4-1-2021

Javier Mengod

Álvaro Perez

David Ros

Álvaro Fraidias

CASINO UNIZAR

Trabajo dirigido Programación de sistemas concurrentes y distribuidos

Tabla de contenido

[1. Planteamiento del problema 2](#_Toc60680505)

[2. Mecanismos empleados para la resolución del problema 3](#_Toc60680506)

[3. Método de resolución de semáforos 3](#_Toc60680507)

[3.1 Clase Cliente 3](#_Toc60680508)

[3.2 Clase Recepción 6](#_Toc60680509)

[3.3 Clase Encargado 7](#_Toc60680510)

[3.4 Clase Croupier 8](#_Toc60680511)

[3.5 Clase bar 9](#_Toc60680512)

[3.6 Clase Tragaperras 9](#_Toc60680513)

[3.7 Clase TrabajoCasino 10](#_Toc60680514)

[4. Método de resolución de citas 11](#_Toc60680515)

[4.1 Encargado 11](#_Toc60680516)

[4.1.1 Variables locales 11](#_Toc60680517)

[4.1.2 Código 11](#_Toc60680518)

[4.2 Croupier 13](#_Toc60680519)

[4.2.1 Variables locales 13](#_Toc60680520)

[4.2.2 Código 14](#_Toc60680521)

[4.3 Cafetería 15](#_Toc60680522)

[4.3.1 Variables locales 15](#_Toc60680523)

[4.3.2 Código 16](#_Toc60680524)

[4.4 Recepcionista 17](#_Toc60680525)

[4.4.1 Variables locales 17](#_Toc60680526)

[4.4.2 Código 17](#_Toc60680527)

[4.5 Cliente 18](#_Toc60680528)

[4.5.1 Variables locales 18](#_Toc60680529)

[4.5.2 Código 19](#_Toc60680530)



# 1. Planteamiento del problema

Un casino donde van llegando clientes. Una vez llega un cliente, va a recepción a cambiar dinero por fichas. En caso de no quedar fichas, el recepcionista irá al almacén a por más fichas. Mientras tanto el cliente esperará en recepción.

Una vez ha obtenido las fichas elige si quiere jugar al póker o a las máquinas tragaperras.

-Si decide jugar al póker, hablará con el encargado que comprobará si hay mesas disponibles para poder jugar. En caso de no haber mesas, podrá elegir entre jugar a las máquinas tragaperras o se irá a la cafetería a esperar. En la sala de espera podrá tomar un refresco siempre y cuando haya refrescos disponibles. En caso de no haber refrescos disponibles, avisará a la empleada para reponerlos. Esta empleada se encarga de reponer los refrescos y de limpiar la sala cada 10 personas que entren por temas de salud.

En caso de haber mesas disponibles, ocupará una mesa y esperará a que se llene para empezar la partida. Antes de que empiece la partida el cliente puede ir a la barra a por un refresco avisando al croupier, cuando la partida va a empezar el croupier avisara a todos los clientes de esa mesa. Por supuesto el ultimo cliente no podrá ir a la barra a por un refresco, pero el resto sí. Una vez se llene la mesa, el crupier empezará la partida.

-Si el cliente decide jugar a las máquinas tragaperras deberá comprobar que haya máquinas libres. En caso de no haber esperará hasta que otro cliente termine. En caso de estar disponibles ocupará una máquina y jugará.

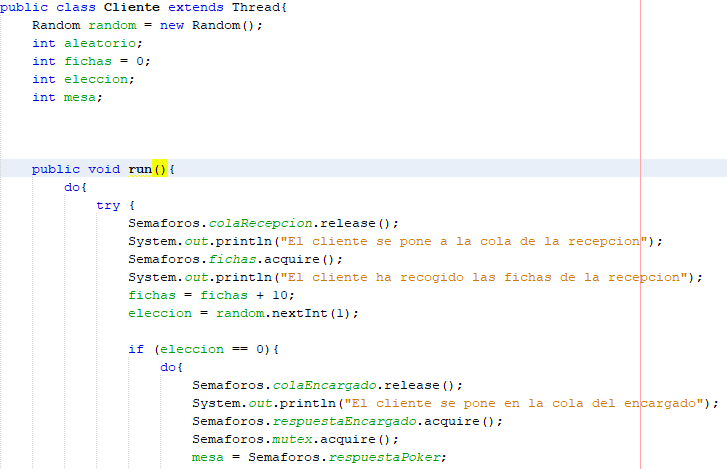
En caso de que el jugador haya querido ir al póker, pero no haya podido ir y este en las tragaperras, tras acabar, revisara si puede ir a jugar al póker, no saldrá del casino hasta que haya jugado al póker;

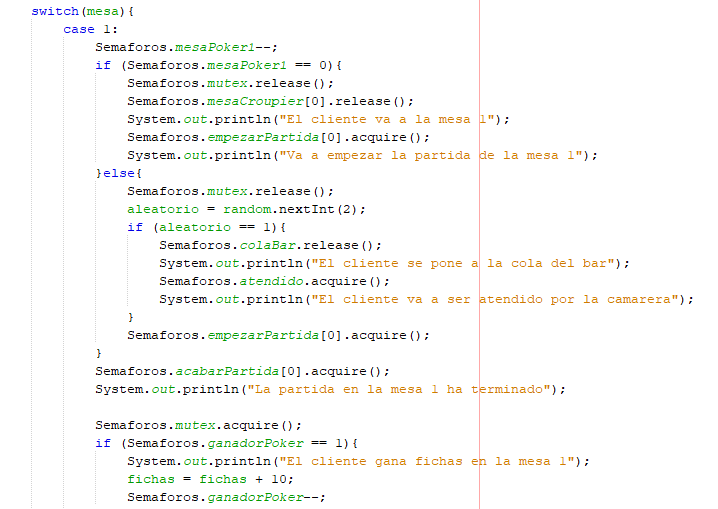
Una vez acabado el juego, tanto en el póker como en las máquinas tragaperras, el cliente podrá elegir entre cobrar las fichas e irse o comprar más fichas y volver a jugar.

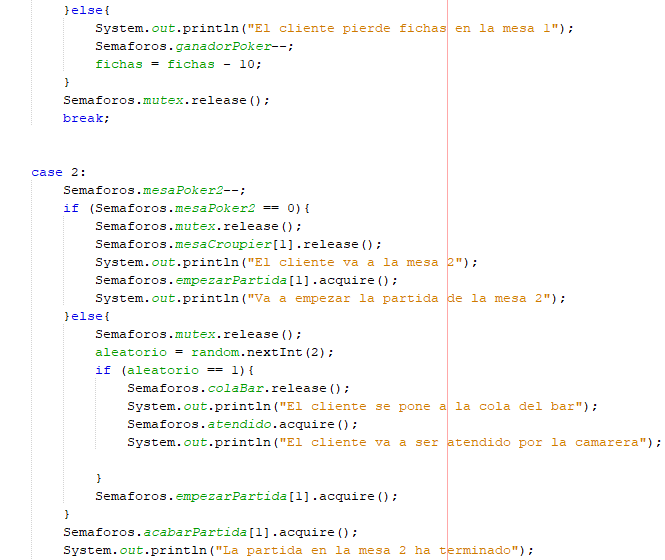
# 2. Mecanismos empleados para la resolución del problema

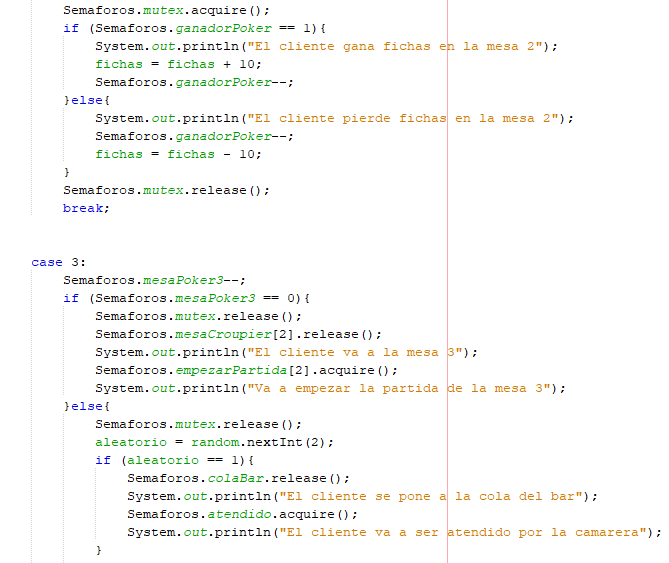
Este problema lo hemos solucionado usando dos de los tres mecanismos de sincronización de procesos vistos en las clases teóricas de la asignatura de Programación de sistemas concurrentes y distribuidos: Semáforos y citas.

