos y Nombre	Firma	Puntos
MARÍA VICTORIA ALCÁZAR CLEMENTE (EJERC. 1)	20616826-V	16.67
CLARA SACEDÓN ORTEGA (EJERC. 1)	05739109-B	16.67
JESÚS CALZADO GONZÁLEZ (EJERC. 3)	05739349-K	16.67
ALEJANDRO PANIAGUA RODRÍGUEZ (EJERC. 3)	05720535-K	16.67
ENRIQUE RUBIO GONZALO (EJERC. 2)	02598786-Q	16.67
DIEGO ALBA RUIZ (EJERC. 2)	06286705-T	16.67

PROBLEMA 1

Se pretende desarrollar una aplicación que genere una recomendación sobre actividades lúdicas en función de las condiciones meteorológicas, del estado de salud de los usuarios y de las características de los espacios donde se puede disfrutar de las actividades de ocio. Para ello se tendrá en cuenta los siguientes aspectos:

- Si la persona está sana y sin síntomas, no ha estado en contacto en las dos últimas semanas con nadie infectado, ha pasado el COVID y tiene cartilla de vacunación (pasaporte COVID en regla) puede realizar cualquiera de las actividades propuestas. En otro caso no podrá realizar ninguna actividad.
- Si la temperatura meteorológica está por debajo de 0 grados, la humedad relativa es menor que 15%, y hay precipitaciones de nieve o de agua, entonces lo mejor es quedarse en casa.
- Si la temperatura meteorológica está por debajo de 0 grados, la humedad relativa es menor que 15%, y no hay precipitaciones de nieve o de agua, se puede ir a esquiar, si no se supera el aforo permitido por la legislación pertinente.
- Si la temperatura meteorológica está entre 0 y 15 grados, y no hay precipitaciones de agua, entonces es posible ir a hacer senderismo, si no se supera aforo del espacio previsto.
- Si la temperatura meteorológica está entre 15 y 25 grados, no llueve, y no está nublado y no hay una humedad relativa superior al 60%, entonces se puede ir a hacer turismo al aire libre, si la ciudad no tiene restricciones de confinamiento.
- Si la temperatura meteorológica está entre 25 y 35 grados, y no llueve, la recomendación es irse de cañas, si el establecimiento no tiene problemas de aforo.
- Si la temperatura meteorológica es mayor que 30 grados, y no llueve, la recomendación es irse a la playa o a la piscina. La piscina no puede superar el aforo permitido.

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

```
if (persona_sana && !sintomas && ! contacto_ ultimas2semanas && COVID_pasado &&
cartilla_vacunacion)
        actividadPermitida = "todas"
endif

if (temperatura < 0 && humedad < 15 && (nieve || agua))

        actividadPermitida = "ninguna"

else if (temperatura < 0 && humedad < 15 && !(nieve || agua) && !aforoPistaSuperado)

   actividadPermitida = "esquiar"

else if (temperatura >= 0 && temperatura < 15 && !agua && !aforoEspacioSuperado)

   actividadPermitida = "senderismo"

else if (temperatura >= 15 && temperatura < 25 && !agua && !nublado && humedad < 60 &&
!restricciones)

   actividadPermitida = "turismo"

else if (temperatura >= 25 && temperatura <= 35 && !agua && !aforoEstablecimientoSuperado)

   actividadPermitida = "cañas"

else if (temperatura > 30 && !agua && !aforoPiscinaSuperado)

   actividadPermitida = "piscina o playa"

else if (temperatura > 30 && temperatura <= 35 && !agua && ! aforoPiscinaSuperado &&
!aforoEstablecimientoSuperado)

   actividadPermitida = "piscina,playa o cañas"
endif

return actividadPermitida
```

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

Parámetros método “recomendaciónActividad” :

- persona_sana → boolean (true,false)
- sintomas → boolean (true,false)
- contacto_ ultimas2semanas → boolean (true,false)
- COVID_pasado → boolean (true,false)
- cartilla_vacunacion → boolean (true,false)
- temperatura → int (-infinito a +infinito)
- humedad → int (-infinito a +infinito)
- nieve → boolean (true,false)
- agua → boolean(true,false)
- nublado → boolean (true,false)
- aforoPistaSuperado → boolean (true,false)
- aforoEspacioSuperado → boolean (true,false)
- aforoEstablecimientoSuperado → boolean (true,false)
- aforoPiscinaSuperado → boolean (true,false)
- restricciones → boolean (true,false)

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.

PARAMETRO	CLASE DE EQUIVALENCIA	VALORES	VALORES LIMITE	CONJETURA DE ERRORES
persona_sana	True,false	True,false	True,false	null
sintomas	True,false	True,false	True,false	null
contacto_ ultimas2semanas	True,false	True,false	True,false	null
COVID_pasado	True,false	True,false	True,false	null
cartilla_vacunacion	True,false	True,false	True,false	null
temperatura	(-∞,0),[0,15),[15,25), [25,30], (30,35], (35, ∞)	-2,13,20,27,34,40	Variante ligera: 0,15,25,30,35 Variante Pesada: -1,1,14,16,24,26,29,31, 34,36 + ligeros	60
humedad	(-∞,0),[0,15), [15,60),[60, ∞)	-10,7,30,80	Variante ligera: 0,15,60 Variante Pesada: -1,1,14,16,59,61 + ligeros	90
nieve	True,false	True,false	True,false	Null
agua	True,false	True,false	True,false	null
nublado	True,false	True,false	True,false	null
aforoPistaSuperado	True,false	True,false	True,false	null
aforoEspacioSuperado	True,false	True,false	True,false	null
AforoEstablecimiento Superado	True,false	True,false	True,false	null
aforoPiscinaSuperado	True,false	True,false	True,false	null
restricciones	True,false	True,false	True,false	null

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

Combinaciones totales = 3 x 3 x 3 x 3 x 3 x 22 x 14 x 3 x 3 x 3 x 3 x 3 x 3x 3 x 3 = 491051484

