Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

**IIC2233 Programación Avanzada (2022-1)**

# Tarea 1

## Entrega

- **Avance de tarea**
  - **Fecha y hora:** martes 5 de abril de 2022, 20:00
  - **Lugar:** Repositorio personal de GitHub — Carpeta: `Tareas/T1/`
- **Tarea**
  - **Fecha y hora:** miércoles 13 de abril de 2022, 20:00
  - **Lugar:** Repositorio personal de GitHub — Carpeta: `Tareas/T1/`
- `README.md`
  - **Fecha y hora:** miércoles 13 de abril de 2022, 22:00
  - **Lugar:** Repositorio personal de GitHub — Carpeta: `Tareas/T1/`

## Objetivos

- Aplicar conceptos de programación orientada a objetos (POO) para modelar y resolver un problema.
- Utilizar *properties*, clases abstractas y polimorfismo como herramientas de modelación.
- Comunicar diseños orientados a objetos a través de documentación externa.
- Procesar *input* del usuario de forma robusta, manejando potenciales errores de formato.

# Índice

# 1. *DCCasino*

La vuelta a la presencialidad es algo que todo el mundo esperaba, pero en uno de estos tan anhelados días te reencuentras con uno de esos problemas que esperabas nunca tener que resolver: A la entrada de la universidad te encuentras con *BigCat*, una *gangster* que te prestó dinero durante la pandemia para comprar ~~un PC Gamer~~ los cuadernos de tus ramos. Angustiado por no tener como pagarle, buscas una vía rápida para ganar dinero y saldar tus cuentas. Es entonces que recuerdas que el nuevo y flamante *DCCasino* ha abierto en la ciudad, y decides ir a probar tu suerte para salir de esta deuda.


Figura 1: Logo de *DCCasino*

# 2. Flujo del programa

*DCCasino* consiste en una simulación de un Casino, donde el objetivo será recaudar una cantidad de dinero equivalente a `DEUDA_TOTAL`[1]. Para lograr tal hazaña, deberás jugar distintos juegos apostando tu dinero en favor de tratar de ganar más, por lo que siempre estarás en constante riesgo de perder dinero.

La interacción con el programa será exclusivamente mediante los Menús impresos en la consola. En cada **Menú** deberás tomar distintas decisiones que afectarán en tu desempeño dentro del *DCCasino*. Dentro de las acciones posibles, podrás **apostar** en los diferentes **juegos** teniendo como resultado una pérdida o ganancia de **dinero**, junto con un cambio en las emociones del jugador y una pérdida de **energía**. Para recomponer fuerzas y mejorar tus habilidades como apostador, tendrás la opción de ver la **carta de bebestibles**, que te permitirá comprar distintos objetos que te harán recuperar energía y afectarán en tu desempeño dentro del programa. Por último, tendrás la opción de **ver el estado del jugador**, que te mostrará todas las características del estado de tu personaje y te servirán de ayuda para planificar una estrategia adecuada y así recaudar el dinero de la deuda.

Al iniciar el programa se abrirá el Menú de Inicio, donde el jugador puede empezar una nueva partida o salir. En caso de empezar una partida, se deberá seleccionar al jugador que te representará en *DCCasino*, extraídos del archivo `jugadores.csv`. Cada uno de estos personajes tendrá sus propias características, ventajas y desventajas, lo que podría ser perjudicial o beneficiosos dependiendo de tu estilo de juego. Una vez seleccionado tu jugador, se deberá mostrar el Menú Principal, desde el cual se podrá apostar en

[1] Las palabras escritas en `ESTE_FORMATO` son parámetros que tendrás que definir e importar desde el archivo `parametros.py`

algún juego, comprar bebestibles, o ver el estado del jugador. Estará en tus manos tomar las decisiones adecuadas para aumentar tu suerte y poder sobrevivir al *DCCasino*.

# 3. Menús

Para esta tarea, la simulación del *DCCasino* será realizada a través de una serie de Menús en la consola. Estos menús deben ser a prueba de **todo tipo de errores de usuario**, es decir, en caso de que se ingrese un input no válido, se deberá advertir correctamente al usuario y volver a desplegar las opciones del menú. Adicionalmente, cada uno debe tener la opción de **volver atrás y salir**, a menos que se diga lo contrario. A continuación se explicarán los menús mínimos a incluir.