# 3. Método de resolución de semáforos

3.1 Clase Cliente 









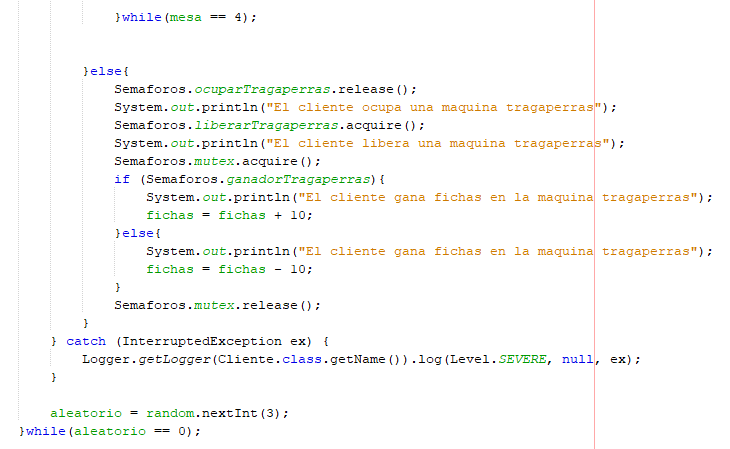


Ilustración : Clase cliente

El cliente lo primero que hace es un signal a la cola de recepción y un wait al semáforo "fichas" hasta que el recepcionista le de las fichas. Entonces aumenta las fichas del cliente en 10 y se comprueba si el cliente quiere ir al póker o a las tragaperras.

Si quiere ir al póker realizará un signal a la cola del encargado y esperará en un wait hasta que el encargado le responda. Su respuesta será ir a una de las tres mesas del póker o si éstas están llenas, ir a las tragaperras. En caso de que vaya a una de las mesas reducirá una variable compartida de la mesa a la que va, si no es el último en llegar a la mesa podrá ir al bar donde hará un signal a la cola del bar y esperará en un wait a ser atendido por el camarero. En caso de que sea el último en llegar, realizará un signal a “mesaCroupier” y esperará a que empiece la partida. Después esperará a que acabe la partida y comprobará si ha ganado o no, si gana aumentará las fichas y sino las perderá.

En caso de que todas las mesas estén ocupadas hará un signal a “ocuparTragaperras” y esperará en “liberarTragaperras”. Tras esto, comprobará si ha ganado o ha perdido y volverá a ir al encargado para jugar al póker.

En caso de que quisiese ir a las tragaperras desde el inicio hará lo citado anteriormente con la condición de en vez de ir al encargado irá o a la recepción o se marchará del casino.

## 3.2 Clase Recepción

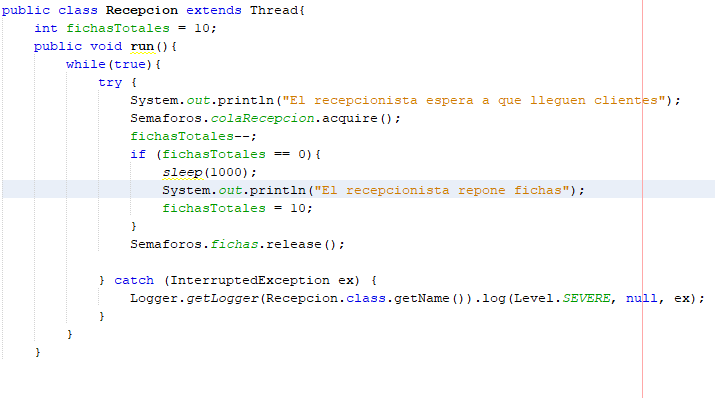
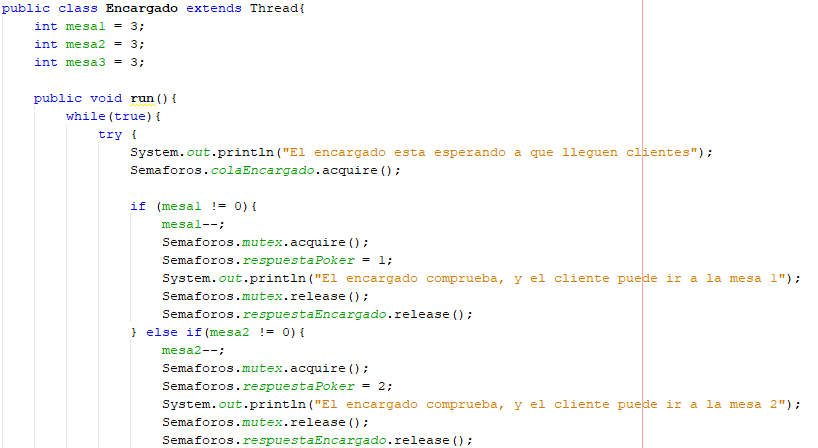


Ilustración : Clase Recepción

En la clase “Recepción” el recepcionista está esperando a que lleguen clientes a la cola de la recepción. Reducirá el número de fichas disponibles en 1 y comprobará si ya no le quedan más fichas. En ese caso deberá ir a reponer fichas. Por último, enviará un signal al semáforo “fichas” para avisar de que ya se han dado las fichas.

## 3.3 Clase Encargado



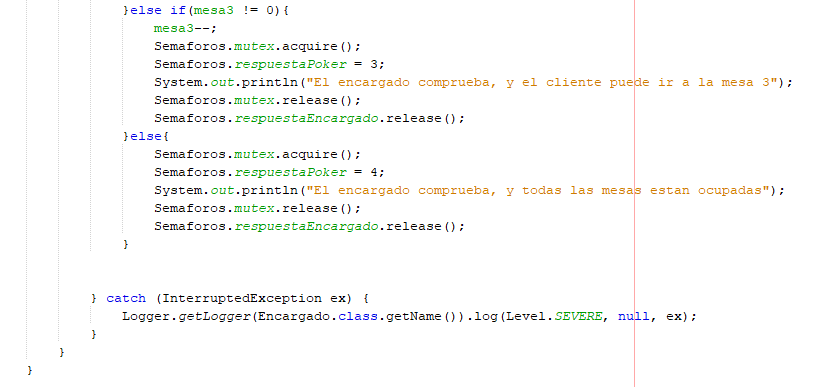
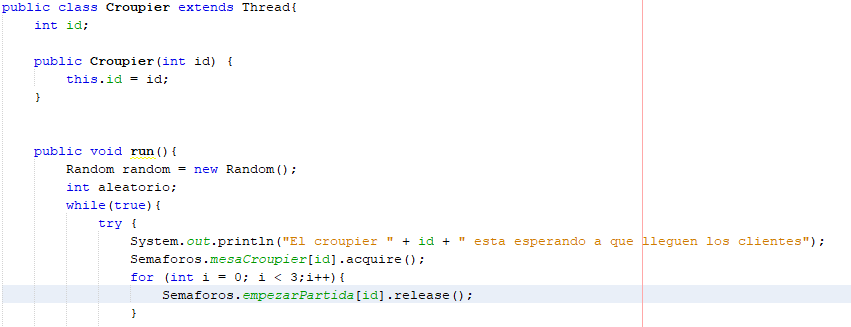


Ilustración : Clase Encargado

El encargado está esperando en “colaEncargado” hasta que lleguen los clientes. Una vez han llegado, comprobará si la mesa 1 está disponible, en ese caso decrementa la variable de esa mesa en uno y pone su respuesta a uno, en caso de no estar disponible hará lo mismo con la 2 y si esta tampoco lo está hará lo propio con la mesa 3. En el caso de que todas las mesas estén ocupadas su respuesta será 4, es decir, ir a jugar a las tragaperras.