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

Test suite 1 = { (true,true,true,true,true,-2,-10, true,true,true,true,true, true,true,null),
(false,true,null,true,true,13,7, true,true,true,true,true, true,true,true),
(true,false,true,true,true,20,30, true,true,true,null,true, true,true,true),
(null,false,false,true,true,27,80, true,true,true,true,true, true,true,true),
(true,true,false,false,null,34,0, true,true,true,true,true, true,true,true),
(false,true,false,false,false,40,15, true,true,null,true,true, true,true,true),
(true,null,true,false,false,0,60, false,true,true,true,true, true,true,true),
(false,false,true,false,false,15,-1, false,false,true,true,true, true,null,true),
(true,true,true,true,false,25,1, false,false,false,true,true, null,true,true),
(false,true,false,true,false,30,14, false,false,false,false,true, true,true,true),
(true,false,false,null,true,35,16, false,false,false,false,false, true,true,true),
(false,false,false,true,true,60,59, false,false,false,null,false, false,true,true),
(true,null,true,false,true,-1,61, true,false,false,false,false, false,false,true),
(false,true,null,false,true,1,90, true,false,false,false,false, false,false,false),
(null,false,true,false,true,14,-10, true,true,false,false,false, false,false,false),
(false,false,false,false,false,16,7, true,true,false,false,false, null,false,false),
(true,true,false,true,false,24,30, true,true,true,false,null, false,false,false),
(false,true,false,true,false,26,80, null,true,true,false,false, false,false,false),
(true,false,true,true,false,29,0, false,true,null,true,false, false,false,false),
(false,null,true,true,false,31,15, false,true,true,true,false, false,false,true,false),
(true,true,true,false,true,34,60, false,true,null,true,true, false,false,false),
(false,true,false,false,true,36,-1, null,false,true,true,true, false,false,false)}

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

CONJUNTO 1:

HUMEDAD	NIEVE
-10	True
-10	False
-10	Null
7	True
7	False
7	Null
30	True
30	False

30	Null
80	True
80	False
80	Null
0	True
0	False
0	Null
15	True
15	False
15	Null
60	True
60	False
60	Null
-1	True
-1	False
-1	Null
1	True
1	False
1	Null
14	True
14	False
14	Null
16	True
16	False
16	Null
59	True
59	False
59	Null
61	True
61	False
61	Null
90	True
90	False
90	Null

Serían 14 x 3 = 42 combinaciones.

CONJUNTO 2:

TEMPERATURA	AGUA
-2	True
-2	False
-2	Null
13	True
13	False
13	Null
20	True
20	False
20	Null
27	True
27	False
27	Null
34	True
34	False
34	Null
40	True
40	False
40	Null
0	True
0	False
0	Null
15	True
15	False
15	Null
25	True
25	False
25	Null
30	True
30	False
30	Null
35	True
35	False
35	Null
-1	True
-1	False

-1	Null
1	True
1	False
1	Null
14	True
14	False
14	Null
16	True
16	False
16	Null
24	True
24	False
24	Null
26	True
26	False
26	Null
29	True
29	False
29	Null
31	True
31	False
31	Null
34	True
34	False
34	Null
36	True
36	False
36	Null
60	True
60	False
60	Null

Serían 22 x 3 = 66 combinaciones.

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

Los casos de prueba que cumplen la cobertura de decisiones se muestran resaltados en verde.

PRIMERA DECISIÓN:

p: persona_sana
q: síntomas
r: contacto_ultimas2semanas
s: COVID_pasado
t: cartilla_vacunacion

p	q	r	s	t	¬q	¬r	p ∧ ¬q ∧ ¬r ∧ s ∧ t	Condición dominante
V	V	V	V	V	F	F	F	q,r
V	V	V	V	F	F	F	F	t,q,r
V	V	V	F	V	F	F	F	s,q,r
V	V	V	F	F	F	F	F	s,t,q,r,
V	V	F	V	V	F	V	F	r,q
V	V	F	V	F	F	V	F	r,t,q
V	V	F	F	V	F	V	F	r,s,q
V	V	F	F	F	F	V	F	r,s,t,q
V	F	V	V	V	V	F	F	q,r
V	F	V	V	F	V	F	F	q,t,r
V	F	V	F	V	V	F	F	q,s,r
V	F	V	F	F	V	F	F	q,s,t,r
V	F	F	V	V	V	V	V	q,r
V	F	F	V	F	V	V	F	q,r,t
V	F	F	F	V	V	V	F	q,r,s
V	F	F	F	F	V	V	F	q,r,s,t
F	V	V	V	V	F	F	F	p,q,r
F	V	V	V	F	F	F	F	p,t,q,r
F	V	V	F	V	F	F	F	p,s,q,r
F	V	V	F	F	F	F	F	p,s,t,q,r
F	V	F	V	V	F	V	F	p,r,q
F	V	F	V	F	F	V	F	p,r,t,q
F	V	F	F	V	F	V	F	p,r,s,q
F	V	F	F	F	F	V	F	p,r,s,t,q
F	F	V	V	V	V	F	F	p,q,r
F	F	V	V	F	V	F	F	p,q,t,r
F	F	V	F	V	V	F	F	p,q,s,r
F	F	V	F	F	V	F	F	p,q,s,t,r
F	F	F	V	V	V	V	F	p,q,r
F	F	F	V	F	V	V	F	s,q,r
F	F	F	F	V	V	V	F	p,q,r,s
F	F	F	F	F	V	V	F	p,q,r,s,t