## 3.1. Menú de Inicio

Al iniciar el programa se deberá desplegar un Menú de inicio, en el cual se den las opciones de iniciar **partida y salir del programa**. Dado que es el primer menú que se muestra en pantalla, no es necesario implementar la opción de **volver atrás**.

```
*** Menú de Inicio ***
----------------------
[1] Iniciar partida
[X] Salir
```

### 3.1.1. Opciones de jugador

Si se selecciona `Iniciar partida`, se debe mostrar todos los personajes ubicados en el archivo `jugadores.csv`, seguido de su **tipo de personalidad**. Además, se deberá dar la opción de escoger al personaje que te representará en el *DCCasino*.

Una vez cumplido lo anterior, se dirigirá el usuario al Menú Principal.

```
     *** Opciones de Jugador ***
-----------------------------------
[1] Juampisasez: Ludopata
[2] Emiliax16: Bebedor
[3] mmunoz12: Casual
[4] catalina-arcila: Tacano
[0] Volver
[X] Salir
```

## 3.2. Menú Principal

En este menú se deben encontrar todas las opciones ha realizar en el *DCCasino* para lograr devolver el dinero y pedir perdón a la *gangster BigCat*. Esta sección debe tener tres opciones principales: **Opciones de Juegos**, **Carta de Bebestibles** y **Ver estado del Jugador**.

```
   *** Menú Principal ***
--------------------------
[1] Opciones de juegos
[2] Comprar bebestible
[3] Habilidades jugador
[0] Volver
[X] Salir
```

### 3.2.1. Opciones de Juegos

Al seleccionar esta opción, se deberá desplegar todos los nombres de los juegos existentes en el *DCCasino*, los cuales estarán ubicados en el archivo `juegos.csv`. Al momento de seleccionar un juego, se preguntará la cantidad de dinero que se desea apostar. Esta cantidad deberá respetar los **límites de apuesta** señalados en la sección de Juego. En caso de que el jugador **no posea** suficiente dinero, o la apuesta no se encuentre entre los límites establecidos, se deberá **notificar** en la consola y redirigir al menú anterior. Por otra parte, si la apuesta es válida, se deberá calcular la probabilidad de ganar [2] en base a la expresión especificada en la sección Juego. Una vez que ganes o pierdas, se deberán aplicar las medidas señaladas en Juego y deberás **volver** al **Menú Principal**.

```
   *** Opciones de Juegos ***
 -------------------------------
   [1] Cara o Sello
   [2] Ruleta de la suerte
   [3] Poker
   [4] Black Jack
   [0] Volver
   [X] Salir
```

### 3.2.2. Carta de Bebestibles

Es importante que al ser seleccionado un bebestible se presente la información de este, que debe contener como mínimo su **nombre**, el **tipo** al que pertenece (Jugo, gaseosa o brebaje mágico) y el **precio**. Además, deberás dar las opciones de **volver** y **salir**. Una vez seleccionado el bebestible, se deberá verificar que el jugador posea el **dinero suficiente** para comprar el producto. En caso de ser válido, se deberá **descontar** el dinero al jugador y aplicar los **efectos** del bebestible señalados en Bebestibles. En caso de **no poseer** suficiente dinero, se deberá **notificar** en la consola y volver a desplegar la carta de bebestibles.

```
          *** Bebestibles ***
 ------------------------------------
  N° |  Nombre         | Tipo     |  Precio
 [1] |  Gatorate       | Jugo     |  $119
 [2] |  Sprite         | Gaseosa  |  $151
 [3] |  Jugostamm      | Jugo     |  $149
 [4] |  Orina de gato  | Jugo     |  $139

 [0] Volver
 [X] Salir
```

### 3.2.3. Ver estado del Jugador

Al seleccionar esta opción, se deberá desplegar en consola toda la información relevante del estado del jugador. Esta deberá incluir como mínimo: el **nombre**, **personalidad**, **energía**, **suerte**, **dinero**, **frustración**, **ego**, **carisma**, **confianza** y el **juego favorito**. Además, deberás mostrar el **dinero faltante** para poder saldar la deuda hacia *BigCat*. Puedes incluir más información si lo consideras relevante.