## 3.4 Clase Croupier



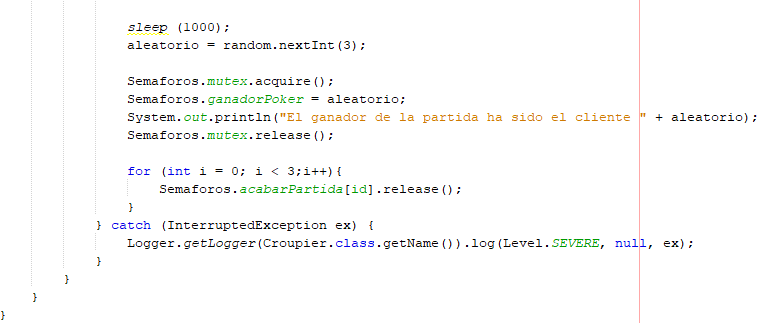


Ilustración : Clase Croupier

El croupier esperará a que lleguen los tres clientes a su mesa, avisará de que va a empezar la partida, se jugará la partida, se elegirá un ganador de la partida y se avisará a los clientes de que ya ha acabado la partida.

## 3.5 Clase bar

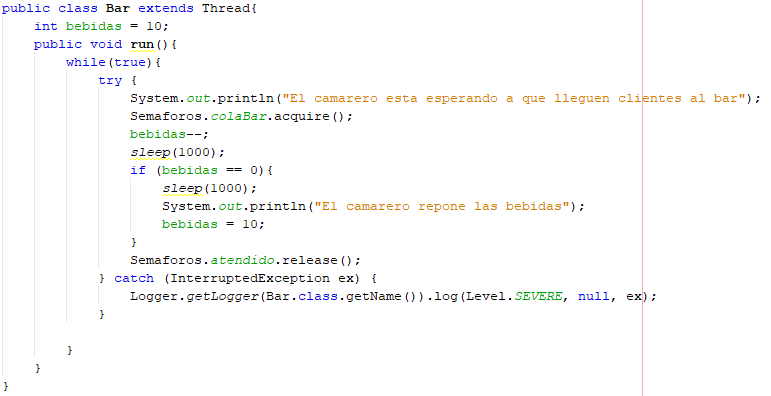


Ilustración : Clase Bar

El camarero espera a que haya clientes en la cola del bar, reducirá en uno la variable “bebidas” y comprobará si ya no quedan bebidas. En este caso deberá de reponer las bebidas. Por último avisará al cliente que ya ha sido atendido.

## 3.6 Clase Tragaperras

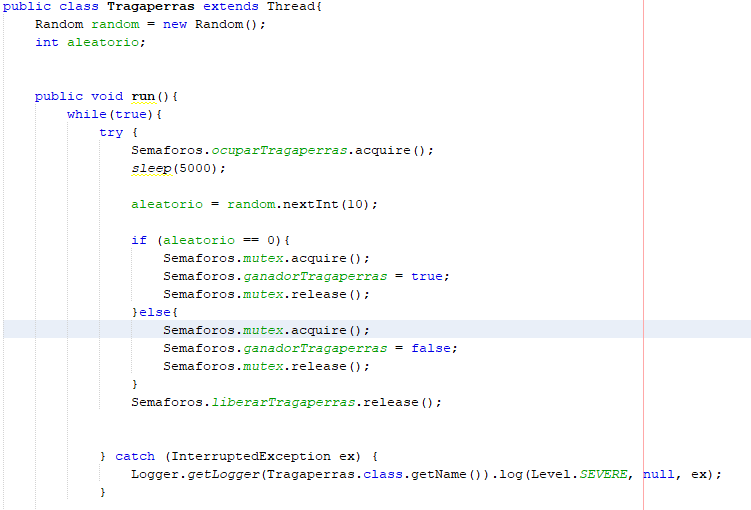


Ilustración : Clase Tragaperras

En esta clase se espera hasta que un cliente llegue a ocupar una tragaperras. Mediante un random se comprobará si el cliente ha ganado o no. Por último, se liberará la tragaperras para que pueda ser utilizada por otro cliente.

## 3.7 Clase TrabajoCasino

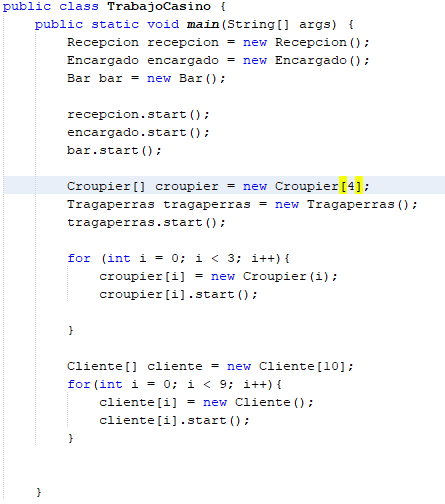


Ilustración : Clase TrabajoCasino

En esta clase inicializamos el resto de las clases una vez excepto “Croupier” que la lanzamos 3 veces y “Cliente” que la lanzamos 10 veces.

# 4. Método de resolución de citas

## 4.1 Encargado

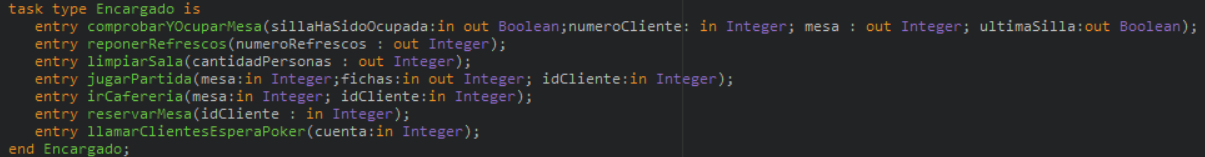


Ilustración : Tarea Encargado

4.1.1 Variables locales

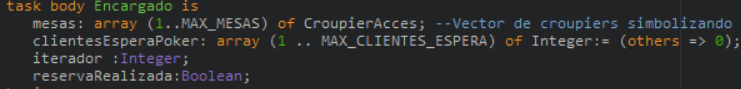


Ilustración : Constantes

* **mesas**: almacena en un array las instancias de la tarea Croupier. La posición del array es la mesa a la que el Croupier tiene que atender.
* **clientesEsperaPoker**: almacena en un array los clientes que están esperando a jugar al poker y no pueden jugar porque están las mesas llenas.
* **iterador**: contiene a la variable que modificaremos para poder recorrer los arrays.
* **reservaRealizada**: booleano que almacena si se ha realizado una reserva.

