Universidad Americana
Profesor: Adrián Badilla Ch.

Taller 02

"Those who can imagine anything, can create the impossible."
— Alan Turing

Instrucciones: Escriba el código en Python 3 que funcione como describe el enunciado. Entregue el examen mediante un link hacia el repositorio personal en git. Debe indicar las dependencias y d manera conciso un .txt explicando cómo instalarlas, la arquitectura de su proyecto y su funcionalidad-alcance; esto antes de la hora especificada en el campus.

El taller es de carácter absolutamente individual por lo que no se permite compartir código; tomarlo de otra fuente será considerado como trampa-plagio.

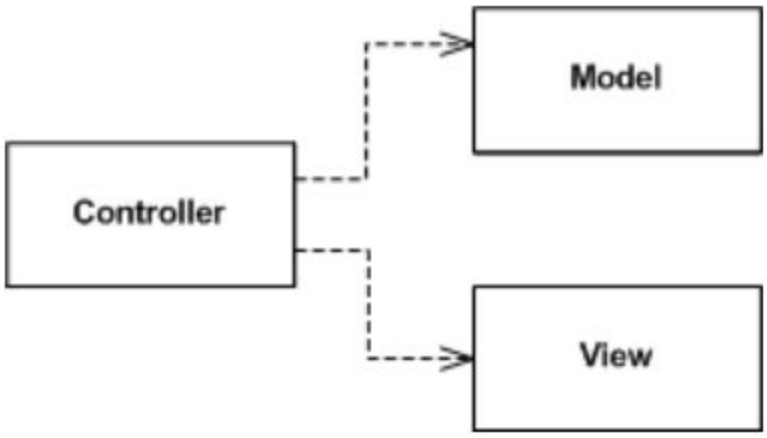
Enunciado:

Basados en la clase # 3, en el patrón de diseño estructural Model-View-Controller:
Crear su propio acercamiento personalizado a la situación de crear 3 modelos, donde uno deba ser un objeto compuesto** y uno un objeto sin dependencias. *

Ejemplo1:
-> Departamento (cod, nombre, lista_productos-> referencia a productos) **
-> Producto (descripcion, precio) **
-> Descuento (porcentaje, duracion(04/02/2010-02/06/2015) *

Ejemplo2:
-> Banda (genero, nombre, lista_musicos -> referencia musicos) **
-> Musico (instrumento, nombre) **
-> Festival (nombre, fecha) *

Crear sus clases propias definiciones de Model, View y Controller. El objetivo es familiarizar al estudiante con el paradigma MVC, donde los modelos son únicamente la definición del objeto de la vida real, la vista despliega únicamente información al usuario y el controlador es la interacción de las partes.



Evaluación

1. (35%) Solicitud de los datos al usuario
   a) (10%) Se solicita información de moverse en el programa
   b) (10%) Se solicita información del modelo independiente
   c) (15%) Se solicita información de los modelos compuestos
2. (10%) Se implementa métodos en el controlador para agregar y modificar elementos
3. (35%) Se generan vistas para ver tanto el modelo independiente como los modelos compuestos, se capturan las excepciones del usuario
   a) (15%) Se puede ver cada modelo por separado.
   b) (10%) Se puede ver un objeto en específico de cada modelo.
   c) (10%) Se captura en la creación de objetos los errores y se maneja el error.
4. (20%) La estructura del programa
   a) (10%) El programa puede correrse hacia toda dirección sin problemas de bucles.
   b) (10%) Se limpia la consola y se restablecen los valores iniciales si desea volver a comenzar desde 0. También debe incluir la opción de salir en todo momento.

Nota: En cada uno de los anteriores rubros se evaluarán las buenas prácticas de programación.