Facultad de Ciencias Sociales y Administrativas

Licenciatura en Informática y Desarrollo de Software

PRÁCTICO DE ENSEÑANZA
ASIGNATURA: Ingeniería Del Conocimiento **PRÁCTICO Nº:** 1 **FECHA:** 13/08/24 **"Introducción Python"** **DOCENTES RESPONSABLES:** Prof. Giordani Osvaldo, Prof. Gonzalez Juan
NOMBRE Y APELLIDO DEL ALUMNO: **CURSO Y COMISIÓN:**
OBJETIVO: • Comprender los aspectos básicos del lenguaje de programación Python

A. **Consignas**

1. Lee la siguiente introducción sobre Python:

   - ¿Qué es Python? - Explicación del lenguaje Python - AWS

2. Realiza los siguientes ejercicios, tenga en cuenta de una vez realizados, realizar un repositorio en GitHub:

   2.1. Crea un código que imprima en pantalla la expresión "Mi Primer Código En Python."

   2.2. Crea un código que imprima en pantalla la siguiente expresión.

```
1 | A  B  C
2 | D  E  F
3 | G  H  I
```

   2.3. Crea un código que le permita ingresar una respuesta al usuario, haciéndole la siguiente pregunta:

   ¿Qué estás estudiando?

   El código debe poder imprimir en pantalla lo ingresado por el usuario (utilizando print).

   2.4. Crea un código que le permita ingresar una respuesta al usuario, haciéndole la siguiente pregunta:

   ¿En qué país vives?

   El código debe poder imprimir en pantalla lo ingresado por el usuario (utilizando print).

   2.5. Declara dos variables, llamadas nombre y edad.

   Asigna a la variable nombre el valor "David Bowman", y a la edad, el valor 51.

   2.6. Crea tres variables:

   - nombre
   - apellido
   - nombrecompleto

A nombre, asígnale el valor "Julia", y en apellido, asigna el valor "Roberts". Finalmente, construye la variable nombrecompleto concatenando las variables (recuerda sumar un espacio intermedio).

2.7. Declara la variable materia, asígnale el valor "Ingeniería del conocimiento", y muestra en pantalla la frase:

Estás estudiando “materia”

Para ello deberás concatenar la primera parte de la frase con el valor que asumirá la variable. Recuerda agregar un espacio antes de concatenar la variable al resto del texto.

2.8. Convierte el valor de num1 (num1=35) en un int e imprime el tipo de dato que resulta

2.9. Necesitamos imprimir el nombre y número de asociado dentro de la siguiente frase:

“Estimado/a (nombre_asociado), su número de asociado es: (numero_asociado)”

2.10. Muestra en pantalla el cociente (división al piso) de los siguientes dos números: 874 dividido entre 27.

Debes mostrar sólo el valor numérico que resulta de esta operación.

2.11. Redondea el número 10.676767 al entero más próximo, y muestra en pantalla el resultado.

2.12. Gestión de inventario con tuplas:

Consigna: Una tienda tiene un inventario de productos, cada producto tiene un nombre, precio y cantidad disponible. Representa cada producto como una tupla (nombre, precio, cantidad). Escribe una función que reciba una lista de productos (tuplas) y devuelva el producto más caro.

productos = [("laptop", 1200, 5), ("mouse", 25, 50), ("teclado", 100, 30)]

2.13. **Registro de estudiantes con diccionarios:**

Consigna: Una escuela lleva un registro de estudiantes donde la clave es el número de matrícula y el valor es un diccionario con nombre, edad y calificaciones en distintas materias. Escribe una función que reciba el registro de estudiantes y devuelva el promedio de calificaciones de un estudiante dado su número de matrícula.

```
estudiantes = {
    101: {"nombre": "Ana", "edad": 16, "calificaciones": {"matemáticas": 85, "ciencias": 90}},
    102: {"nombre": "Luis", "edad": 17, "calificaciones": {"matemáticas": 78, "ciencias": 88}}
}
```

