**AeroDescuentos**

Se está desarrollando para una aerolínea su módulo de liquidación de tiquetes aéreos. Para el mismo, se tiene una función que aplica descuentos a la tarifa base del vuelo dependiendo del tiempo de antelación de la reserva y la edad del pasajero. Los descuentos SON ACUMULABLES.

*Normativa 005, sobre los descuentos:*

* ***15%*** *de descuento sacando el billete con antelación superior a 20 días.*
* ***5%*** *a los pasajeros con edad inferior a 18 años y* ***8%*** *a los pasajeros con edad superior a 65 años.*

La siguiente es la especificación de la función que se usará en el módulo del cálculo de los descuentos:

/\*\* 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 [XXXXXXXXXXXXXXXXXXXXXXXXXXX]

\*\*/ public long calculoTarifa(long tarifaBase, int diasAntelacion, int edad)

1. **De acuerdo con lo indicado, y teniendo en cuenta que NO hay precondiciones, ¿en qué casos se debería arrojar una excepción de tipo ExcepcionParametrosInvalidos?. Agregue esto a la especificación.**

* No puede ser ingresada una edad menor 0 años.
* No puede ser ingresada una edad mayor a 110 años pues es poco probable que pase.
* No puede haber días de antelación menores a 0.
* No puede haber una tarifa base menor a 0.

1. **En la siguiente tabla enumere un conjunto de clases de equivalencia que -según usted- creen una buena división del conjunto de datos de entrada de la función anterior:**

|  |  |  |
| --- | --- | --- |
| **Número** | **Clase de equivalencia (en lenguaje natural o matemático).** | **Resultado correcto / incorrecto.** |
| 1. | Validar si el cliente tiene menos de 18 años | Correcto |
| 2. | Validar si el cliente tiene más de 65 años | Correcto |
| 3. | Validar si el cliente reservo el tiquete con una antelación mayor a 20 días | Correcto |

1. **Para cada clase de equivalencia, defina un caso de prueba específico, definiendo: parámetros de entrada y resultados esperados.**

/\*\*

Calcular la tarifa de cada billete dependiendo si la edad del pasajero es menor o mayor a 18 años de edad.

@param tarifaBase valor base del vuelo

@param edad - edad del pasajero

@throws ExcepcionParametrosInvalidos edad como un entero negativo o mayor a 110

\*\*/

@return Devuelve un numero con el precio del boleto dependiendo si aplicó o no los descuentos

public long descuentoeEdad18(long tarifaBase, int edad)

/\*\*

Calcular la tarifa de cada billete dependiendo si la edad del pasajero es menor o mayor a 65 años de edad.

@param tarifaBase valor base del vuelo

@param edad - edad del pasajero

@return De vuelve un numero con el precio del boleto dependiendo si aplicó o no los descuentos

@throws ExcepcionParametrosInvalidos edad como un entero negativo o mayor a 110

\*\*/

public long descuentoeEdad65(long tarifaBase,int edad)

/\*\*

Calcular la tarifa de cada billete dependiendo con los días de antelación con las que el pasajero halla reservado el vuelo.

@param tarifaBase valor base del vuelo

@param diasAntelacion dias de antelación del vuelo

@return De vuelve un numero con el precio del boleto dependiendo si aplicó o no los descuentos

@throws ExcepcionParametrosInvalidos edad como un entero negativo o mayor a 110

\*\*/

public long descuentoeEdad65(long tarifaBase, int diasAntelacion)

1. **A partir de las clases de equivalencia identificadas en el punto 2, identifique las condiciones límite o de frontera de estas.**

Cuando la edad es de 0 años y la edad es de 18 años

Cuando la edad es de 65 años y de 110 años

Cuando los días de antelación son 20

1. **Para cada una de las condiciones de frontera anteriores, defina casos de prueba específicos.**

**/\*\***

Verificar que este entre 0 y 18 años

@test

Public void validarEdad()

**\*\*/**

**/\*\***

Verificar que este entre 65 y 110 años

@test

Public void validarEdad2()

**\*\*/**

**/\*\***

Verigicar que los dias esten entre 0 a 20.

@test

Public void validarDiasAntelacion()

**\*\*/**