SEGUNDA DECISIÓN:

p: temperatura < 0 r: nieve

q: humedad < 15 s: agua

p	q	r	s	r ∨ s	p ∧ q ∧ (r ∨ s)	Condición Dominante
V	V	V	V	V	V	p,q,r,s
V	V	V	F	V	V	p,q,r
V	V	F	V	V	V	p,q,s
V	V	F	F	F	F	r,s
V	F	V	V	V	F	q
V	F	V	F	V	F	q
V	F	F	V	V	F	q
V	F	F	F	F	F	q,r,s
F	V	V	V	V	F	p
F	V	V	F	V	F	p
F	V	F	V	V	F	p
F	V	F	F	F	F	p,r,s
F	F	V	V	V	F	p,q
F	F	V	F	V	F	p,q
F	F	F	V	V	F	p,q
F	F	F	F	F	F	p,q,r,s

TERCERA DECISIÓN:

p: temperatura < 0 r: nieve t: aforoPistaSuperado

q: humedad < 15 s: agua

p	q	r	s	t	(r ∨ s)	¬ (r ∨ s)	¬t	p ∧ q ∧ ¬ (r ∨ s) ∧ ¬t	Condición Dominante
V	V	V	V	V	V	F	F	F	R,s,t
V	V	V	V	F	V	F	V	F	R,s
V	V	V	F	V	V	F	F	F	R,s,t
V	V	V	F	F	V	F	V	F	R,s
V	V	F	V	V	V	F	F	F	R,s,t
V	V	F	V	F	V	F	V	F	R,s
V	V	F	F	V	F	V	F	F	R,s,t
V	V	F	F	F	F	V	V	V	P,q,r,s,t
V	F	V	V	V	V	F	F	F	Q,r,s,t
V	F	V	V	F	V	F	V	F	Q,r,s
V	F	V	F	V	V	F	F	F	Q,r,s,t
V	F	V	F	F	V	F	V	F	Q,r,s
V	F	F	V	V	V	F	F	F	Q,r,s,t
V	F	F	V	F	V	F	V	F	Q,r,s
V	F	F	F	V	F	V	F	F	q,t
V	F	F	F	F	F	V	V	F	Q
F	V	V	V	V	V	F	F	F	P,r,s,t
F	V	V	V	F	V	F	V	F	P,r,s
F	V	V	F	V	V	F	F	F	P,r,s,t
F	V	V	F	F	V	F	V	F	P,r,s
F	V	F	V	V	V	F	F	F	P,r,s,t
F	V	F	V	F	V	F	V	F	P,r,s
F	V	F	F	V	F	V	F	F	P,t
F	V	F	F	F	F	V	V	F	P
F	F	V	V	V	V	F	F	F	P,q,r,s,t
F	F	V	V	F	V	F	V	F	P,q,r,s
F	F	V	F	V	V	F	F	F	P,q,r,s,t
F	F	V	F	F	V	F	V	F	P,q,r,s
F	F	F	V	V	V	F	F	F	P,q,r,s,t
F	F	F	V	F	V	F	V	F	P,q,r,s
F	F	F	F	V	F	V	F	F	P,q,t
F	F	F	F	F	F	V	V	F	P,q

CUARTA DECISIÓN:

p: temperatura [0,15) q: agua r: aforoEspacioSuperado

p	q	r	¬q	¬r	p ∧ ¬q ∧ ¬r	Condición dominante
V	V	V	F	F	F	q,r
V	V	F	F	V	F	r
V	F	V	V	F	F	s
V	F	F	V	V	V	p,q,r
F	V	V	F	F	F	p,q,r
F	V	F	F	V	F	p,q
F	F	V	V	F	F	p,r
F	F	F	V	V	F	p

QUINTA DECISIÓN:

p: temperatura [15,25) r: nublado t: restricciones
q: agua s: humedad < 60

p	q	r	s	t	¬q	¬r	¬t	p ∧ ¬q ∧ ¬r ∧ s ∧ ¬t	Condición dominante
V	V	V	V	V	F	F	F	F	q,r,t
V	V	V	V	F	F	F	V	F	q,r
V	V	V	F	V	F	F	F	F	s,q,r,t
V	V	V	F	F	F	F	V	F	s,q,r
V	V	F	V	V	F	V	F	F	q,t
V	V	F	V	F	F	V	V	V	q
V	V	F	F	V	F	V	F	F	s,q,t
V	V	F	F	F	F	V	V	F	t,q
V	F	V	V	V	V	F	F	F	r,,t
V	F	V	V	F	V	F	V	F	r
V	F	V	F	V	V	F	F	F	s,r,,t
V	F	V	F	F	V	F	V	F	t,r
V	F	F	V	V	V	V	F	F	t
V	F	F	V	F	V	V	V	V	p,q,r,s,t
V	F	F	F	V	V	V	F	F	s,u
V	F	F	F	F	V	V	V	F	s
F	V	V	V	V	F	F	F	F	p,q,r,t
F	V	V	V	F	F	F	V	F	p,q,r
F	V	V	F	V	F	F	F	F	p,s,q,r,t
F	V	V	F	F	F	F	V	F	p,s,q,r
F	V	F	V	V	F	V	F	F	p,q,t
F	V	F	V	F	F	V	V	F	p,q
F	V	F	F	V	F	V	F	F	p,s,q,t
F	V	F	F	F	F	V	V	F	p,s
F	F	V	V	V	V	F	F	F	p,r,t
F	F	V	V	F	V	F	V	F	p,r
F	F	V	F	V	V	F	F	F	p,s,r,t
F	F	V	F	F	V	F	V	F	p,s,r
F	F	F	V	V	V	V	F	F	p,t
F	F	F	V	F	V	V	V	F	p
F	F	F	F	V	V	V	F	F	p,s,t
F	F	F	F	F	V	V	V	F	p,s

SEXTA DECISIÓN:

p: temperatura [25,35] r: aforoEstablecimientoSuperado

q: agua

p	q	r	¬q	¬r	p ∧ ¬q ∧ ¬r	Condición Dominante
V	V	V	F	F	F	q,s
V	V	F	F	V	F	q
V	F	V	V	F	F	r
V	F	F	V	V	V	p,q,r
F	V	V	F	F	F	p,q,r
F	V	F	F	V	F	p,q
F	F	V	V	F	F	p,r
F	F	F	V	V	F	p