4.1.2 Código

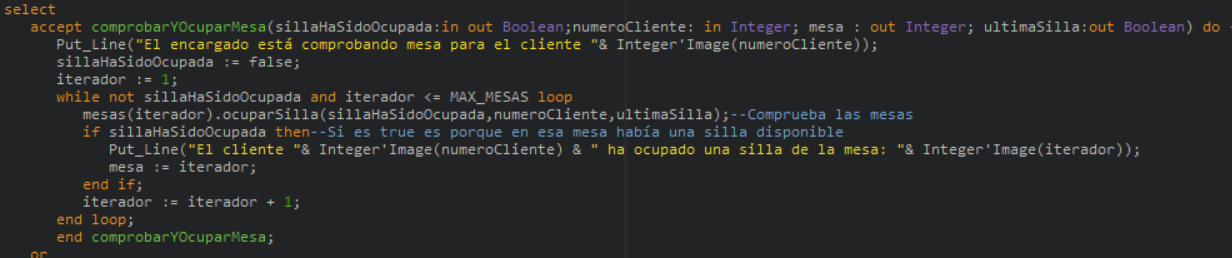


Ilustración : Entrada comprobarYOcuparMesa de la tarea Encargado

Esta entrada, se encarga de ocupar una mesa cuando el cliente lo llama. Si la ha podido ocupar sillaHaSidoOcupada será True, sino será False. Para ello se llama a todos los Crupier hasta que encuentra un sitio libre. Cuando lo encuentra sale de la iteración.

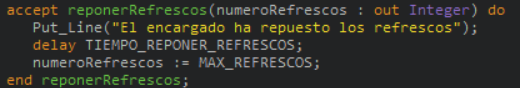


Ilustración : Entrada reponerRefrescos de la tarea Encargado

Esta entrada solamente modificará la variable numeroRefrescos.

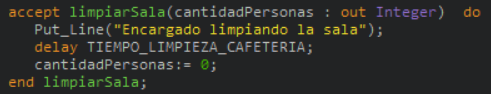


Ilustración : Entrada limpiarSala de la tarea Encargado

Esta entrada limpiará la sala, sacando a las personas en ella. Para simbolizar esto, se resetea la variable “cantidadPersonas” a 0.



Ilustración : Entrada jugarPartida de la tarea Encargado

Esta entrada llamará al Croupier concreto para que el cliente empiece la partida en la mesa del Croupier.



Ilustración : Entrada irCafeteria de la tarea Encargado

Esta entrada llamará a la entrada del Croupier para ir a la cafetería. Al igual que la entrada anterior, se podría hacer sin contactar con el encargado. Pero seria necesario hacer los arrays variables globales y eso no nos interesa ya que en un sistema distribuido estaría el array dividido.

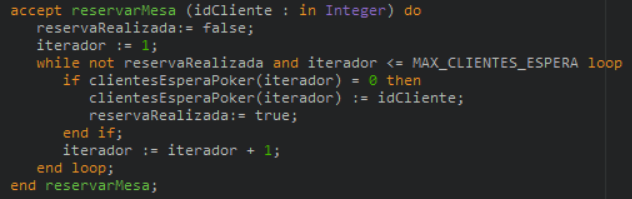


Ilustración : Entrada reservaMesa de la tarea Encargado

Esta entrada posiciona un cliente en el array de clientesEsperaPoker para jugar al poker, para que una vez que se liberen las mesas los que estaban esperando empiecen a jugar.

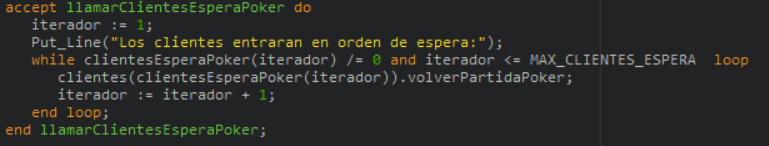


Ilustración : Entrada llamarClientesEsperaPoker

Esta entrada hace que los jugadores que estaban esperando para jugar al poker porque estaban todas las mesas llenas, empiecen a jugar. Para ello llama a todos los clientes esperando.

## 4.2 Croupier

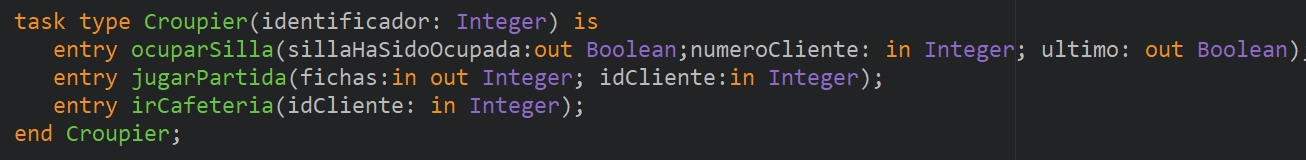


Ilustración : Tarea Croupier

4.2.1 Variables locales

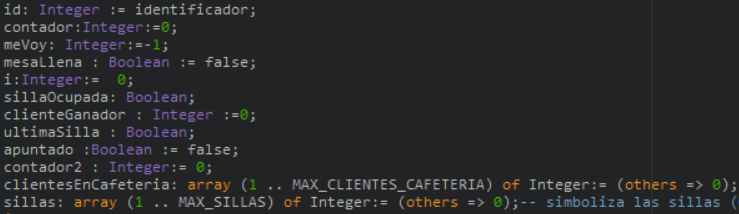


Ilustración : Variables locales

* **id**: es el identificador del Croupier, equivalente al número de mesa que tiene que atender dicho Croupier
* **contador**: variable que contiene el numero de veces que se ejecuta jugarPartida.
* **mesaLLena**: booleano que representa si la mesa del Croupier está llena.
* **i**: variable que utilizaremos para recorrer los bucles.
* **sillaOcupada**: variable booleana que usaremos para almacenar si una silla está ocupada.
* **clienteGanador**: es el identificador del cliente que ganará la apuesta.
* **ultimaSilla**: booleano que sirve para representar si es la ultima silla o no. La finalidad de este booleano es que el último cliente que se siente en la mesa no pueda ir a tomar refrescos.
* **apuntado**: booleano que representa si los clientes han sido registrados como que han ido a la cafetería.
* **clientesEnCafeteria**: array de enteros en el que se almacenan los id de los clientes que se han ido a la cafetería para posteriormente llamarlos.
* **sillas**: array de enteros en el que se almacenan los id de los clientes que se han sentado en la silla cuya posición del array ocupan.

### 4.2.2 Código

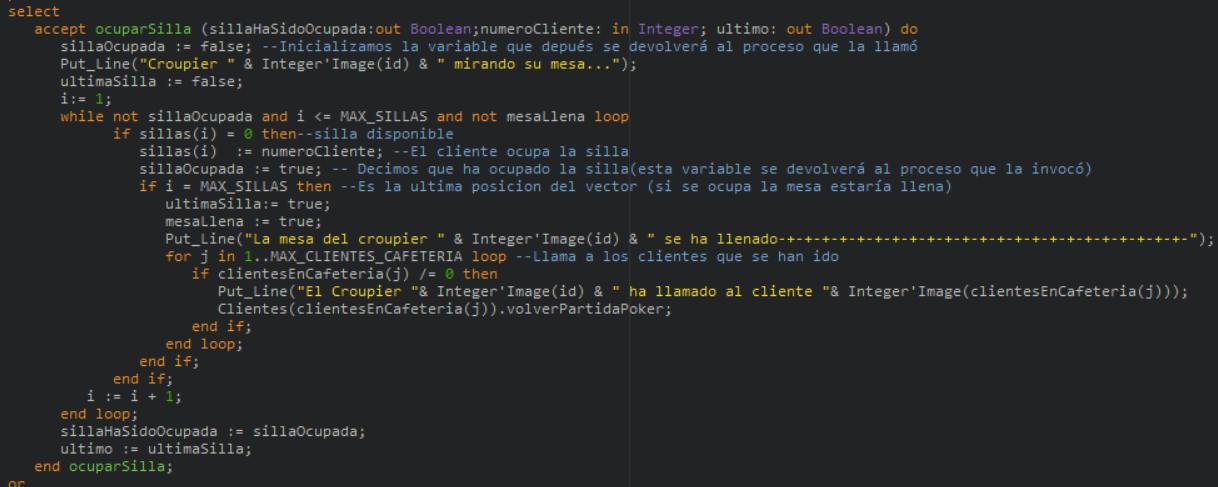


Ilustración : Entrada ocuparSilla de la tarea Croupier

La entrada “ocuparSilla” se encarga de comprobar si hay sillas disponibles en el Croupier. Para ello recorre el bucle en busca de una silla vacía (sillas(i)=0). Si es cualquier posición menos la última, actualiza el valor de la silla correspondiente por el del id del cliente que la ocupa y pone “sillaOcupada”. Si es la última posición tendrá además de hacer lo mismo que antes poner “ultimaSilla” a true y “mesaLLena” a true y tendrá que llamar a los clientes que se han ido a la cafetería. Para ello recorremos el array de “clientesEnCafeteria” llamando a los que estén apuntados (es decir, sean distintos de 0).

