***Analisis de requisitos***

Se espera un software que emule la gestion de un casino y que cumpla los siguientes requisitos

* Debe estar equlibrado para ofrecer un juego entretenido
* Se debe poder guardar y cargar la partida correctamente
* En caso de instalar maquinas poder cancelar si no nos gusta el lugar
* Que se realice de forma grafica mediante displays y pulsaciones del teclado.

***INDICE***

**Tycoon.py 2**

**Clases\_tycoon.py 3--4**

**Display\_tycoon.py 5--9**

TYCOON.PY

MAIN

IMPORTAMOS

Xa que funcione el programa correctamente necesitamos importar playsound para reproducir sonidos ,os para utilizar la lambda clear, que nos dejara la pantalla a 0 para que no se acumule lo que printeamos y listener, que hara posible capturar las teclas pulsadas

Importamos los 2 modulos propios , se encuentran en estos las clases necesarias para el funcionamiento del programa.

from clases\_tycoon import \*

from display\_tycoon import \*

estructura:

SIGUIENTE

ESTADO

VER MAPA

VER INFORME

COMPRAR

COMPRAR MAQUINA

COMPRAR DECORACION

CONFIGURACION

GUARDAR

CARGAR

SALIR

Tenemos un bucle while en linea 39 que hace que nos desplacemos por los menus hasta que decidamos salir.

If, elif y else se encargan de condicionar la entrada a los menus según nuestra decision en los displays

Conociendo esto y viendo los objetos creados casino de clases\_tycoon.py y display de display\_tycoon.py podemos encontrar facilmente mediante esta documentacion de donde proviene cada funcion. Asi pues x ej si queremos ver

display.opciones\_estado() vemos que es de display, vamos a class display\_tycoon y buscamos dicho metodo , veremos que hace y con que metodos interactua.

CLASES\_TYCOON.PY

Class Kasino :

mapa= []

dinero = 0

día = 0

maquinas = dicc

decoración =dicc

fluidez = 3

* Estas variables de clase las usaremos para guardar y cargar partida.

def \_\_Init\_\_()

self.beneficios\_maquinas=dicc

def crear\_mapa ()

Nos crea el mapa que usaremos durante el juego, según ancho y largo dados como parámetro. Por defecto

40x40

def fama()

Este método nos dará mediante una formula la variable fama , que dependerá de la decoración del casino e influirá en las visitas.

fama = (10\*paredes)+(10\*suelo)+(10\*reforma) + modificador\_fama

* El modificador esta explicado en el programa.

def game()

Este método lo usaremos para printear el mapa, pondremos un título y luego mediante -join- printearemos la lista-mapa

def siguiente()

UNDER CONSTRUCTION

def compra()

Este método se ejecutara al comprar cualquier articulo y añadirá dicho artículo al dicc correspondiente para -saber- que lo tenemos y nos descontara el dinero si lo tenemos, en caso contrario nos avisara que no nos llega y volveremos

def guardar\_mapa()

Este método guarda el mapa kasino.mapa en un txt para poder recuperar los datos en otro momento y seguir asi la partida

def cargar\_mapa()

Este método utiliza lo escrito en el txt mediante guardar mapa() para recuperar kasino.mapa igual que estaba en el momento en que guardamos partida

def guardar\_otras()

Este método guardara en 1 documento los dicc kasino maquina y kasino decoración y los ints día y dinero ,en esta ocasión se escribirán todos juntos ya que es fácil recuperarlos

def cargar\_otras()

Este método recuperara los dicc kasino máquinas y kasino decoración y los ints día y dinero.

Todos los métodos de títulos asci hasta el final de esta clase (kasino) podrían ser funciones independientes pero las pongo aquí para agruparlas

def título()

def proveedores()

def titulo\_maquinas()

def titulo\_constructor()

titulo\_decoracion()

def titulo\_estado()

def titulo\_informe()

def titulo\_configuracion()

* Todos los métodos anteriores son títulos asci para printear

Class objetos:

def \_\_init\_\_()

Aquí tenemos una lista de cada maquina para printearla en el mapa y otra lista por cada maquina con 2 ,elementos ,ganancias por dia y precio de compra de la propia maquina

def entrega\_lista \_maquina ():

Tiene un parámetro que es maquina, que representa la maquina que queremos printear, este método devuelve la lista para printear dicha maquina

DISPLAY\_TYCOON.PY

class display\_tycoon

Esta clase la utilizamos para los displays, todos los displays tienen una estructura similar donde tenemos el metodo que aporta la clase grafica y el metodo controlador. La parte grafica es lo que printeamos y el controlador gestiona todo , es decir printea lo grafico, detecta el movimiento y actúa correspondientemente segun la posicion en la que estemos y la tecla pulsada