SÉPTIMA DECISIÓN:

p: temperatura > 30 q: agua r: aforoPiscinaSuperado

p	q	r	¬q	¬r	p ∧ ¬q ∧ ¬r	Condición dominante
V	V	V	F	F	F	q,r
V	V	F	F	V	F	q
V	F	V	V	F	F	r
V	F	F	V	V	V	p,q,r
F	V	V	F	F	F	p,q,r
F	V	F	F	V	F	p,q
F	F	V	V	F	F	p,r
F	F	F	V	V	F	p

OCTAVA DECISIÓN:

p: temperatura (30,35]
q: agua
r: aforoPiscinaSuperado
s: aforoEstablecimientoSuperado

p	q	r	s	¬q	¬r	¬s	p ∧ ¬q ∧ ¬r ∧ ¬s	Condición dominante
V	V	V	V	F	F	F	F	q,r,s
V	V	V	F	F	F	V	F	q,r
V	V	F	V	F	V	F	F	q,s
V	V	F	F	F	V	V	F	q
V	F	V	V	V	F	F	F	r,s
V	F	V	F	V	F	V	F	r
V	F	F	V	V	V	F	F	s
V	F	F	F	V	V	V	V	p,q,r,s
F	V	V	V	F	F	F	F	p,q,r,s
F	V	V	F	F	F	V	F	p,q,r
F	V	F	V	F	V	F	F	p,q,s
F	V	F	F	F	V	V	F	p,q
F	F	V	V	V	F	F	F	p,r,s
F	F	V	F	V	F	V	F	p,r
F	F	F	V	V	V	F	F	p,s
F	F	F	F	V	V	V	F	p

Conjunto de casos de prueba que generan cobertura de decisiones:

persona_sana	sintomas	contacto_ultimas2semanas	COVID_pasado	cartilla_vacunación	temperatura	humedad
V	V	F	F	F	-	-
V	V	V	V	V	-	-
-	-	-	-	-	-1	14
-	-	-	-	-	-1	12
-	-	-	-	-	-2	7
-	-	-	-	-	-3	16
-	-	-	-	-	14	-
-	-	-	-	-	-1	-
-	-	-	-	-	16	56
-	-	-	-	-	17	62
-	-	-	-	-	27	-
-	-	-	-	-	14	-
-	-	-	-	-	34	-
-	-	-	-	-	45	-
-	-	-	-	-	32	-
-	-	-	-	-	25	-

nieve	agua	nublado	aforopistasuperado	aforoespaciosuperado	aforoestablecimientosuperado	aforopiscinasuperado	restricciones
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
F	V	-	-	-	-	-	-
F	F	-	-	-	-	-	-
F	F	-	V	-	-	-	-
V	V	-	F	-	-	-	-
-	V	-	-	V	-	-	-
-	F	-	-	F	-	-	-
-	F	V	-	-	-	-	V
-	F	V	-	-	-	-	F
-	V	-	-	-	V	-	-
-	F	-	-	-	F	-	-
-	V	-	-	-	-	F	-
-	V	-	-	-	-	V	-
-	V	-	-	-	V	V	-
-	F	-	-	-	F	F	-

Se pueden combinar las filas 1-4 y 2-3:

persona_sana	sintomas	contacto_ultimas2semanas	COVID_pasado	cartilla_vacunación	temperatura	humedad
V	V	F	F	F	-1	12
V	V	V	V	V	-1	14
-	-	-	-	-	-2	7
-	-	-	-	-	-3	16
-	-	-	-	-	14	-
-	-	-	-	-	-1	-
-	-	-	-	-	16	56
-	-	-	-	-	17	62
-	-	-	-	-	27	-
-	-	-	-	-	14	-
-	-	-	-	-	34	-
-	-	-	-	-	45	-
-	-	-	-	-	32	-
-	-	-	-	-	25	-

nieve	agua	nublado	aforopistasuperado	aforoespaciosuperado	aforoestablecimientosuperado	aforopiscinasuperado	restricciones
F	F	-	-	-	-	-	-
F	V	-	-	-	-	-	-
F	F	-	V	-	-	-	-
V	V	-	F	-	-	-	-
-	V	-	-	V	-	-	-
-	F	-	-	F	-	-	-
-	F	V	-	-	-	-	V
-	F	V	-	-	-	-	F
-	V	-	-	-	V	-	-
-	F	-	-	-	F	-	-
-	V	-	-	-	-	F	-
-	V	-	-	-	-	V	-
-	V	-	-	-	V	V	-
-	F	-	-	-	F	F	-

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

Las condiciones marcadas con sombreado amarillo en las tablas del ejercicio anterior cubren el criterio de condiciones y decisiones modificadas porque conseguimos que todos los valores sean dominantes.

La decisión se sigue evaluando a verdadero y falso al menos una vez.

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

Aunque con la que más casos de prueba se realizarían seria con el apartado 4,pensamos que con el apartado 6 obtendríamos una cobertura optima ya que se encuentra entre los tres criterios y consideramos que sería una cantidad realista de combinaciones probadas para este problema.

PROBLEMA 2

En su nuevo plan de transportes públicos de la JCCM, se nos ha pedido implementar una aplicación que determine el precio del billete del medio de transporte, en función de las condiciones de salud y etarias de una persona, y en función del estado de la pandemia. En este sentido, se vigilará la IA de Castilla-La Mancha (podemos suponerla "constante" durante la ejecución del proyecto, lo que requerirá una inicialización del entorno), con una reducción de plazas como sigue: si es menor que 100 no habrá restricciones de espacio (nivel 0), si está entre 100 y 200 (nivel 1) el aforo del medio de transporte se reduce al 80%, si está entre 201 y 300 (nivel 2), se reduce al 60%, si está entre 301 y 500 (nivel 3) el aforo se reduce al 40%, y si es superior a 501 (nivel 4) al 30%.