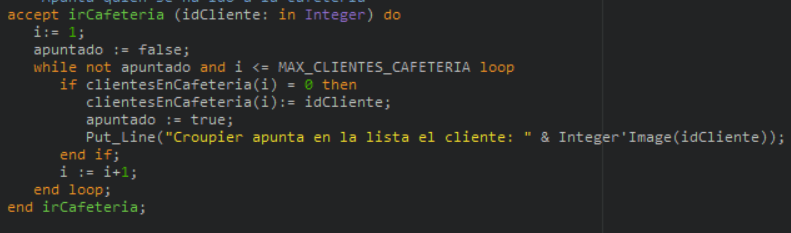


Ilustración : Entrada irCafeteria de la tarea Croupier

En la entrada “irCafeteria”, el croupier apuntará el cliente que se va a la cafetería en el array “clientesEnCafeteria”.

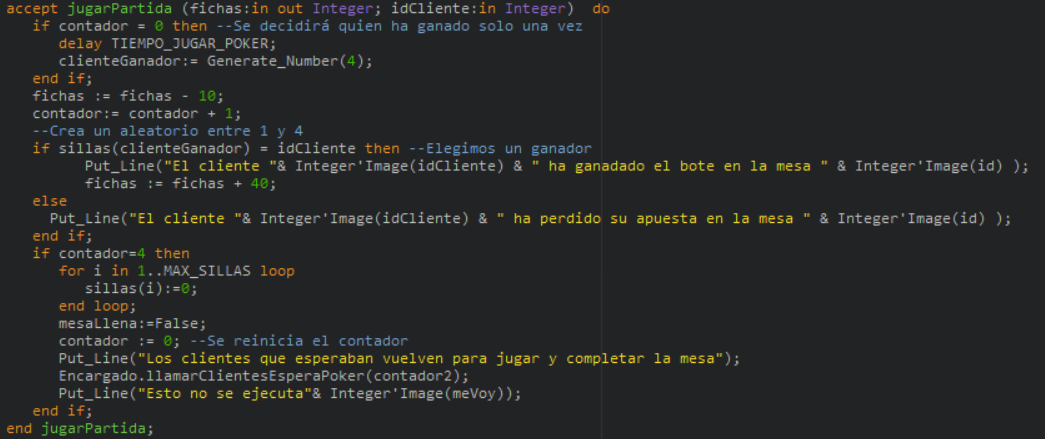


Ilustración : Entrada jugarPartida de la tarea Croupier

La entrada “jugarPartida” es la que simulará como juegan los clientes. Para ello hemos puesto un contador para saber el numero de clientes que ha ejecutado el código. El primer cliente que ejecute el código lo ejecutara con contador=0 y por tanto se asignara el cliente que ha ganado ya con un aleatorio. Después, todos los clientes aumentarán en uno el contador y se le restarán las fichas que apuestan. El cliente ganador recibirá las fichas correspondientes. Cuando se ejecute el último cliente el contador será igual a 4 y por tanto echaremos de las sillas a los clientes que estaban jugando para posteriormente llamar a los clientes que estaban esperando.

## 4.3 Cafetería

4.3.1 Variables locales

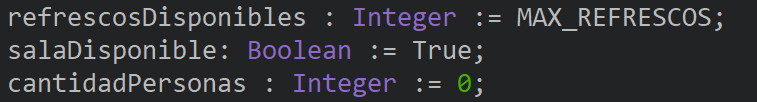


Ilustración : Variables locales

* **RefrescosDisponibles**: Lleva la cuenta de los refrescos disponibles en la cafetería. Cada vez que un cliente quiere un refresco se restará de esta variable.
* **SalaDisponible**: Es la variable booleana que bloquea la entrada “entrarSala” para en caso de estar el encargado limpiando la sala, no se pueda acceder a la sala.
* **CantidadPersonas**: Esta variable contiene la cuenta de personas que entran en la sala para en caso de llenar un máximo, poder llamar al encargado para que limpie la sala.

### 4.3.2 Código

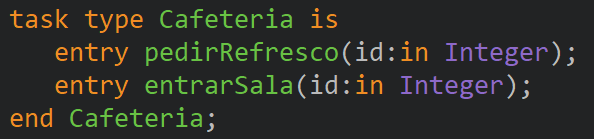


Ilustración :Tarea Cafeteria

Esta tarea se encarga de administrar todo lo relacionado con la cafetería. Se mantiene en un bucle infinito esperando recibir peticiones a sus dos entradas:

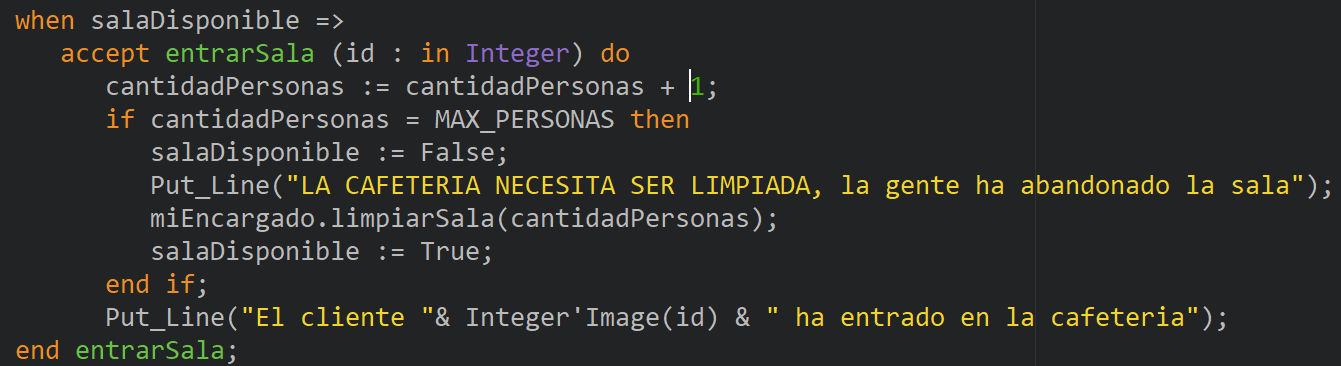


Ilustración : Entrada "EntrarSala" de la tarea "Cafeteria".

Esta entrada se ejecuta siempre y cuando la sala esté disponible.

Una vez se ejecute lo que hace es sumar en un contador que ha pasado una persona más ya que debemos tener una cuenta para que si llega a 10 personas, se avise al encargado para limpiar la sala y no puedan entrar más personas a dicha sala.

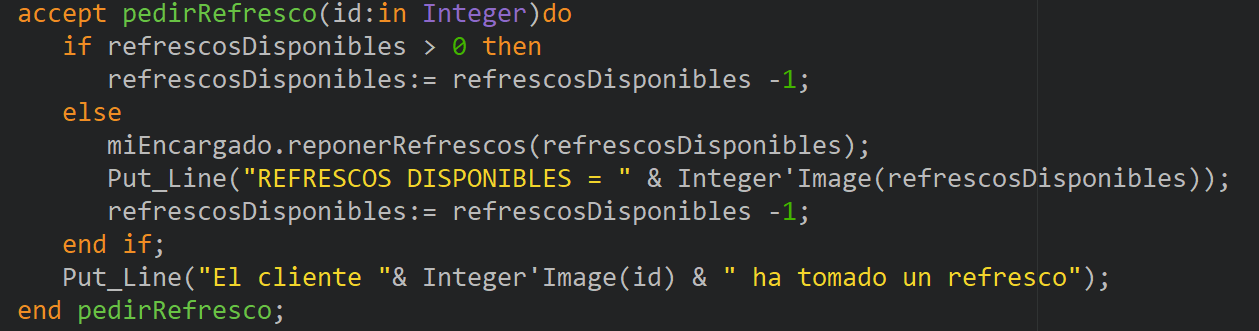


Ilustración : Entrada pedirRefresco de la tarea Cafetería

En esta entrada se comprueba que haya refrescos disponibles, en caso de haber, el cliente cogerá uno, se lo tomará y se irá. En caso de no haber, la cafetería avisa al encargado para que reponga los refrescos.

