**CASOS DE EQUIVALENCIA**

**PRIMER CASO DE EQUIVALENCIA: MUERTO**

!Person.isAlive();

Definición : atributo de la clase Person definada como true

**Casos de prueba:**

Person.setAlive(false);

**SEGUNDO CASO DE EQUIVALENCIA: BAJO EDAD**

Person.getAge() <18 & Person.getAge() > 0 & Person.getAlive()

Definición: Atributo Age de la clase Person definida con un valor entero menor a 18

**Casos de prueba:**

**Caso de prueba 1:** Person.setAge(12);

**Caso de prueba 2:** Person.setAge(16)

**TERCER CASO DE EQUIVALENCIA: EDAD INVÁLIDA**

Person.getAge() < 0 & Person.getAlive()

Definición : Atributo Age de la clase Person definido con un valor negativo

**Caso de prueba:**

**Caso de prueba 1:** Person.setAge(-5);

**Caso de prueba 2:** Person.setAge(-100);

**CUARTO CASO DE EQUIVALENCIA: DUPLICADO**

Person1.getID() == Person2.getID() & Person.getAlive()

Definición: Atributo ID de la clase Person ya se encuentra registrado en los votos

**Caso prueba 1:** Person.setID(1234);

**Caso prueba 2:** Person.setID(1234);

**QUINTO CASO DE EQUIVALENCIA: EDAD EXCESIVA**

Person.getID() > 120 & Person.getAlive()

Definición: Atributo Age de la clase Person definido con un valor mayor a 120

**Caso de prueba 1:** Person.setAge(150);

**Caso de prueba 2:** Person.setAge(1000);

**SEXTO CASO DE EQUIVALENCIA: PERSONA VÁLIDA**

Person.getAlive() & Person.getGender() != Undefined & Person.getAge() > = 18 & Person.getAge() <= 120 & !Duplicated

Definición: La persona es válida si está viva, el genero esta definidio y tiene una edad entre los 18 y 120 años y ademas no tiene un registro de voto

**Caso de prueba 1:** new Person(“Steven”,2124523,23,MALE,true);

**AERODESCUENTOS**

**DESCUENTOS:**

1. 15% descuento sacando el billete con antelación superior a 20 días.
2. **5%** a los pasajeros menores a 18 años y **8%**  mayores de 65.

/\*\* calcular la tarifa de cada billete según el trayecto, la antelación en la que se obtiene el billete y la edad del pasajero, de acuerdo con la normativa 005.

@param tarifaBase valor base del vuelo

@param diasAntelacion dias de antelación del vuelo

@param edad - edad del pasajero

@throws ExcepcionParametrosInvalidos [tarifaBase>0, diasAntelación >0 , edad >0] \*\*/

public long calculoTarifa(long tarifaBase, int diasAntelacion, int edad)

|  |  |  |
| --- | --- | --- |
| **Número** | **Clase de equivalencia** | **Resultado correcto/incorrecto** |
| **1** | tarifaBase>0 | **US$1000/US$-12** |
| **2** | diasAntelación>0 | **30/-10** |
| **3** | Edad>0 & Edad < 120 | **40/0** |
| **4** | diasAntelación:Integer edad,tarifaBase:long | **Edad:12,tarifa:24.3,días:28** |
| **5** |  |  |

**Clase de equivalencia 1, caso de prueba:**

@Test

public long validarTarifa(){

calculoTarifa(-12,20,18);

}

**Datos de entrada:**

tarifaBase: -12

diasAntelación: 20

edad: 18

**Resultado esperado:**

Tarifa base del tiquete inválido

**Clase de equivalencia 2, caso de prueba:**

@Test

public long validarTarifa(){

calculoTarifa(453.2,-2,23);

}

**Datos de entrada:**

tarifaBase: 453.2

diasAntelación: -2

edad: 23

**Resultado esperado:**

Días de antelación de la reserva inválido

**Clase de equivalencia 3, caso de prueba:**

@Test

public long validarTarifa(){

calculoTarifa(1000,40,0);

}

**Datos de entrada:**

tarifaBase: 1000

diasAntelación: 40

edad: 0

**Resultado esperado:**

Edad inválida

**Clase de equivalencia 4, caso de prueba:**

@Test

public long validarTarifa(){

calculoTarifa(234,30,”veinte”);

}

**Datos de entrada:**

tarifaBase: 234

diasAntelación: 30

edad: “veinte”

**Resultado esperado:**

Tipo de dato Inválido