A fin de evitar movimientos innecesarios, se establecerá un incremento del precio del billete. Para ello se tendrá en cuenta las siguientes reglas para calcular el precio del billete:

• Independientemente del estado de la IA, una persona enferma, con contacto reciente en los últimos 10 días con infectados o con síntomas sospechosos de COVID no podrá viajar. Una persona con pasaporte COVID y no enferma podrá viajar si hay plaza, independientemente de su tipo de profesión.
• Si no hay restricciones de espacio (nivel 0), cualquier persona, independientemente de su edad podrá viajar, y tendrá un descuento del 60% si es menor que 23 años, y del 80% si es mayor que 65. No se establecen prioridades de transporte.
• En Nivel 1, los menores de 23 años tendrán un descuento de 30%, y los mayores de 65 tendrán un descuento de 50%. No se establecen prioridades de transporte, pero se reducen las plazas, con lo que solo se dará precio (plantéese lanzar algún tipo de excepción) si hay plazas disponibles.
• En Nivel 2, los menores de 23 años no tendrán descuentos, y los mayores de 65 años tendrá un incremento del 20%. En este nivel, de la capacidad disminuida posible, se reserva un 60% de las plazas a profesionales con profesiones imprescindibles.
• En Nivel 3, los menores de 23 años tendrán una recarga del 20%, y los mayores de 65 años tendrá un incremento del 50%. En este nivel de capacidad, se reserva un espacio disponible del 80% de la capacidad reducida a profesionales imprescindibles.
• En Nivel 4, los menores de 23 años tendrán una recarga del 50%, y los mayores de 65 años no podrán viajar. Se reserva un espacio del 90% para personas con profesiones imprescindibles.

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

NOTA: Devolverá el precio del billete calculado o en caso de no poder realizarse el viaje devolverá -1.

```
PuedeViajar=false
PrecioBillete=precioEstandar

If (CondicionSalud == noEnfermo && PasaporteCOVID == True &&
aforoDisponible>0)
        puedeViajar=true;
else
        puedeViajar= false;
endif

if(contagios<100 && edad<23)  // Nivel 0
        /*  Descuento de 60% */
        precioBillete = precioBillete * 0,4;
else if (contagios<100 && edad>65)
        /* Descuento del 80% */
        precioBillete = precioBillete *0,2;
endif


if(contagios>=100 && contagios<201 && edad<23) //Nivel 1

        aforoDisponible = aforoDisponible * 0,8;
        /* Descuento del 30% */
        precioBillete = precioBillete * 0,7;

else if (contagios>=100 && contagios<201 && edad>65)
        aforoDisponible = aforoDisponible * 0,8;
        /* Descuento del 50% */
        precioBillete = precioBillete * 0,5;
endif


if (contagios>=201 && contagios<301 && (esPrescindible && hayplazasprescindibles) ||
noesPrescindible=false && hayplazasprofesionales) &&edad>65 )//Nivel 2

                aforoDisponible = aforoDisponible * 0,6;
                plazasProfesionales = 0.6*aforoDisponible;
                plazasPrescindibles=0.4*aforoDisponible;

                /* Incremento del 20% */
                precioBillete = precioBillete * 1,2;
endif
```

```
if (contagios>=301 && contagios<=501 && (esPrescindible && hayplazasprescindibles) || noesPrescindible && hayplazasprofesionales) && edad<23)//Nivel 3

        aforoDisponible = aforoDisponible * 0,4;
        plazasProfesionales = 0.8*aforoDisponible;
        plazasPrescindibles=0.2*aforoDisponible;

        /* Incremento del 20% */
        precioBillete = precioBillete * 1,2;

else if (contagios>=301 && contagios<=501 && (esPrescindible && hayplazasprescindibles) || noesPrescindible && hayplazasprofesionales) && edad>65)

        aforoDisponible = aforoDisponible * 0,4;
        plazasProfesionales = 0.8*aforoDisponible;
        plazasPrescindibles=0.2*aforoDisponible;

        /* Incremento del 50% */
        precioBillete = precioBillete * 1,5;

endif

if(contagios>501 && (esPrescindible && hayplazasprescindibles) || noesPrescindible && hayplazasprofesionales) && edad<23) //nivel 4

        aforoDisponible = aforoDisponible * 0,3;
        plazasProfesionales = 0.9*aforoDisponible;
        plazasPrescindibles=0.1*aforoDisponible;

        /* Incremento del 50% */
        precioBillete = precioBillete * 1,5;

else if (contagios>501 && (esPrescindible && hayplazasprescindibles) || noesPrescindible && hayplazasprofesionales) && edad>65)

        aforoDisponible = aforoDisponible * 0,3;
        plazasProfesionales = 0.9*aforoDisponible;
        plazasPrescindibles=0.1*aforoDisponible;

        puedeViajar = false;
endif

if  (puedeViajar)
        return precioBillete;
else
        return -1;
endif
```

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

Parámetros método “calcularPrecioBillete”:

- precio_estandar → int (-infinito a +infinito)
- condición_salud → boolean (true,false) (noEnfermo=true,enfermo=false)
- pasaporte_covid→ boolean (true,false)
- aforo_disponible → int (-infinito a +infinito)
- contagios → int (-infinito a +infinito)
- edad → int (-infinito a +infinito)
- prescindible → boolean (true,false) (prescindible=true,noesprescindible=false)

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.

PARAMETRO	CLASE DE EQUIVALENCIA	VALORES	VALORES LIMITE	CONJETURA DE ERRORES
precio_estandar	(-∞,0),[0, ∞)	-5,5	Variante ligera: 0 Variante Pesada: -1,1	-500000
condición_salud	True,false	True,false	True,false	null
pasaporte_covid	True,false	True,false	True,false	null
Aforo_disponible	(-∞,0),[0, ∞)	-3,3	Variante ligera: 0 Variante Pesada: -1,1	700000
contagios	(-∞,0), [0,100), [100,201), [201,301), [301,501],(501,∞)	-1,20,150,230, 360,600	Variante ligera: 0,100,201,501 Variante Pesada: 1,99,101,200,202,500,502	-700000000
edad	(-∞,0),[0,23),[23,65),[65,∞)	-1,15,37,99	Variante ligera: 0,23,65 Variante Pesada: 1,22,24,64,66	300
prescindible	True,false	True,false	True,false	null

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

Combinaciones totales = 6x3x3x6x18x13x3=5793.