## 4.4 Recepcionista

### 4.4.1 Variables locales



Esta tarea solo tiene una variable:

* **fichasDisponibles**: Lleva la cuenta de las fichas disponibles en el casino. En caso de gastarse, la recepcionista irá al almacén a por más.

### 4.4.2 Código

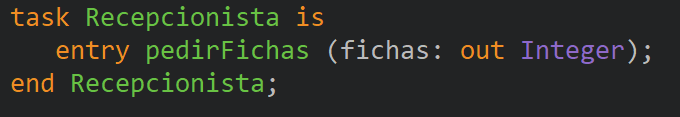


Ilustración : Tarea Recepcionista

Esta tarea se mantiene en un bucle infinito a la espera de atender su única llamada: “pedirFichas”.

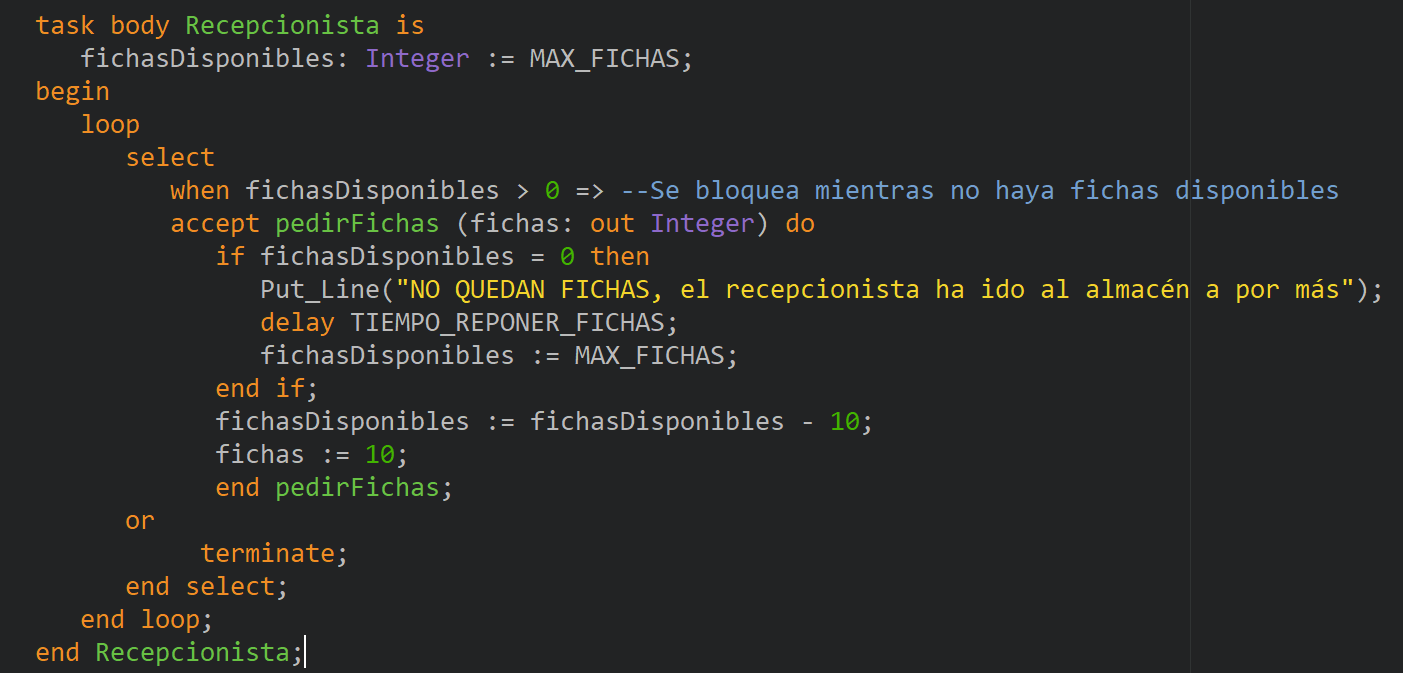
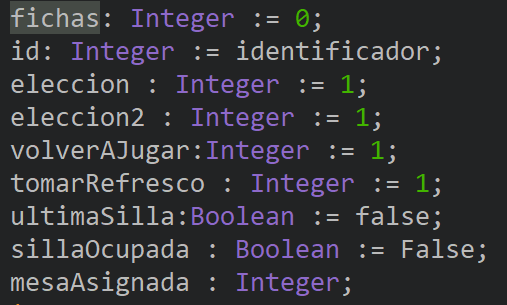


Ilustración : Tarea Recepcionista

En esta llamada se comprueba si el número de fichas es mayor del necesario para entregarle dichas fichas al cliente que ha llamado a esta tarea. En caso de no haber fichas se hará un delay simulando que el recepcionista ha ido al almacén a reponerlas.

## 4.5 Cliente

### 4.5.1 Variables locales

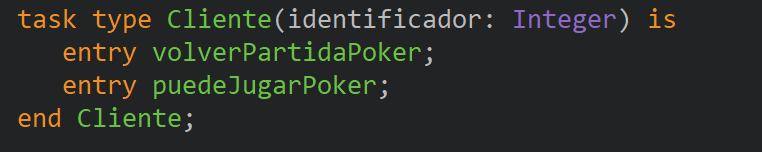


* **Fichas**: Lleva la cuenta de las fichas que tiene cada cliente ya que sin fichas no puede jugar.
* **id**: Almacena el identificador del cliente para llamarlo en las diferentes llamadas entre los procesos o para simplemente identificar cada jugador.
* **eleccion**: En esta variable se decide si el cliente irá de primeras a jugar una partida de poker o no.
* **eleccion2**: Sirve para decidir en caso de no ir al poker o en caso de ir al poker y no encontrar silla, si el cliente quiere ir a las máquinas tragaperras o a tomar un refresco.
* **volverAJugar**: Todo el código de cliente está dentro de un bucle cuya condición es que se cumpla que quiere jugar otra vez.

Para ello inicializamos esta variable a 1 decidiendo que quiere jugar y al final del código generamos un aleatorio que se guarda en esta variable para comprobar si el cliente quiere jugar otra partida.

* **tomarRefresco**: En esta variable se comprueba con un aleatorio si el cliente una vez haya ocupado una silla en el poker quiere ir a tomar un refresco.
* **ultimaSilla**: Sirve para comprobar si un cliente se ha sentado en la última silla de una mesa para en caso de ser verdadera, este cliente no pueda ir a tomar un refresco.
* **sillaOcupada**: Variable booleana que devuelve true si el cliente ha ocupado una silla en alguna mesa del poker o si no la ha ocupado porque están todas las mesas llenas.
* **mesaAsignada**: En esta variable se almacena la mesa asignada (posición del vector croupiers) en caso de que el cliente haya ocupado una silla en alguna mesa.