.En resumen los métodos de esta clase forman grupos donde hay un método gestor que es el que realiza la tarea y para ello se apoya en los otros métodos complementarios del grupo, como podría ser el display , común en todos o metodos mas específicos designados para realizar una tarea en concreto

def \_\_init\_\_()

Tenemos las listas o listas de listas que se insertan en la parte grafica para moverse entre las casillas del menú, cada casilla seria un elemento de estas listas

Tambien tenemos otras variables necesarias para esta clase como posx posy y key que también nos ayudaran a desplazarnos .

def reset ()

Este método simplemente vuelve a dejar las variables que contiene como estaban al inicio del programa, por ejemplo las listas de listas necesitamos que sea asi para que al volver a un menú visitado antes estemos de nuevo en la primera casilla.

def key\_actions (lista,x):

Este metodo cogera la tecla pulsada y realizara distintas acciones dependiendo de donde la pulsemos, segun en que lista y en que parte,recibira 2 parametros , la lista por la que nos desplazaremos y x que representara el menu en el que estamos ya que pulsar algo en la casilla [0][0] cambia dependiendo de donde estemos.

La primera parte del método es común a todos ya que simula el desplazamiento de las direcciones pero cada apartado al cual accedemos según el parámetro x cambia las acciones ya que representa el menú en el que nos encontramos.

Para hacer este método universal a todas las listas usamos try y except, asi en vez de dar index error simplemente no hará nada , que es lo que queremos.

if x == 1 : # Utilizamos este para el display TUTORIAL

elif x == 2: # Utilizamos este para el display GAME OPTIONS

elif x == 3 : # Utilizamos este para el display OPCION COMPRA

elif x == 4: # Utilizamos este para el display MAQUINAS

elif x == 5 : # utilizaremos este display para el menu CONSTRUCTOR

elif x == 6 : # utilizaremos este display para el menu VER MAPA

elif x == 7 : # utilizaremos este numero para el display DECORACION

elif x == 8 : # utilizaremos este numero para el display CONFIRMA INSTALACION

elif x == 9 : # Utilizamos este para el display OPCION COMPRA

elif x == 10: # Utilizamos este para el display CONFIGURACION

Este es un método extenso

METODOS GAME OPTIONS 2 def game\_options () Y def display\_game\_options ()

""" Metodos en grupo, display\_game\_options() muestra la parte grafica e integra la lista para desplazarnos por el menu, game\_options() es el metodo principal y gestiona la funcion integrada key\_recorder() que captura la tecla pulsada y mediante key\_actions() realiza las acciones correspondientes, se devuelve el resultado sobre que opcion de juego escogemos """

def game\_options

def display\_game\_options ()

A PARTIR DE AQUÍ IGUAL QUE EL JUEGO SE DIVIDE EN 4 MENUS DIVIDIREMOS NUESTROS METODOS EN 4 GRUPOS, CADA GRUPO TRABAJARA EN CONJUNTO PARA CUMPLIR EL OBJETIVO SEGÚN SU PROPOSITO.

GRUPOS: COMPRAR , ESTADO , CONFIGURACION , SIGUIENTE

La mayoría de las veces un método nos

LOS SIGUIENTES METODOS REPRESENTAN LA OPCION DE JUEGO COMPRAR

COMPRAR

MAQUINAS DECORACION

Colocar maquinas

En esta ocasión como al comprar una maquina hay que instalarla en 1 mapa y tener en cuenta medidas y otras cosas, nos apoyamos en el grupo colocar maquinas que mediante una serie de métodos comprobaran que el objeto cabe, nos pedirán confirmación y lo situaran en el mapa

Esta es la estructura que sigue el menú de compra, dentro de comprar se ejecutara las otras , en la mayoría de veces sin salir al main, es decir, métodos opción compra mediante un display nos da la opción de maquinas, decoración y salir. Si cogemos maquinas o decoración automáticamente nos llevara a ese método sin salir al main si no lo veo necesario.Asi ya que la mayoría de opciones son simples, se resuelven desde aquí.

