Universidad Nacional Mayor de San Marcos
Universidad del Perú, Decana de América

FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
PROGRAMA: ESPECIALIZACION EN PYTHON

MÓDULO I

Reforzamiento N°04

Importante: Subir todas sus soluciones a su repositorio donde se encuentra las soluciones de los anteriores reforzamientos.

1. Crear una función para encontrar el error en el siguiente bloque de código. Crea una excepción para evitar que tu programa se bloquee y además imprime un mensaje al usuario la causa y/o solución:

   suma = 80 + "Hola Pythonista"

2. Crear una función y dentro la respectiva excepción para el siguiente bloque de código para que tu programa no se bloquee y además imprime un mensaje al usuario la causa y/o solución:

   lista = [2, 6, 10, 4, 5, 23, 1]

   lista[10]

3. Crear una función y dentro la respectiva excepción para el siguiente bloque de código para que tu programa no se bloquee y además imprime un mensaje al usuario la causa y/o solución:

   persona= { **'nombre'**:'Xavier', **'apellido'**:'Rodriguez', '**dni**':'63325345'}
   persona['profesion']

4. Crear una función y dentro la respectiva excepción para el siguiente bloque de código para que tu programa no se bloquee y además imprime un mensaje al usuario la causa y/o solución:

   string = "Hello Pythonista"
   print(string/0)

Módulos

5. Crear un módulo que validará nombres de usuarios. Dicho módulo, deberá tener una función y cumplir con los siguientes requisitos:

   - Crear un archivo principal (main.py) donde importará al módulo creado.
   - El nombre de usuario debe tener una longitud mínima de 7 caracteres y un máximo de 12.
   - El nombre de usuario debe ser alfanumérico (usar **isalnum()**)
   - Nombre de usuario con menos de 7 caracteres, retorna el mensaje "El nombre de usuario debe contener al menos 7 caracteres".
   - Nombre de usuario con más de 12 caracteres, retorna el mensaje "El nombre de usuario no puede contener más de 12 caracteres".
   - Nombre de usuario con caracteres distintos a los alfanuméricos, retorna el mensaje "El nombre de usuario puede contener solo letras y números".
   - Si el nombre de usuario válido, la función retornará True.

6. Creando un archivo principal donde llamará a un módulo (**operaciones.py**) el cuál contendrá las siguientes funciones.
   - La primera función cargará una un número entero que pedirá al usuario que ingrese por consola un valor.
   - La segunda función solamente sumará dos valores.
   - Desde el archivo principal importar al módulo y sumar dos valores usando las funciones anteriormente creadas en el módulo

7. Creando un archivo principal (main.py) donde importará a un módulo (operaciones.py) el cuál contendrá las siguientes funciones:
   - Una función que genere 30 números enteros aleatorios entre 0 y 100, que muestre en pantalla esta lista.
   - Otra función que ordene los valores de una lista y volver a mostrarla.

8. Creando un archivo principal (main.py) donde importará a un módulo (operaciones.py) el cuál contendrá las siguientes funciones:

- Una función que realizará la carga de un valor entero.
- Una función que mostrará por pantalla la raíz cuadrada de dicho valor
- Y otra función el valor elevado al cuadrado y al cubo de dicho número.
- Utilizar el módulo math de python.

Archivos

9. Crear una función que pida al usuario un número entero entre 1 y 20 y guarde en un fichero (que no existe) con el nombre tabla.txt la tabla de multiplicar de ese número, done n es el número introducido.

10. Crear una función donde se permitirá guardar el nombre, apellido y edad de un usuario en un fichero (agenda.txt), cada usuario tiene que estar guardado en una línea diferente y cada dato de una persona tiene que estar separados por comas.

11. Crear una función que creará el archivo calificaciones.txt que contiene las calificaciones de un curso.

    Escribir un programa que contenga las siguientes funciones:

    - Una función que guarde el nombre, 2 notas y el promedio (el promedio lo calculará la función antes de escribirlo en el fichero)
    - Y una función que leerá el fichero anterior y dirá si el alumno X, aprobó o no, tener en cuenta que si tiene un promedio mayor a día tendrá un mensaje de "Alumno X, aprobado" de lo contrario "Alumno X, desaprobado"

Decoradores

12. Crear una función decoradora que dará los buenos días antes de ejecutar una función a ser decorada y luego de ser ejecutada lanzará un mensaje diciendo "Hasta luego".

    La función a decorar pedirá el nombre de una persona y retornará el mensaje "Soy 'nombre'".

13. Haciendo el uso de *args y **kwargs aplicarlo debidamente para usar decorar una función que recibirá 4 argumentos los cuales se multiplicaran entre ellos.
    Mensaje previo al usar el decorador "Está por ejecutarse la función" y mensaje luego de usar el decorador "La función ha sido ejecutado correctamente"

14. Crear un decorador donde imprimirá la cantidad de argumentos que tiene la función a decorar usando *args y **kwargs.
    Mensaje previo "La cantidad de argumentos que tiene la función es" mensaje post "La función decoradora terminó de ejecutarse correctamente"

    Ejemplo: Al usar una función suma que creamos y usamos

    suma(4, 1, 10, 2, 50, 64)

    El decorador determinará la cantidad de argumentos que tiene la función al decorarla.

    Salida:

    "La cantidad de argumentos que tiene la función es"

    5

    "La función decoradora terminó de ejecutarse correctamente"

Consumo de JSON:

15. Queremos consumir un JSON que se encuentra alojado en la nube el cual nos traerá los datos de una persona como la edad, nombre, email, dirección o nombre de la compañía en donde trabaja.

    La url a consumir usando Python es la siguiente:

    https://jsonplaceholder.typicode.com/users

Obtener respectivamente los valores necesarios para formar la siguiente oración:

Bienvenido "**nombre**" "**apellido**", usted tiene "**edad**" años. El correo que nos proporcionó es "**correo**" y la compañía actual donde trabaja se llama "**nombreCompañía**". Dentro de sus datos también encontramos una website que es: "**nombreWeb**"

Finalmente agregar un usuario al json obtenido pero sólo con los datos de nombre, apellido, edad y compañía donde trabaja y finalmente mostrarlo en consola (sólo ese usuario nuevo)

Indicaciones:

- Cada solución ser realiza en un diferente archivo Python *.py
- Correo a enviar soluciones y link de repositorio: cerseuufisi@gmail.com
- Asunto: Reforzamiento 04 – Módulo I
- Fecha máxima de entrega: Sábado 24 de febrero hasta las 23:59 horas.