[2] Para ver si la probabilidad fue exitosa o no, puedes hacer uso del método random.random y comparar el valor con la probabilidad de éxito

```
        *** Ver estado del Jugador ***
-------------------------------------------
    Nombre: mmunoz12
    Personalidad: Casual
    Energía: 56
    Suerte: 3
    Dinero: 282
    Frustración: 34
    Ego: 15
    Carisma: 18
    Confianza: 23
    Juego favorito: blackjack
    Dinero faltante: $3600
    [0] Volver
    [X] Salir
```

Cabe recordar que todos los menús presentados en las imágenes anteriores son **representativos**, por lo cual puedes realizar todas las modificaciones que prefieras mientras cumplas con lo **mínimo pedido** en el enunciado.

# 4. Entidades

En esta sección se detallarán las entidades que necesitarás para simular *DCCasino*. Deberás utilizar conceptos claves de **Programación Orientada a Objetos** como herencia, clases abstractas, polimorfismo, *properties* y relaciones, que pueden ser de agregación o composición. Ten en cuenta, que cada uno de estos elementos debe ser incluido en el programa al menos una vez, y deberás descubrir dónde implementarlos según lo propuesto por el enunciado.

Las entidades principales de *DCCasino* son **Jugadores, Casino, Juegos y Bebestibles**. Debes incluir como mínimo las características nombradas a continuación, pero siéntete libre de añadir nuevos atributos y métodos si lo estimas necesario.

## 4.1. Casino

Es el mapa general donde se jugara la partida de *DCCasino*. En esta tendrás al jugador, los juegos y los bebestibles, por lo que deberás interactuar entre las entidades para ir jugando. A continuación, se enunciarán las características de la entidad **Casino**.

- **Jugador**: Jugador elegido para participar en el casino y así pagar su deuda.
- **Bebestibles**: Conjunto de los distintos bebestibles que se puede tomar el jugador.
- **Juegos**: Conjunto de juegos de azar con los que interactúa el jugador.
- **Dinero faltante**: Corresponde a un `int` que tiene como objetivo mostrar el dinero que te falta para terminar de pagar la deuda

El casino puede realizar las siguientes acciónes:

- **Evento especial**: El casino tiene la posibilidad de que haya un evento especial después de saber el resultado de un juego. El evento que se genere tiene una probabilidad de ocurrencia de `PROBABILIDAD_EVENTO` y consistirá en entregarle un bebestible **aleatorio** gratis al jugador, el cual

deberá ser consumido de forma **inmediata**. Esto implica que luego de que el jugador haya apostado y sepa el resultado, se deberá notificar si ocurrió el evento especial o no, de acuerdo con la probabilidad de ocurrencia.

- **Jugar**: Esta acción consiste en simular el evento de un juego. Cada juego posee características distintas, por lo que deberás realizar una serie de acciones para poder llevar a cabo el proceso de una apuesta y aplicar los cálculos y resultados correspondientes.

### 4.2. Jugador

Los jugadores son los personajes de *DCCasino*. Interactúan en el **casino**, donde pueden jugar distintos **juegos**, y tomar distintos **bebestibles** durante la simulación del *DCCasino*. El objetivo del jugador es conseguir el dinero necesario para pagar la deuda que tienes con *BigCat*. Para lograr este objetivo, deberás ir jugando a los distintos juegos de azar y tener cuidado de no quedar sin dinero.

Al ir apostando en el *DCCasino*, el jugador irá perdiendo **energía** y esta puede ser recuperada a través de **bebestibles** que se pueden comprar con **dinero** en el bar. Ten en cuenta que el jugador comenzará con ciertas características iniciales, pero estas deberán **mantenerse** dentro del rango establecido para cada una de ellas. Por último, el jugador podrá tener distintas personalidades y cada una de estas tendrá una habilidad especial que podrá ocupar en distintos momentos de la simulación. A continuación, se presentarán las características del jugador:

- **Nombre**: Corresponde a un `str` que representa el nombre del jugador.
- **Energía**: corresponde a un `int` entre `0` y `100` que representa la cantidad de energía del jugador.
- **Suerte**: Corresponde a un `int` `0` y `50` que va a afectar en las posibilidades de ganar en el juego.
- **Dinero**: Corresponde a un `int`, que sera ocupado para apostar, comprar bebestibles, para ganar o perder. Este valor no puede ser negativo.
- **Frustración**: Corresponde a un `int` entre `0` y `100` que puede afectar en el calculo de cuanta energía se gasta al jugar.
- **Ego**: Corresponde a un `int` entre `0` y `15`. Este se ocupa para calcular cierto cambios al apostar.
- **Personalidad**: Corresponde a un `str`, que posee el tipo de personalidad del jugador.
- **Juegos jugados**: corresponde a una `list` que debe poseer los juegos, que se han jugado de manera ordenada desde más antiguo al último jugado.
- **Carisma**: corresponde a una `int` entre `0` y `50`. Este afecta en el calculo de la probabilidad de ganar.
- **Confianza**: corresponde a una `int` entre `0` y `30`. Este afecta en el calculo de la probabilidad de ganar.
- **Juego favorito**: este atributo consiste en un `str`, el cual es el juego favorito del jugador.

Además, el jugador puede realizar las siguientes acciones:

- **Comprar bebestible**: Está acción permite comprar y utilizar los distintos bebestibles que te entregaran distintos beneficios que se pueden ver en la sección de bebestibles. Esta acción solo se debe poder hacer si tienes dinero suficiente para comprar un bebestible. En caso contrario, se deberá advertir correctamente al usuario y volver al menú anterior.
- **Apostar**: Esta acción te permitirá jugar alguno de los juegos donde deberás apostar cierta cantidad de dinero. Para ello, debes calcular una probabilidad de ganar y ver si ocurre o no. En caso de tener éxito, se **doblará** el dinero apostado. En caso contrario, **perderás** el dinero apostado.

Además, los jugadores al apostar gastaran cierta cantidad de energía, esta deberá ser restada luego de jugar cada juego. Esta perdida se calcula a partir de la siguiente formula:

$$\texttt{round}^{3}(Ego + Frustracion) * \texttt{0.15})$$

- **Ego**: Es la característica de Ego del jugador. Se encuentra en el archivo `jugadores.csv`
- **Frustración**: Es la característica de Frustración del jugador. Se encuentra en el archivo `jugadores.csv`

Por otro lado, cada jugador tiene una probabilidad de ganar según sus atributos personales y el valor de su apuesta. Esta se calcular a partir de la siguiente formula:

$$\texttt{min}(1, \texttt{max}(0, \frac{(Suerte * \texttt{15} - Apuesta * \texttt{0.4} + Confianza * \texttt{3} * Favorito + Carisma * \texttt{2})}{\texttt{1000}}))$$

- **Suerte**: Es la característica de Suerte del jugador. Se encuentra en el archivo `jugadores.csv`.
- **Confianza**: Es la característica de Confianza del jugador. Se encuentra en el archivo `jugadores.csv`
- **Carisma**: Es la característica de Carisma del jugador. Se encuentra en el archivo `jugadores.csv`
- **Favorito**: Valor que es `1` si es su juego favorito.
- **Apuesta**: Corresponde a la apuesta realizada por el jugador.

### 4.2.1. Tipos de personalidad de jugadores

Además, existen distintos tipos de personalidades, cada una con diferentes parámetros y acciones que pueden mejorar al jugador.

- **Ludópata**: Los ludópatas al ser personas adictas a las apuestas poseen una acción especial que les modificará sus características. Esta acción se llama `ludopatia`, el cual aumenta el ego del jugador en `2` y la suerte en `3` cada vez que realice alguna apuesta, independiente del resultado que obtenga. Sin embargo, si el jugador pierde, la frustración aumentará en `5`. Ambos efectos deberán aplicarse luego de que el jugador haya apostado y sepa su resultado.
- **Tacaño**: Los Tacaños al ser personas avaras, tienen un trato con el *DCCasino* donde cada vez que apuestan, se debe aplicar su acción. La acción de los Tacaños se llama `tacano_extremo`, la cual consiste en que cada vez que el jugador apueste menos que el `PORCENTAJE_APUESTA_TACANO` de su dinero actual, tendrá una bonificación de `BONIFICACION_TACANO` de dinero si es que gana la apuesta.
- **Bebedor**: Este jugador al ser adicto a beber distintos tipos de bebidas, jugos y brebajes de dudosa procedencia, posee una acción que le entrega un `MULTIPLICADOR_BONIFICACION_BEBEDOR` extra en las bonificaciones entregadas por las bebidas. Esto implica que todos los efectos producidos por una bebida deberán ser multiplicados por este parámetro. Esta acción se debe llamar `cliente_recurrente`.
- **Casual**: Este es un jugador principiante, por esta razón, si está apostando por primera vez en cualquier juego, conseguirá una bonificación de `BONIFICACION_SUERTE_CASUAL` de suerte que se mantendrá a lo largo de su estadía en el *DCCasino*. Esta acción solo podrá ser ejecutada una única vez en su primera apuesta y se llama `suerte_principiante`.