METODOS OPCION COMPRA

""" Metodos en grupo, display\_maquinas() muestra la parte grafica e integra la lista para desplazarnos por el menu, maquinas() es el metodo principal y gestiona

la funcion integrada key\_recorder() que captura la tecla pulsada y mediante key\_actions() realiza las acciones correspondientes, En este caso la respuesta se gestiona desde

aqui,en maquinas() si la respuesta es comprar maquina se gestionaran una serie de metodos para colocar el objeto. Seran explicados debajo de cada uno.

def display\_maquinas ()

def maquinas ()

METODOS COMPRAR MAQUINAS

""" Metodos en grupo, display\_maquinas() muestra la parte grafica e integra la lista para desplazarnos por el menu, maquinas() es el metodo principal y gestiona la funcion integrada key\_recorder() que captura la tecla pulsada y mediante key\_actions() realiza las acciones correspondientes, En este caso la respuesta se gestiona desde aqui,en maquinas() si la respuesta es comprar maquina se gestionaran una serie de metodos para colocar el objeto.

def display\_maquinas ()

def maquinas ()

METODOS COLOCAR MAQUINAS

""" Estos metodos son invocados desde maquinas () en METODOS COMPRA MAQUINAS (arriba) , Es una agrupacion de metodos y coloca\_cosas es el gestor de este grupo ,utilizara display\_coloca\_cosas(), medir\_espacio(), clona\_mapa() e instalar() para colocar la maquina elegida anteriormente en maquinas(),arriba, en el mapa del casino, tiene key\_recorder() integrado y utiliza esta y key\_action() para desplazarnos por el mapa

medir\_espacio() mirara si la maquina elegida cabe en la posicion seleccionada, y dependiendo de eso nos dara permiso ono para instalar

clona\_mapa() es un pequeño metodo para copiar lista de listas

instalar() coloca la maquina en el mapa

display\_coloca\_cosas Nos muestra el mapa y la esquina superior izquierda del objeto seleccionado , pudiendo desplazarla para seleccionar la posicion de instalacion """

def confirma\_compra (mapa) Nos muestra un mapa clonado con el articulo instalado, si aceptamos se instalara en nuestro mapa y lo tendremos, sino se descarta el mapa clonado y volvemos

def display\_coloca\_cosas ()

def medir\_espacio (casino,objeto,posx,posy):

def clona\_mapa (mapa)

def instalar (casino,objeto,posx,posy)

def coloca\_cosas ()

def display\_confirma\_compra()

def confirma\_compra (mapa)

METODOS DECORATION

""" Metodos en grupo, display\_decoration() muestra la parte grafica e integra la lista para desplazarnos por el menu, decoration() es el metodo principal y gestiona la funcion integrada key\_recorder() que captura la tecla pulsada y mediante key\_actions() realiza las acciones correspondientes, se encarga de mostrar todas las opciones de decoracion y en caso de comprar añadirlas al dicc correspondiente para que influyan en la -fama- del casino """

def display\_decoracion ()

def decoration ()

LOS SIGUIENTES METODOS REPRESENTAN LA OPCION DE JUEGO ESTADO

ESTADO

MAPA INFORME

Esta es la estructura que sigue la parte de juego que corresponde al menú estado,al pulsar estado nos dará a elegir entre mapa e informe, mapa nos llevara a los métodos ver mapa que únicamente nos muestran el mapa y nos dan la opción de salir, y al pulsar informe lo mismo pero nos muestra datos sobre maquinas, decoración, visitas, ganancias…

METODOS ESTADO CASINO

Metodos en grupo, display\_opcion\_estado() muestra la parte grafica e integra la lista para desplazarnos por el menu, opciones\_estado() es el metodo principal y gestiona la funcion integrada key\_recorder() que captura la tecla pulsada y mediante key\_actions() realiza las acciones correspondientes,depende de lo que se elija se invocara desde aqui metodos mapa o metodos informe, los dos grupos siguientes a este

def display\_opcion\_estado ()

def opciones\_estado ()

METODOS VER MAPA

Metodos en grupo, display\_ver\_mapa() muestra el mapa en su estado actual, ver\_mapa() es el metodo principal y gestiona la funcion integrada key\_recorder() que captura la tecla pulsada y mediante key\_actions() realiza las acciones correspondientes,En este caso solo hay una opcion, volver,nos muestra el mapa hasta que decidamos volver

def display\_ver\_mapa ()

def ver\_mapa ()

METODOS VER INFORME

Metodos en grupo, display\_estado() muestra un informe actual delcasino, con ladecoracion y maquinas adquiridas y su correspondiente fama, visitas y ganancia, estado() es el metodo principal y gestiona la funcion integrada key\_recorder() que captura la tecla pulsada y mediante key\_actions() realiza las acciones correspondientes,En este caso solo hay una opcion, volver,nos muestra el informe hasta que decidamos volver

def display\_estado ()

def estado ()

METODOS CONFIGURATION

Metodos en grupo, display\_configuracion muestra la parte grafica e integra la lista para desplazarnos por el menu , configuration() es el metodo principal y gestiona la funcion integrada key\_recorder() que captura la tecla pulsada y mediante key\_actions() realiza las acciones correspondientes,depende de lo que se elija se obtendra un resultado que en esta ocasion se enviara a main para que este realice acciones en consecuencia

def display\_opcion\_estado ()

def configuration ()