### 4.5.2 Código



Como esta tarea es muy compleja, vamos a explicarla por partes:

A) Cliente decide ir al poker:

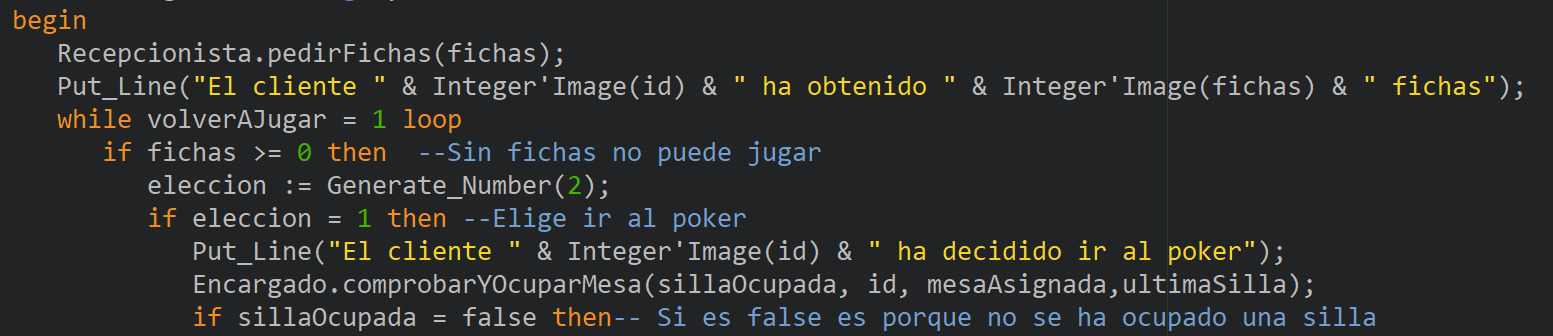


Ilustración : Cliente decide ir al poker.

Lo primero que hace el cliente es ir a recepción a pedir fichas ya que sin fichas no puede jugar al poker. Una vez vuelve elige con un aleatorio si decide jugar al poker o no.

En caso de querer jugar al poker avisa al encargado y este le dice si puede ocupar una silla en alguna mesa o si están todas las mesas llenas.

A.A) Encuentra una silla:

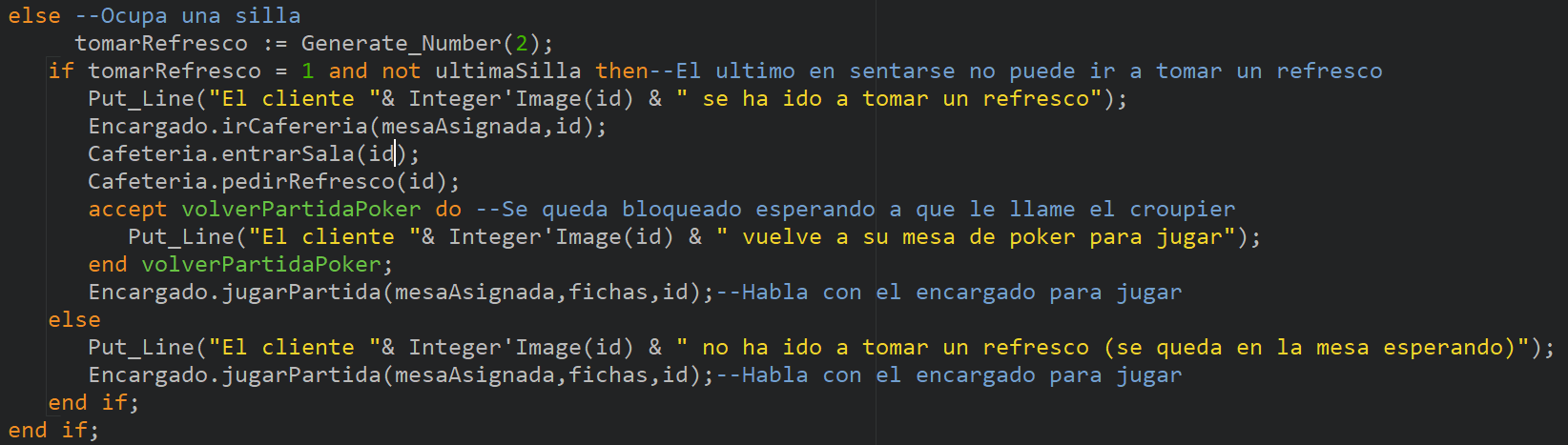


Ilustración : Encuentra una silla

Una vez ocupado una silla se genera un aleatorio que decide si el cliente quiere ir a tomar un refresco o no. En caso de irse a tomar un refresco avisa al encargado de que se va a la cafetería diciendo su identificador y la mesa donde estaba. Después entra en la sala de la cafetería y pide un café.

Por último, se queda bloqueado en una entrada esperando que alguien lo invoque (le avisará el croupier cuando se llene su mesa). Una vez se desbloquee, avisará al encargado que quiere jugar una partida de poker.

En caso de no haber ido a tomar un refresco, simplemente avisa al encargado que quiere jugar la partida y se queda bloqueado esperando.

A.B) No ocupa una silla:

Esta parte del código es demasiado extensa y la hemos dividido en dos partes:

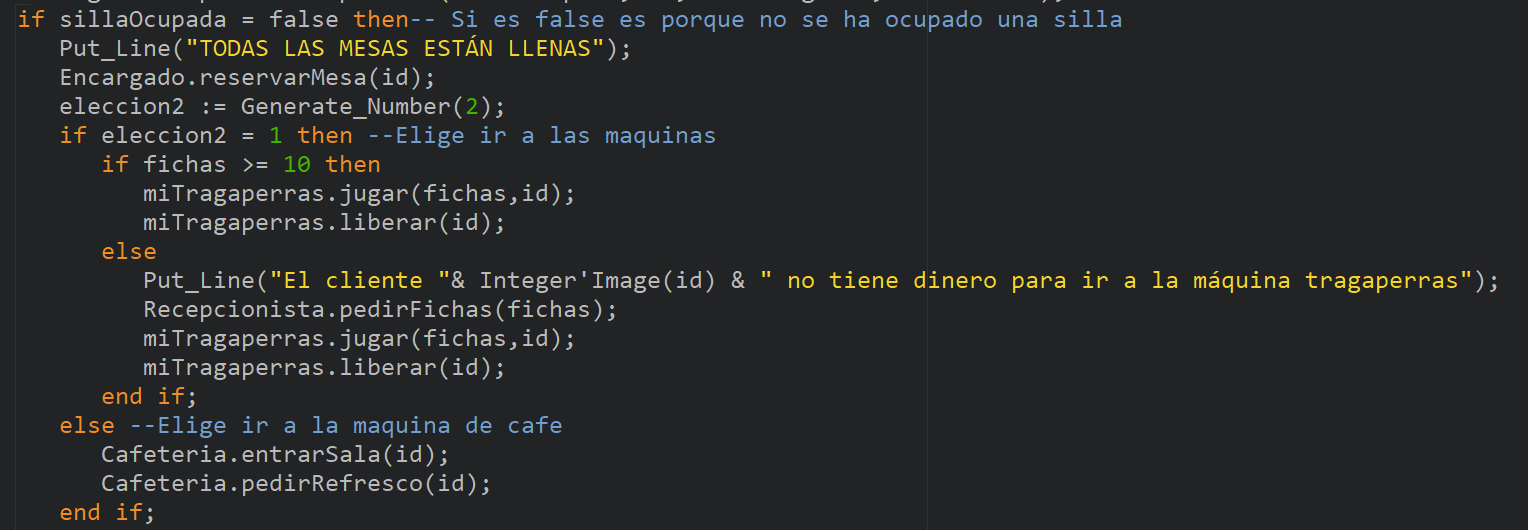


Ilustración : Cliente no ocupa una silla parte 1

Si no puede ocupar una mesa, reserva una silla para cuando se acabe alguna de las partidas que se están celebrándose, el croupier le avise.

Una vez hecho esto decide si quiere ir a las máquinas tragaperras o ir a tomar un refresco. Para simular esto hemos hecho un aleatorio que se almacene en le variable “eleccion2” explicada anteriormente.

