**ENTREGABLE**

**INVARIANTE DE REPRESENTACIÓN**

Ejercicio 1:

**public class BolaPool {**

**// Variables de instancia**

**int numero;**

**String tipo;**

**public boolean validar() {**

**if (this.numero < 1 || this.numero > 15 ) {**

**return false;**

**}**

**else if (this.numero == 8 && this.tipo == "negro") {**

**return true;**

**}**

**else if (this.numero <= 7 && this.tipo == "lisa") {**

**return true;**

**}**

**else if (this.numero >= 9 && this.tipo == "rayada") {**

**return true;**

**}**

**else {**

**return false;**

**}**

**}**

**}**

/\*

\* Una bola con número 0, ó negativo, ó 16

\* y superior no es una bola de pool válida.

\*

\* Si una bola posee el número 8 sólo es válida

\* si su tipo es "negro"

\*

\* Si una bola posee un número entre 1 y 6 inclusive

\* sólo es válida si es de tipo "lisa".

\*

\* Si una bola posee un número entre 8 y 15 inclusive

\* sólo es válida si es de tipo "rayada".

\*

\*/

Ejercicio 2:

**public class Caloventor {**

**boolean encendido;**

**char modo;**

**int velocidad;**

**public boolean validar() {**

**if (this.modo != 'c' && this.modo != 'f') {**

**return false;**

**}**

**if (this.velocidad < 0 || this.velocidad > 3) {**

**return false;**

**}**

**if (this.encendido == false || this.velocidad != 0) {**

**return false;**

**}**

**return true;**

**}**

**}**

/\*

\* Un caloventar sólo es válido si su modo es 'c' o 'f'.

\* De "otro modo" no sería válido.

\*

\* Un caloventor válido sólo posee velocidades entre 0 y 3.

\* Velocidad negativas ó superiores ó iguales 4 son inválidas.

\*

\* Si el estado encendido es falso (apagado), la velocidad no

\* puede ser otra velocidad que cero. Otra velocidad provocaría

\* un caloventor fuera del invariante de representación.

\*

\*

\*/

Ejercicio 3

**public class Tateti {**

**char[][] tablero;**

**char proximoTurno;**

**boolean terminado;**

**char ganador;**

**public boolean estaCorrecto() {**

/\*

\* Si el tamaño del arreglo tablero es diferente

\* de 3 el tablero está mal construido y no es un tateti válido

\*/

**if (this.tablero.length != 3) {**

**return false;**

**}**

/\*

\* Si el tamaño de cada arreglo del arreglo

\* tablero es diferente de 3 el tablero

\* está mal construido y no es un tateti válido

\*/

**for (int i = 0; i < this.tablero.length; i++) {**

**if (this.tablero[i].length != 3) {**

**return false;**

**}**

**}**

// Si el juego está terminado

**if (this.terminado == true) {**

**// El ganador sólo puede ser 'x' ó 'o' ó a lo sumo empate ' '.**

**// Otro resultado no es válido**

**if (this.ganador != 'x' && this.ganador != 'o' && this.ganador != ' ') {**

**return false;**

**}**

// si hay un lugar vacío en el tablero

// terminado, alguien hace trampa

**for (int a = 0; a < 3; a++) {**

**for (int b = 0; b < 3; b++) {**

**if (tablero[a][b] == ' ') {**

**return false;**

**}**

**}**

**}**

// Si el ganador (quien acaba de mover la ficha es 'x'

// El próximo jugador no debe ser el mismo (más alla

// de que el juego ha terminado, el próximo, debería

// ser el jugador opuesto.

**if (this.ganador == 'x' || this.proximoTurno == 'x') {**

**return false;**

**}**

// Si el ganador (quien acaba de mover la ficha es 'o'

// El próximo jugador no debe ser el mismo (más alla

// de que el juego ha terminado, el próximo, debería

// ser el jugador opuesto.

**if (this.ganador == 'o' || this.proximoTurno == 'o') {**

**return false;**

**}**

// Juego no terminado

**} else {**

// Si el juego aún continúa, el próximo jugador no puede ser

// alguien distinto de 'x' o 'o'

**if (this.proximoTurno != 'x' && this.proximoTurno != 'o') {**

**return false;**

**}**

// Un juego que aún continúa no debería

// tener una cantidad de casillas ocupadas

// igual al total del tablero.

**int contar = 0;**

**for (int a = 0; a < 3; a++) {**

**for (int b = 0; b < 3; b++) {**

**if (tablero[a][b] != ' ') {**

**contar++; // cuenta espacios vacios**

**}**

**}**

**}**

**if (contar == 9) {**

**return false;**

// si las posiciones distintas a vacio es igual al

// total entonces este juego deberia estar

// terminado.

**}**

**}**

// Tableros que no hayan sido determinado inválidos

// por las definiciones anteriores, entonces son

// válidos.

**return true;**

**}**

**}**

Ejercicio 4

**public Vuelo**

**{**

**int nroVuelo;**

**Pasajero[] pasajeros;**

**String origen;**

**String destino;**

**String[] escalas;**

**String ultimaEscala;**

**Hora hSalidaProgramada;**

**Hora hPartida;**

**int capacidad;**

**public boolean validar() {**

// Si el avión despegó que la

// hora de partida sea mayor que la programada

**if (this.hPartida != null) {**

**if(this.hSalidaProgramada.convertirASegundos() > this.hPartida.convertirASegundos() ) {**

**return false;**

**}**

**}**

// Si hay 2 pasajeros con el mismo pasaporte

// el vuelo es inválido.

**for (int a = 0; a < this.pasajeros.length; a++ ) {**

**for (int b = a + 1; b < this.pasajeros.length; b++ ) {**

**if (this.pasajeros[a].pasaporte == this.pasajeros[b].pasaporte ) {**

**return false;**

**}**

**}**

**}**

// detectando el índice de la escala actual

// en el arreglo de escalas.

**int actual;**

**for (int e = 0; e < this.escalas.length; e++) {**

**if (ultimaEscala.equals(this.escalas[e])) {**

**actual = e;**

**}**

**}**

// Si la última escala no está en el arreglo

// entonces algo está mal.

**if (e == null) {**

**return false;**

**}**

// Para cada pasajero se busca el índice

// del arreglo de escalas que coincide

// con el destino del propio pasajero.

**for (int a = 0; a < this.pasajeros.length; a++ ) {**

**int deberiaBajar;**

**for (int e = 0; e < this.escalas.length; e++) {**

**if (this.pasajeros[a].destino.equals(this.escalas[e])) {**

**deberiaBajar = e;**

**}**

**}**

// Si ese indice es menor que la ubicación actual

// entonces el pasajero se pasó de su destino

**if (deberiaBajar < actual) {**

**return false;**

**}**

**}**

// Si hay más pasajeros que su capacidad

// el vuelo fue sobrevendido

**if (this.capacidad < this.pasajeros.length) {**

**return false;**

**}**

// Si el arreglo de escalas no comienza por el origen

// algo está mal.

**if (!(this.origen.equals(this.escalas[0]))) {**

**return false;**

**}**

// Si el arreglo de escalas no termina con el destino

// algo está mal.

**if (!(this.destino.equals(this.escalas[this.escalas.length - 1]))) {**

**return false;**

**}**

**}**

// Si no fué falso por todo lo anterior

// entonces es un vuelo válido.

**return true;**

**}**

**public class Pasajero**

**{**

**int pasaporte;**

**String origen;**

**String destino;**

**}**