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

Como usamos each use, hay que coger el máximo número de valores del parámetro que más valores tiene, es decir, en nuestro caso haremos 18 pruebas con valores posibles dentro del test suite 1 ya que el parámetro Contagios es el que tiene mayor número de valores.

Test suite 1 = {(-5, true, false, -3,-1,-1, true),
(5, false, null,3,20,15, false),
(0, null, true, 0,150,37, null),
(-1, true, true, -1,230,99, true),
(1, true, false, 1,360,0, true),
(-500000, false, null,700000,600,23, false),
(-5, true, false,3,0,65, false),
(5, false, null, -3,100,1, null),
(0, false, true, -1,201,22, false),
(-1, true, true, 1,501,24, true),
(1, true, true,3,1,64, null),
(-500000, false, false,3,99,66, true),
(-5, true, null, -3,101,300, null),
(5, false, true, -3,200,24, false),
(0, false, true, -1,202,22, false),
(1, true, null, -1,500,15, null),
(-1, true, false, 1,502,37, true),
(0, true, false,1,-700000000,99, false)}

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

CONJUNTO 1:

CONTAGIOS	PASAPORTE COVID
-1	True
-1	False
-1	Null
20	True
20	False
20	Null
150	True
150	False
150	Null
230	True
230	False
230	Null
360	True
360	False
360	Null
600	True
600	False
600	Null
0	True
0	False
0	Null
100	True
100	False
100	Null
201	True
201	False
201	Null
501	True
501	False
501	Null
1	True
1	False
1	Null
99	True
99	False
99	Null
101	True
101	False
101	Null

200	True
200	False
200	Null
202	True
202	False
202	Null
500	True
500	False
500	Null
502	True
502	False
502	Null
-700000000	True
-700000000	False
-700000000	Null

Serían 18 x 3 = 54 combinaciones.

CONJUNTO 2:

EDAD	PRESCINDIBLE
-1	True
-1	False
-1	Null
15	True
15	False
15	Null
37	True
37	False
37	Null
99	True
99	False
99	Null
0	True
0	False
0	Null
23	True
23	False
23	Null
65	True
65	False
65	Null
1	True

1	False
1	Null
22	True
22	False
22	Null
24	True
24	False
24	Null
64	True
64	False
64	Null
66	True
66	False
66	Null
300	True
300	False
300	Null

Serían 13 x 3 = 39 combinaciones.

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

Precio_estandar	Condicion_salud	Pasaporte_covid	Aforo_disponible	Contagios	Edad	prescindible	Decision Cumplida
-	F	V	10	50	20	-	Primera decisión (TRUE) Segunda decisión (TRUE) Undécima decisión (TRUE)
-	F	V	0	110	20	-	Primera decisión (FALSE) Segunda decisión (FALSE) Tercera decisión (FALSE) Cuarta decisión (TRUE)

-	-	-	-	50	70	-	Tercera decisión (TRUE) Cuarta decisión (FALSE) Quinta decisión (FALSE)
-	-	-	-	110	70	-	Quinta decisión (TRUE) Sexta decisión (FALSE) Séptima decisión (FALSE) Octava decisión (FALSE) Novena decisión (FALSE) Decima decisión (FALSE)
-	-	-	100	220	70	V	Sexta decisión (TRUE)
-	-	-	100	320	20	V	Séptima decisión (TRUE)
-	-	-	100	320	70	V	Octava decisión (TRUE)
-	-	-	100	520	20	V	Novena decisión (TRUE)
-	-	-	100	520	70	V	Decima decisión (TRUE) Undécima decisión (FALSE)

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

PRIMERA DECISIÓN:

A: condición_salud
B: pasaporte_covid
C: aforo_disponible

A	B	C	A & B & C	Condición dominante
V	V	V	V	A,B,C
V	V	F	F	C
V	F	V	F	B
V	F	F	F	B,C
F	V	V	F	A
F	V	F	F	A,C
F	F	V	F	A,B
F	F	F	F	A,B,C

SEGUNDA Y TERCERA DECISIÓN:

A: contagios
B: edad

A	B	A & B	Condición dominante
V	V	V	A,B
V	F	F	B
F	V	F	A
F	F	F	A,B

CUARTA Y QUINTA DECISIÓN:

A: contagios

B: contagios

C: edad

A	B	C	A & B & C	Condición dominante
V	V	V	V	A,B,C
V	V	F	F	C
V	F	V	F	B
V	F	F	F	B,C
F	V	V	F	A
F	V	F	F	A,C
F	F	V	F	A,B
F	F	F	F	A,B,C

SEXTA, SEPTIMA Y OCTAVA DECISIÓN:

A: contagios

B: contagios

C: prescindible

D: aforo

E: prescindible

F: aforo

G: edad

NOTA: Al tener 7 variables la tabla de verdad nos daría 49 combinaciones, para abreviar hemos puesto las filas con las condiciones dominantes para cada variable.

A	B	C	D	E	F	G	A & B & ((C & D) | (E & F)) & G	Condición dominante
V	V	V	V	V	V	V	V	A,B,C,D,E,F,G
F	F	F	F	F	F	F	F	A,B,C,D,E,F,G

NOVENA Y DECIMA DECISIÓN:

A: contagios

B: prescindible

C: aforo

D: prescindible

E: aforo

F: edad

NOTA: Al tener 6 variables la tabla de verdad nos daría 36 combinaciones, para abreviar hemos puesto las filas con las condiciones dominantes para cada variable.