Si elige ir a las tragaperras se comprobará que el cliente tiene más fichas de las necesarias (10 fichas) para poder jugar. Para jugar a las máquinas hace dos llamadas, “jugar” y “liberar” en los cuales ocupa y libera una máquina tragaperras.

Si elige ir a la cafetería entra en la sala de dicha cafetería y en caso de poder, pide un refresco.

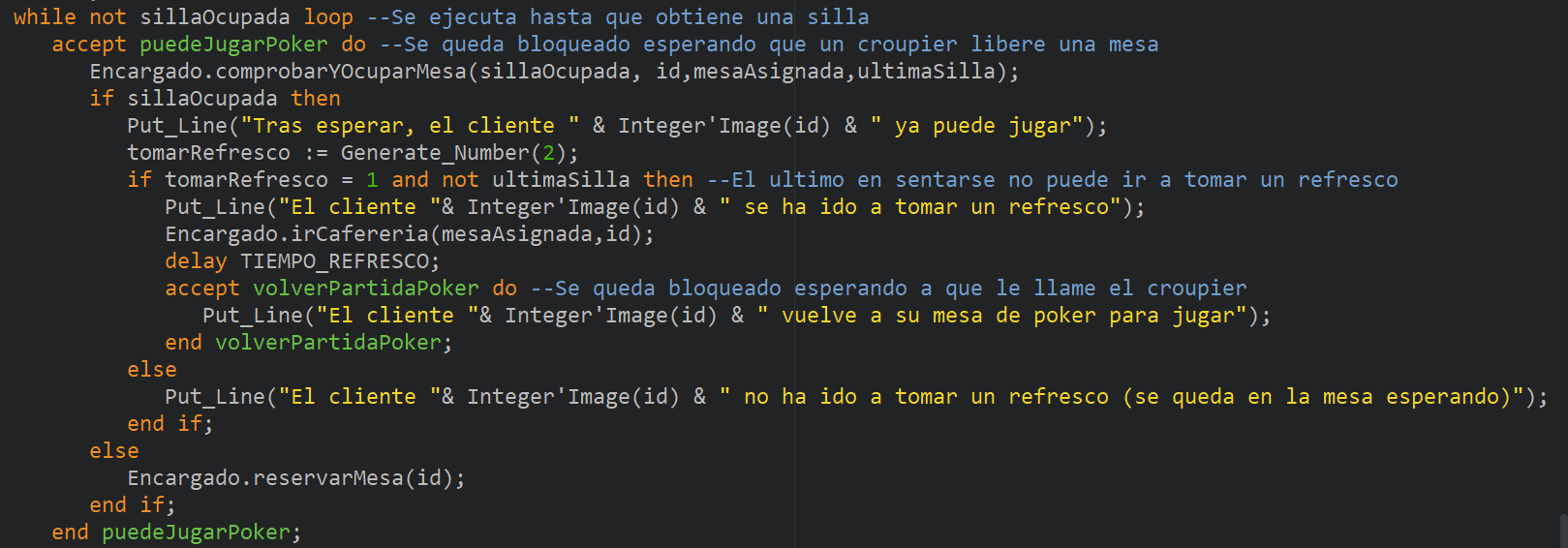


Ilustración : Cliente no ocupa una silla parte 2

Después de ir a la cafetería o a las máquinas tragaperras se mantendrá en un bucle infinito cuya condición es que ocupe una silla (variable sillaOcupada), es decir, hasta que no ocupe el cliente una silla, se ejecutará.

Una vez dentro del bucle, se mantendrá bloqueado esperando a que se pueda jugar ya que anteriormente el cliente reservó una silla. Para esto hemos hecho una entrada llamada “puedeJugarPoker” que será invocada por el croupier de la mesa donde ha reservado.

Una vez desbloqueado comprueba que haya una silla libre ya que aunque haya reservado una silla, puede venir otro cliente y quitársela. En caso de ocupar la silla, el código es igual al ejecutado cuando el cliente encuentra una silla de primeras: Tiene la opción de tomar un refresco o quedarse en la mesa esperando en caso de no ser el último cliente.

En caso de no ocupar tampoco una silla esta vez, vuelve a reservar una silla hablando con el encargado y vuelve al inicio del bucle mencionado previamente.

B) Cliente decide no ir al poker:

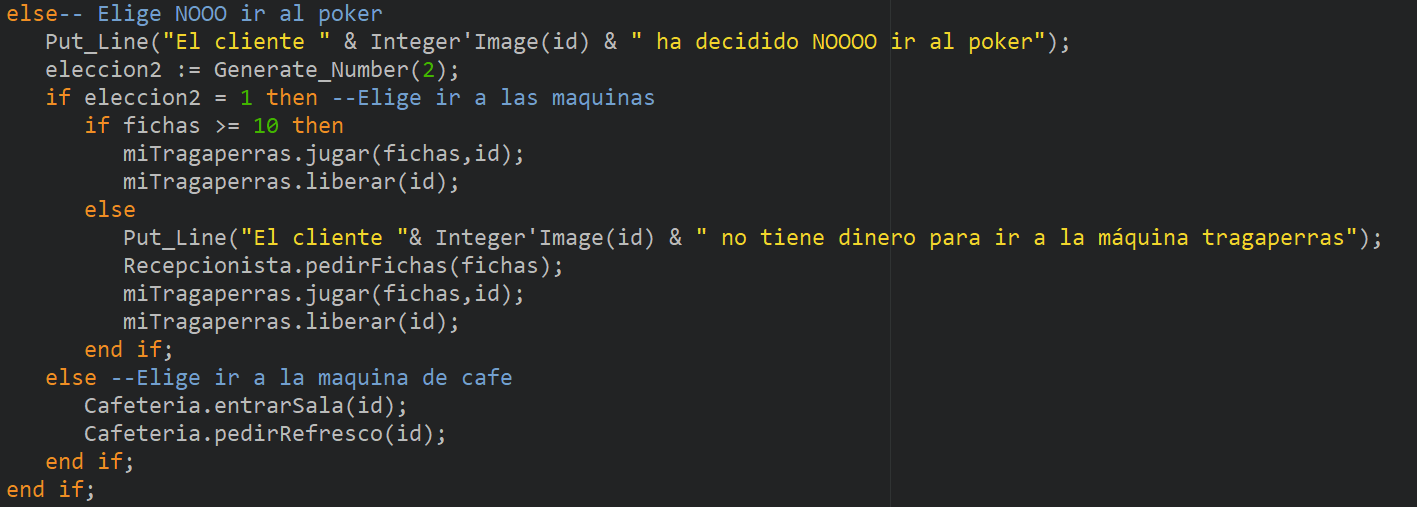


Ilustración : Cliente elige no ir al poker

Este código es exactamente igual al ejecutado en caso de que el cliente haya intentado ocupar una silla, pero no hubiera podido.

En esta parte del código el cliente decide si ir a la máquina tragaperras o a la cafetería.

En caso de ir a la máquina tragaperras, se comprueba que exista fichas suficientes para poder jugar. En caso de poder, ocupa una máquina y juega. Después la libera.

Si elige ir a la cafetería a tomar un refresco primero comprobará si puede entrar en la sala. En caso de poder entrará y pedirá un café.

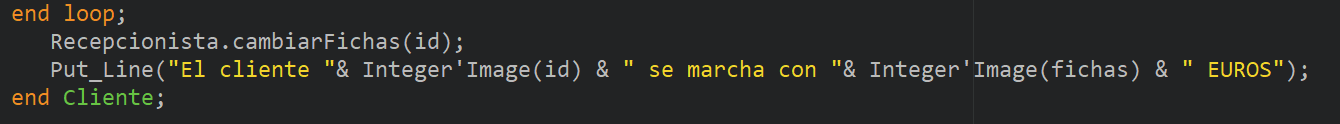


Ilustración : Final cliente

Por último, el cliente cambia las fichas por dinero y se va del casino.