[3] Para que la energía conserve su valor de `int`, deberás redondear el resultado utilizando este `built-in`.

### 4.3. Juego

La clase juego en el *DCCasino* consiste en la clase que posee a los distintos juegos que poseerá la simulación, en estos juegos el jugador deberá apostar para así pagar la deuda y poder ganar el juego. Dentro de estos juegos existirá una esperanza que sera la probabilidad de lograr ganar en el juego. A continuación estas serán los atributos y acciones que deberás crear y la información estará en el archivo `juegos.csv`.

- **Nombre**: Corresponde al nombre del juego en formato `str` .
- **Esperanza**: Corresponde a la esperanza de ganar en el juego y esta en el formato `int`.
- **Apuesta mínima**: Corresponde a la apuesta mínima para poder jugar. Estará en formato `int`.
- **Apuesta máxima**: Corresponde a la apuesta máxima para poder jugar. Estará en formato `int`.

Además, cada Juego tendrá las siguientes acciones:

- **Entregar resultados**: Esta acción se debe encargar de aplicar todos los cambios a los atributos del jugador después de terminar los juegos. En caso de **perder**: Aumentara la frustración del jugador en `FRUSTRACION_PERDER`, bajará la confianza en `CONFIANZA_PERDER` y se deberá restar la cantidad de dinero apostado. En caso de **ganar**: Aumentará el ego del jugador en `EGO_GANAR`, el carisma aumentará en `CARISMA_GANAR` y la frustración bajará en `FRUSTRACION_GANAR`.
- **Probabilidad de ganar**: En esta acción se deberá calcular la probabilidad de ganar el juego, según la siguiente fórmula:

$$\texttt{min}(1, Probabilidad\_ganar\_jugador - (\frac{Apuesta - (Favorito * \texttt{50} - (Esperanza * \texttt{30})}{\texttt{10000}}))$$

- **Esperanza**: Es el atributo que posee cada juego y simboliza la esperanza del juego.
- **Apuesta**: Es la cantidad de **dinero** apostada en el juego por el jugador.
- **Favorito**: valor que es `1` si es su juego favorito y `0` si no es su juego favorito.
- **Probabilidad ganar jugador**: Es la probabilidad que tiene de ganar el jugador en base a sus características y el valor de la apuesta.

### 4.4. Bebestibles

Los bebestibles en *DCCasino* juegan un rol muy importante, ya que estos te entregan un aumento en alguna de las características del jugador. Los bebestibles pueden ser de tipo **Gaseosa, Jugos o Brebaje mágico**. Por último, los atributos de esta entidad se encuentran en `bebestibles.csv`.csv y son:

- **Nombre**: Corresponde a un `str` que representa el nombre del bebestible.
- **Tipo**: Corresponde a un `str` que representa el tipo de bebestible que esta tomando que puede ser **Gaseosa, Jugo o Brebaje mágico**.
- **Precio**: Corresponde a un `int` que representa el valor monetario para comprar el brebaje al tratar de comprar en tu turno.

Además, deberás aplicar su beneficio al jugador mediante la siguiente acción.

- **Consumir**: Esta acción recibe la entidad del jugador a la cual va a aplicarle los beneficios del bebestible. Estos beneficios **dependerán** del **tipo** de bebestible. Sin embargo, todos los bebestibles recuperarán un valor de energía **aleatorio** entre `MIN_ENERGIA_BEBESTIBLE` y `MAX_ENERGIA_BEBESTIBLE`, incluyendo extremos. Deberás informar por consola cada uno de los beneficios que se le apliquen al jugador indicando la característica afectada y su magnitud.

#### 4.4.1. Tipos de Bebestibles

Además, existen distintos tipos de bebestibles, cada una contribuyendo con distintas cualidades al jugador.