A	B	C	D	E	F	A & ((B & C) | (D & E)) & F	Condición dominante
V	V	V	V	V	V	V	A,B,C,D,E,F,G
F	F	F	F	F	F	F	A,B,C,D,E,F,G

UNDECIMA DECISIÓN:

A: viaje_posible

A	A	Condición dominante
V	V	A
F	F	A

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?

Según nuestro criterio la 4 sería la menos optima, ya que se tomaría un tiempo excesivo en realizar todas las pruebas de la combinatoria posible. La que mejor cobertura alcanzaría y que además tenga un tiempo optimo seria *pairwise,* ya que la de *each-use* tiene un tiempo bastante bueno, pero no llega a alcanzar la misma cobertura porque tiene menos pruebas totales.

PROBLEMA 3

(Ver enunciado en el PDF)

Consideración inicial: Pertinencia funcional en el rango [10,35) = 1

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

```
    public static int calcularAdecuacion(int[] array){
      final int[] completitudFuncional = {0,1,2,2,3,4};
      final int[] correcionFuncional = {0,1,1,2,3,5};
      final int[] persitenciaFuncional = {0,1,2,3,4,5};
          //HE CAMBIADO EL 2 POR EL 1 EN EL RANGO 10,35

      int indice = 0;

      int[] arrayAux = new int [3];

      for(int i = 0; i < 3; i++){
          if(array[i] >= 0 && array[i] < 10){
              indice = 0;
          }else if(array[i] >= 10 && array[i] < 35){
              indice = 1;
          }else if(array[i] >= 35 && array[i] < 50){
              indice = 2;
          }else if(array[i] >= 50 && array[i] < 70){
              indice = 3;
          }else if(array[i] >= 70 && array[i] < 90){
              indice = 4;
          }else if(array[i] >= 90 && array[i] <= 100){
              indice = 5;
          }

          if(i == 0){
             arrayAux[i] = completitudFuncional[indice];
          }else if(i == 1){
              arrayAux[i] = correcionFuncional[indice];
          }else{
              arrayAux[i] = persitenciaFuncional[indice];
          }

      }

      return min(arrayAux);
  }

  public static int calcularMantenibilidad(int[] array){
      final int[] modularidad = {0,1,2,2,3,4};
      final int[] reusabilidad = {0,1,2,2,3,5};
      final int[] analizabilidad = {0,0,1,2,3,5};
          //HE CAMBIADO EL 2 POR EL 1 EN EL RANGO 10,35
      final int[] capacidadModif = {0,1,2,3,4,5};
      final int[] capacidadProbado = {0,1,1,2,4,4};
```

```
        int indice = 0;

        int[] arrayAux = new int [5];

        for(int i = 0; i < 5; i++){
            if(array[i] >= 0 && array[i] < 10){
                indice = 0;
            }else if(array[i] >= 10 && array[i] < 35){
                indice = 1;
            }else if(array[i] >= 35 && array[i] < 50){
                indice = 2;
            }else if(array[i] >= 50 && array[i] < 70){
                indice = 3;
            }else if(array[i] >= 70 && array[i] < 90){
                indice = 4;
            }else if(array[i] >= 90 && array[i] <= 100){
                indice = 5;
            }

            if(i == 0){
               arrayAux[i] = modularidad[indice];
            }else if(i == 1){
                arrayAux[i] = reusabilidad[indice];
            }else if (i == 2){
                arrayAux[i] = analizabilidad[indice];
            }else if (i == 3){
                arrayAux[i] = capacidadModif[indice];
            }else if (i == 4){
                arrayAux[i] = capacidadProbado[indice];
            }

        }

        return min(arrayAux);
    }

    public static int calcularCalidadGlobal (int adecuacion, int
mantenibilidad) throws Exception{
        final int [][] tabla =
{{1,1,1,1,1},{1,2,2,2,2},{2,2,3,3,3},{3,3,3,3,4},{3,3,4,4,5}};
        if (adecuacion < 1 || mantenibilidad < 1) throw new
Exception("Alguna de las mediciones base no es valida");
        return tabla[adecuacion - 1][mantenibilidad - 1];
    }
```

***El método min() queda fuera del alcance de estas pruebas, consiste en un método que calcula el valor minimo de un array de enteros, se parte de que ya esta implementado correctamente.**
**** Además suponemos que el método calcularCalidadGlobal solo va a ser llamado con valores retornados por las funciones calcularMantenibilidad y calcularAdecuación**

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

Completitud funcional: entero
Corrección funcional: entero
Pertinencia funcional: entero
Modularidad: entero
Reusabilidad: entero
Analizabilidad: entero
Capacidad de ser modificado: entero
Capacidad de ser probado: entero

Todas estas variables pueden tomar valores enteros entre -∞ y +∞ (infinitos)

Se identifican estas porque son las entradas de los métodos calcularAdecuacion y calcularMantenibilidad (introducidas en orden en arrays en cada método) y el resultado de estos métodos se pasa como parámetro al método calcularCalidadGlobal (siempre se debe invocar con los valores de retorno de las funciones anteriores), cuyo valor de retorno nos dirá si se puede certificar o no o si hay un error.

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.

Parametro	Clase de equivalencia	Valores	Valores Límite	Conjetura de Errores
completitud funcional	(-∞,0),[0,10),[10,35),[35,50),[50,70),[70,90),[90,100],(100,+∞)	-2,8,15,40,58,80,95,110	Variante ligera: 0,10,35,50,70,90,100 Variante pesada: -1,1,9,11,34,36,49,51,69,71,89,91,99,101	-2147483647
correccion funcional	(-∞,0),[0,10),[10,35),[35,50),[50,70),[70,90),[90,100],(100,+∞)	-2,8,15,40,58,80,95,110	Variante ligera: 0,10,35,50,70,90,100 Variante pesada: -1,1,9,11,34,36,49,51,69,71,89,91,99,101	-2147483647
pertinencia funcional	(-∞,0),[0,10),[10,35),[35,50),[50,70),[70,90),[90,100],(100,+∞)	-2,8,15,40,58,80,95,110	Variante ligera: 0,10,35,50,70,90,100 Variante pesada: -1,1,9,11,34,36,49,51,69,71,89,91,99,101	-2147483647
modularidad	(-∞,0),[0,10),[10,35),[35,50),[50,70),[70,90),[90,100],(100,+∞)	-2,8,15,40,58,80,95,110	Variante ligera: 0,10,35,50,70,90,100 Variante pesada: -1,1,9,11,34,36,49,51,69,71,89,91,99,101	-2147483647

