Python Módulo I – Examen final

Parte práctica

Se tomará en cuenta las **buenas prácticas**, **las tabulaciones o sangrados respectivos** y así como el uso de **flake8** para el adecuado diseño de código de nuestros problemas

1. Escriba un programa donde tendrá los siguientes requisitos (4 ptos):

Reglas:

- Crear una clase llamada **Empleado** donde sus atributos deben ser nombre, edad, sueldo y de nacionalidad **peruana**, tendrá un método para solicitar su nombre y otro para solicitar su edad. Así como un método cumpleaños donde cada vez que invoque aumentará en un año la edad de la persona.

- Crear la instancia de la clase Empleado y usar el nuevo método **aumentoSueldo** que incrementar su sueldo en un 30% (mínimo instanciar la clase 2 veces, mostrar por pantalla dicho sueldo ya incrementado).

- Crear un siguiente método **prediccion()** que retorne un mensaje donde indique que: "En el año XXXX tendrá XX años", el año se ingresará por parámetro y la edad también, realizar una validación si la edad ingresada por parámetro es menor a la del constructor indicar que no es posible realizar la operación (Mostrar por pantalla este valor)

Aplicando la definición de Herencias. Crear una clase llamada Persona (Que heredará de la anterior Clase) y agregar un atributo sueldo a esta clase

- Crear un método **transferencia** y mostrar saldo (mostrará el saldo actual que tiene la persona) para la clase mencionada

- El método transferencia hace que la clase Empleado que llame al método pueda transferir la cantidad monto al objeto Empleado2 por consiguiente deberá ir actualizando también el saldo o monto que tiene el otro empleado en su cuenta cada vez que use el método transferencia

- Comprobar si no se tiene dinero suficiente no se ejecuta la acción e imprimir "Saldo insuficiente". Comprobar instanciado la clase realizando una transferencia y con dos personas.

2. Utilizar el concepto de **módulo** necesariamente. Y Escribir un programa para el manejo de listas el cuál cumplirá los siguientes requerimientos (**2 ptos**):

Reglas:

- Crear una función que le permitirá almacenar X números aleatorios en una lista y finalmente los imprimirá por consola al llamar la función. X solo puede ser entero. No otro tipo de dato. Y también un índice existente en la lista.
- Crear una función que le permita almacenar los números **no** repetidos de la lista anterior, retornar este valor e imprimirlo por consola.
- Crear una función donde se creará una lista para ordenar de mayor a menor la lista que se creó en el ítem anterior (número no repetidos) y otra lista para ordenarlas de menor a mayor, retornar este valor e imprimirlos por consola.
- Crear una función para indicar cuál es el mayor número par de la lista (lista del ítem 2), retornar este valor e imprimirlo por consola.
- Crear el archivo principal.py, donde solo llamarás las anteriores funciones que se encontrarán alojadas en un módulo

3. (**2 ptos**) Crear un decorador conteo.

Reglas:

- El decorador retornará la cantidad de parámetros que hayas usado en la función y que a su vez evaluará que deba ser mayor que 1 para procesar esta lógica, caso contrario indicarlo con un mensaje respectivamente.
- Al final de la función decorada indicará mediante un mensaje que la función fue ejecutada.
- La función que vas a crear va a capturar, la edad, nombre de un alumnos y la hora y el minuto en que fue registrado (usar la librería correspondiente de tiempo)
  Mostrando un mensaje siguiente: "Pedro de 30 años ha sido registrado a las 16 horas con 20 minutos"
- La función que será decorada también estará pasando 4 notas que calculará la media del estudiante.

4. (**2 ptos**) Crear un programa usando decoradores para mostrar solo la hora y el minuto del momento que se usa el decorador

Reglas:

- Al ejecutar el decorador mostrará un mensaje: "El decorador está siendo ejecutado a las H con minutos"

- Crear la función decorador adecuadamente que sumará los elementos de la función que pasará como parámetro de la función decoradora

- Crear una función, **por ejemplo:** usando 6 números e indicar el mayor de todos ellos (o x números) para decorarla con la función anterior.

- Usar la propiedad de N parámetros para la función a decorar usando sus **key y values** (**) y visualizar los resultados usando el decorador implementado con un mínimo tres ejemplos.

5. Agregar a tu **repositorio de exámenes** tu carpeta con el nombre de "examen final" con tus soluciones y realizar un **push** con todas tus soluciones de los anteriores problemas. (**2 ptos.)**

Indicaciones:

- Tiempo total del examen: 110 minutos

- Enviar el link del repositorio que has creado en github respectivamente y en cuál estarán todas sus soluciones al siguiente correo:

  docente.cerseu.unmsm@gmail.com

- Asunto: Parte práctica – Examen final