- **Jugo**: Los jugos son un tipo de bebestible muy natural, y por esta razón generalmente entregan muchos nutrientes que afectan de distintas maneras a cada persona. Estos cambios dependerán del largo de los nombres de los jugos. Si el largo del nombre es de `4` o menos aumentará tu ego en `4`, si el nombre está entre `5` y `7` tu suerte aumentará en `7` y si tiene `8` o más caracteres tu frustración disminuirá en `10`, pero tu ego aumenta en `11`.
- **Gaseosa**: Es un bebestible muy dulce que generalmente lo tomas los días con calor con unos cubos de hielo. Existen distintas marcas con distintos sabores. Por esta razón, producen distintos cambios en las características de una persona según su personalidad. Si la personalidad del jugador es tacaño o ludópata, te disminuirá la frustración en `5`, pero si eres alguien bebedor o casual aumentará tu frustración en `5`. Finalmente aumentará tu ego en `6`, independiente de la personalidad.
- **Brebaje mágico**: Los brebajes mágicos son una fusión de sabores y de líquidos. Estas son muy fuertes y generalmente afectan de maneras mágicas a cada persona. Los efectos que se aplican son tanto los de **Gaseosa** como la de **Jugo**. Además, producirán un aumento de `5` en carisma.

## 5. Archivos

Los siguientes archivos contienen la base de datos de los distintos juegos, jugadores y bebestibles disponibles en *DCCasino*. Todos estos archivos los necesitarás para tu programa. Podrás notar que cada uno de los siguientes archivos viene con un encabezado (*header*) en la primera línea que indica a cuál columna corresponde cada uno de los elementos de las siguientes filas, separadas por comas (`","`). Ten en cuenta que el encabezado muestra el nombre de la columna de los elementos, por lo que tu programa debe **responder correctamente** en caso de utilizar otros archivos con **distinto**.

### 5.1. `juegos.csv`

Este archivo contiene todas las opciones de juegos que el usuario podrá seleccionar para la partida de *DCCasino*, junto con sus características que son descritas en la siguiente tabla:

| Nombre | Tipo de Dato | Descripción |
|---|---|---|
| Nombre | `str` | Indica el nombre del juego. |
| Esperanza | `int` | Corresponde a la esperanza de ganar el juego. |
| Apuesta Mínima | `int` | Indica la apuesta mínima del juego. |
| Apuesta Máxima | `int` | Indica la apuesta máxima del juego. |

Un ejemplo del archivo `juegos.csv` es el siguiente:

```
nombre,esperanza,apuesta minima,apuesta maxima
cara o sello,50,0,100
ruleta de la suerte,20,40,230
poker,45,20,2000
blackjack,13,300,500
```

### 5.2. `jugadores.csv`

Este archivo contiene todas las opciones de jugadores que el usuario podrá seleccionar para comenzar una partida de *DCCasino*, junto con sus características que son descritas en la siguiente tabla:

| Nombre | Tipo de Dato | Descripción |
|---|---|---|
| Nombre | `str` | Indica el nombre del jugador |
| Personalidad | `str` | Indica el tipo del jugador |
| Energía | `int` | Indica la energía del jugador. |
| Suerte | `int` | Indica la suerte del jugador en cada juego. |
| Dinero | `int` | Indica el dinero del jugador. |
| Frustración | `int` | Indica el nivel de frustración. |
| Ego | `int` | Indica el ego del jugador. |
| Carisma | `int` | Indica la carisma del jugador. |
| Confianza | `int` | Indica la confianza del jugador. |
| Juego favorito | `str` | Indica el juego favorito del jugador |

Un ejemplo del archivo `jugadores.csv` es el siguiente:

```
nombre,personalidad,energia,suerte,dinero,frustracion,ego,carisma,confianza,
juego favorito
jtvaldivia,Ludopata,23,2,777,9,3,5,6,2632,93,9,10,19,cara o sello
Emiliax16,Bebedor,89,32,4954,48,8,46,15,ruleta de la suerte
drcid98,Casual,56,3,282,34,15,18,23,blackjack
catalina-arcila,Tacano,18,50,5097,45,14,17,11,blackjack
Maxy15,Tacano,44,5,6697,73,1,27,16,ruleta de la suerte
```