reusabilidad	(-∞,0),[0,10),[10,35),[35,50),[50,70),[70,90),[90,100],(100,+∞)	-2,8,15,40,58,80,95,110	Variante ligera: 0,10,35,50,70,90,100 Variante pesada: -1,1,9,11,34,36,49,51,69,71,89,91,99,101	-2147483647
analizabilidad	(-∞,0),[0,10),[10,35),[35,50),[50,70),[70,90),[90,100],(100,+∞)	-2,8,15,40,58,80,95,110	Variante ligera: 0,10,35,50,70,90,100 Variante pesada: -1,1,9,11,34,36,49,51,69,71,89,91,99,101	-2147483647
capacidad de ser modificado	(-∞,0),[0,10),[10,35),[35,50),[50,70),[70,90),[90,100],(100,+∞)	-2,8,15,40,58,80,95,110	Variante ligera: 0,10,35,50,70,90,100 Variante pesada: -1,1,9,11,34,36,49,51,69,71,89,91,99,101	-2147483647
capacidad de ser probado	(-∞,0),[0,10),[10,35),[35,50),[50,70),[70,90),[90,100],(100,+∞)	-2,8,15,40,58,80,95,110	Variante ligera: 0,10,35,50,70,90,100 Variante pesada: -1,1,9,11,34,36,49,51,69,71,89,91,99,101	-2147483647

Cuidado, hay valores negativos pero al copiar la tabla aparecen con salto de linea

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

Como la cantidad de valores para todas las variables es 30 y tenemos 8 variables el resultado es 30^8=656.100.000.000

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

En este caso hay 30 valores, que además se repiten en las 8 variables dando lugar a una test suite de 30 casos:

(-2,x,x,x,x,x,x,x)

(8,x,x,x,x,x,x,x)

(15,x,x,x,x,x,x,x)

(40,x,x,x,x,x,x,x)

(58,x,x,x,x,x,x,x)

(80,x,x,x,x,x,x,x)

(95,x,x,x,x,x,x,x)

(110,x,x,x,x,x,x,x)

(0,x,x,x,x,x,x,x)

(10,x,x,x,x,x,x,x)

(35,x,x,x,x,x,x,x)

(50,x,x,x,x,x,x,x)

(70,x,x,x,x,x,x,x)
(90,x,x,x,x,x,x,x)
(100,x,x,x,x,x,x,x)
(-1,x,x,x,x,x,x,x)
(1,x,x,x,x,x,x,x)
(9,x,x,x,x,x,x,x)
(11,x,x,x,x,x,x,x)
(34,x,x,x,x,x,x,x)
(36,x,x,x,x,x,x,x)
(49,x,x,x,x,x,x,x)
(51,x,x,x,x,x,x,x)
(69,x,x,x,x,x,x,x)
(71,x,x,x,x,x,x,x)
(89,x,x,x,x,x,x,x)
(91,x,x,x,x,x,x,x)
(99,x,x,x,x,x,x,x)
(10,x,x,x,x,x,x,x)

Las x representan valores cualesquiera del espacio de valores de prueba.

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

Como las variables tienen los mismos valores, para alcanzar cobertura pairwise, la tabla queda igual para todos los pares de variables

completitud funcional	corrección funcional
90	-2147483647
110	15
95	40
95	91
10	50
70	10
49	99
91	58
58	34
49	1
-2147483647	51
69	50

90	101
51	50
9	-2
35	9
10	91
51	9
15	10
99	70
-1	34
36	10

(la tabla sigue con combinaciones, pueden generarse con la herramienta PICT online Pairwise Pict Online (yuuniworks.com))

Como todas las variables tienen 30 valores de prueba, para cada conjunto habría siempre 30*30=900 casos de prueba.

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

```
if(array[i] >= 0 && array[i] < 10){
    indice = 0;
}else if(array[i] >= 10 && array[i] < 35){
    indice = 1;
}else if(array[i] >= 35 && array[i] < 50){
    indice = 2;
}else if(array[i] >= 50 && array[i] < 70){
    indice = 3;
}else if(array[i] >= 70 && array[i] < 90){
    indice = 4;
}else if(array[i] >= 90 && array[i] <= 100){
    indice = 5;
}
```

El método va comprobando cada parámetro de los 3 o 5 en el array para ver en qué rango encaja por lo que un posible conjunto de casos de prueba para que afecte a todas las decisiones lo encontramos llamando al método calcularMantenibilidad con el array { 5, 40, 56, 77, 99 }. Es imposible que más de una condición se cumpla a la vez en una iteración del bucle.

```
if (adecuacion < 1 || mantenibilidad < 1) throw new Exception("Alguna de
las mediciones base no es valida");
else return tabla[adecuacion - 1][mantenibilidad - 1];
```

adecuación	mantenibilidad	decisión cumplida
0	2	primera decisión = false, segunda decisión=true

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

Para el primer fragmento de decisiones no tiene sentido puesto que las decisiones son exclusivas, nunca se pueden dar a la vez.

A=adecuación < 1

B=mantenibilidad <1

A	B	A or B	Cond. Dominante
T	F	T	A
F	T	T	B
T	T	T	A,B
F	F	F	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 cobertura se puede alcanzar únicamente con los 30 valores que hemos obtenido en el apartado 3 ya que aunque vayan en parámetros distintos pasan por las mismas secciones de código por lo que podríamos completar fácilmente las pruebas si un gran tiempo ni esfuerzo.