### 5.3. `bebestibles.csv`

Este archivo contiene la información de todos los bebestibles que existirán dentro del juego/nicoes confuso usar juego acá, mejor usar DCCasino. Cada consumible tiene un nombre, tipo de bebestible (gaseosa, jugo o brebaje mágico), y su precio.

| Nombre | Tipo de Dato | Descripción |
|---|---|---|
| Nombre | `str` | Indica el nombre del bebestible. |
| Tipo | `str` | Indica el tipo de bebestible. Puede ser `'gaseosa'`, `'jugo'` o `'bebraje mágico'`. |
| Precio | `int` | Indica el precio del bebestible. |

Un ejemplo del archivo `bebestibles.csv` es el siguiente:

```
nombre,tipo,precio
gatorate,Jugo,119
sprite,Gaseosa,151
jugostamm,Jugo,149
orina de gato,Jugo,139
```

### 5.4. `parametros.py`

Para esta tarea, deberás crear un archivo `parametros.py` y completarlo con todos los parámetros mencionados a lo largo del enunciados, los cuales encontrarás en `ESTE_FORMATO` y en ese color. Además, debes agregar cualquier valor constante en tu tarea, junto con los *paths* que utilices.

Recuerda que los parámetros deben ser descriptivos y reconocibles:

```
CIEN = 100 # mal parámetro
DEUDA_TOTAL = 3000 # buen parámetro
PROBABILIDAD_EVENTO = 0.2 # buen parámetro
```

Si necesitas agregar algún parámetro que varíe de acuerdo a otros parámetros, una correcta parametrización sería la siguiente:

```
PI = 3.14
RADIO_ESFERA = 3
VOLUMEN_ESFERA = (4/3) * PI * (RADIO_ESFERA ** 3)
```

Dentro del archivo `parametros.py` deberás hacer uso de todos los parámetros almacenados y deberás importarlos correctamente. Si se almacena cualquier información no correspondiente a parámetros, esto tendrá un descuento en tu nota. Por último, no está permito que un parámetro se encuentre ***harcodeado***[4], ya que es una mala práctica y su uso conlleva un descuento.

Para esta tarea, el archivo `parametros.py` no se debe ignorar y debes subirlo a tu repositorio. En caso contrario, tu tarea no se podrá corregir.

## 6. *Bonus*

En esta tarea habrá una serie de *bonus* que podrás obtener. Cabe recalcar que necesitas cumplir los siguientes requerimientos para poder obtener *bonus*:

1. La nota en tu tarea (sin bonus) debe ser **igual o superior a 4.0**[5].
2. El bonus debe estar implementado **en su totalidad**, es decir, **no se dará puntaje intermedio**.

Finalmente, la cantidad máxima de décimas de *bonus* que se podrá obtener serán 3 décimas. Deberás indicar en tu `README` si implementaste alguno de los bonus, y cuáles fueron implementados.

### 6.1. Show (2 décimas)

Para optar a este bonus se debe implementar una nueva acción en el Menú Principal, llamada ver **Show**. Para esto, se debe crear la entidad **Show**. Al ser seleccionada, se deberá pagar un costo fijo de `DINERO_SHOW` en dinero, pero a cambio aumentará la energía del jugador en `ENERGIA_SHOW` y disminuye su frustración en `FRUSTRACION_SHOW`. Deberás informar al usuario por consola de los beneficios entregados, y advertir en caso de no tener el suficiente dinero.

[4] *Hard-coding* es la mala práctica de incrustar datos directamente en el código fuente del programa, en vez de obtener los datos de una fuente externa.

[5] Esta nota es sin considerar posibles descuentos.

## 7. Avance de tarea

En conjunto con el programa, deberás realizar un **diagrama de clases** modelando las entidades necesarias del *DCCasino*. Este diagrama se entregará en dos ocasiones:

Una versión preliminar que refleje cómo planeas modelar tu programa, que corresponderá al **avance** de esta tarea. A partir de los avances entregados, se te brindará un *feedback* general de lo entregado y además, te permitirá optar por **hasta 2 décimas** adicionales en la nota final de tu tarea.

Luego de esto, junto a la entrega final, deberás entregar una **versión final** de tu diagrama que **represente fielmente** la modelación del problema usada en tu programa.

En ambos casos, el diagrama deberá:

- Entregarse en **formato PDF o de imagen**[6].
- Contener todas las clases junto con sus atributos y métodos. También deberás identificar cuáles clases serán abstractas y cuáles no.
- Contener todas las relaciones existentes entre las clases (agregación, composición y herencia).
- No es necesario indicar la cardinalidad ni la visibilidad (público o privado) de los métodos o atributos.

Para realizar el diagrama de clases te recomendamos utilizar **draw.io**, **lucidchart** o aplicaciones similares.

Es conveniente adjuntar a tu diagrama un documento con una explicación general de tu modelación. Esto es con el fin de ayudar en la corrección del ayudante a comprender tu razonamiento.

Tanto el diagrama (en formato PDF o de imagen) como la explicación de su modelación (en formato **Markdown**) deben ubicarse en la misma carpeta de entrega de la tarea.

## 8. `.gitignore`

Para esta tarea **deberás utilizar un** `.gitignore` para ignorar los archivos indicados, este deberá estar dentro de tu carpeta `Tareas/T1/`. Puedes encontrar un ejemplo de `.gitignore` en el siguiente ***link***.

Para esta tarea, los archivos a ignorar corresponden a las bases de datos entregadas para la simulación. Es decir, deberás ignorar:

- `Enunciado.pdf`
- `jugadores.csv`
- `juegos.csv`
- `bebestibles.csv`

Se espera que no se suban archivos autogenerados por las interfaces de desarrollo o los entornos virtuales de Python, como por ejemplo: la carpeta `__pycache__`.

Para este punto es importante que hagan un correcto uso del archivo `.gitignore`, es decir, los archivos no **deben** subirse al repositorio debido al archivo `.gitignore` y no debido a otros medios.

[6] Cualquier otro formato no será considerado como una entrega válida y no tendrá décimas de avance o puntaje en la tarea.

## 9. Entregas atrasadas

Posterior a la fecha de entrega de la tarea se abrirá un formulario de Google Form, en caso de que desees que se corrija un *commit* posterior al recolectado, deberás señalar el nuevo *commit* en el *form*.

El plazo para rellenar el *form* será de 24 horas, en caso de que no lo contestes en dicho plazo, se procederá a corregir el *commit* recolectado.

## 10. Importante: Corrección de la tarea

Para esta tarea, el carácter funcional del programa será el pilar de la corrección, es decir, **sólo se corrigen tareas que se puedan ejecutar**. Por lo tanto, se recomienda hacer periódicamente pruebas de ejecución de su tarea y *push* en sus repositorios.

Cuando se publique la distribución de puntajes, se señalará con color **amarillo** cada ítem que será evaluado a nivel de código, todo aquel que no esté pintado de amarillo significa que será evaluado si y sólo si se puede probar con la ejecución de su tarea.

En tu archivo `README.md` deberás señalar el archivo y la línea donde se encuentran definidas las funciones o clases relacionados a esos ítems.

Finalmente, si durante la realización de tu tarea se te presenta algún problema o situación que pueda afectar tu rendimiento, no dudes en contactar al ayudante jefe de Bienestar al siguiente correo: **bienestar.iic2233@ing.puc.cl**.

## 11. Restricciones y alcances

- Esta tarea es **estrictamente individual**, y está regida por el **Código de honor de Ingeniería**.
- Tu programa debe ser desarrollado en Python 3.8.
- Tu programa debe estar compuesto por uno o más archivos de extensión `.py`.
- Si no se encuentra especificado en el enunciado, supón que el uso de cualquier librería Python está prohibido. Pregunta en la *issue* especial del **foro** si es que es posible utilizar alguna librería en particular.
- Debes adjuntar un archivo `README.md` **conciso y claro**, donde describas los alcances de tu programa, cómo correrlo, las librerías usadas, los supuestos hechos, y las referencias a código externo. **Tendrás hasta 2 horas después del plazo de entrega** de la tarea para subir el `README` a tu repositorio.
- Tu tarea podría sufrir los descuentos descritos en la **guía de descuentos**.
- Entregas con atraso de más de 24 horas tendrán calificación mínima (1,0).
- Cualquier aspecto no especificado queda a tu criterio, siempre que no pase por sobre otro.

Las tareas que no cumplan con las restricciones del enunciado obtendrán la calificación mínima (1,0).