2.14. Análisis de datos meteorológicos con arrays:

Consigna: Un meteorólogo registra las temperaturas diarias durante un mes y las almacena en un array. Escribe una función que reciba este array y devuelva la temperatura media del mes, la máxima y la mínima.

temperaturas = [22.5, 23.0, 21.0, 19.5, 25.0, 26.5, 24.0]

2.15. Manejo de parámetros variables con *args:

Consigna: Escribe una función que reciba un número variable de notas de estudiantes y devuelva la nota promedio. Utiliza *args para recibir las notas.

calcular_promedio(85, 90, 78, 92)

2.16. Creación de un perfil de usuario con **kwargs:

Consigna: Escribe una función que reciba datos de un usuario como nombre, edad, correo electrónico, y cualquier otro dato adicional usando **kwargs. La función debe devolver un diccionario con toda la información del usuario.

crear_perfil(nombre="Luis", edad=25, email="juan@mail.com", ciudad="Mendoza")

2.17. Administración de empleados con tuplas y diccionarios:

Consigna: Una empresa quiere administrar la información de sus empleados, donde cada empleado se representa como una tupla (nombre, edad, salario). Escribe una función que reciba un diccionario donde la clave es el ID del empleado y el valor es la tupla con su información. La función debe devolver un diccionario con los empleados que ganan más de un salario dado.

```
empleados = {
   1: ("Ana", 30, 3000),
   2: ("Luis", 25, 2500),
   3: ("María", 35, 4000)
}
```

2.18. Procesamiento de ventas con arrays:

Consigna: Una tienda quiere procesar sus ventas diarias almacenadas en un array. Escribe una función que reciba el array de ventas diarias y devuelva el total de ventas y el promedio de ventas por día.

ventas_diarias = [200, 450, 300, 400, 350, 500, 600]

2.19. Análisis de resultados deportivos con diccionarios:

Consigna: Un club deportivo registra los resultados de sus partidos en un diccionario donde la clave es el nombre del equipo rival y el valor es una tupla con los goles anotados y recibidos. Escribe una función que calcule el total de goles anotados y recibidos en la temporada.

```
resultados = {
   "Equipo A": (3, 2),
   "Equipo B": (1, 1),
   "Equipo C": (4, 0)
}
```

2.20. **Configuración de una aplicación con **kwargs:**

Consigna: Escribe una función que reciba configuraciones opcionales para una aplicación como modo oscuro, idioma, notificaciones, etc., usando **kwargs. La función debe devolver un diccionario con las configuraciones aplicadas.

configurar_app(modo_oscuro=True, idioma="es", notificaciones=False)

2.21. Ordenamiento de datos con tuplas:

Consigna: Escribe una función que reciba una lista de tuplas donde cada tupla contiene un nombre y una puntuación. La función debe devolver la lista ordenada por puntuación de mayor a menor.

puntuaciones = [("Ana", 85), ("Luis", 90), ("María", 78)]

2.22. Planificación de viajes con tuplas y diccionarios:

Consigna: Una agencia de viajes tiene diferentes paquetes turísticos, cada uno representado como una tupla (destino, precio, duración en días). Escribe una función que reciba una lista de estos paquetes y devuelva un diccionario con los destinos como claves y el precio total (precio por día * duración) como valor.

```
paquetes = [
   ("Paris", 200, 5),
   ("Roma", 150, 4),
   ("Londres", 180, 3)
]
```

2.23. Gestión de inventario con arrays:

Consigna: Una tienda maneja su inventario de productos en un array donde cada índice representa un producto específico y su valor es la cantidad disponible. Escribe una función que reciba el array de inventario y un número de productos vendidos (otro array) y devuelva el inventario actualizado.

```
inventario = [50, 30, 20, 10]
ventas = [5, 10, 5, 2]
```

2.24. Organización de eventos con *args:

Consigna: Escribe una función que reciba un número variable de nombres de eventos y los imprima en un formato de lista numerada. Utiliza *args para recibir los nombres de los eventos.

organizar_eventos("Concierto", "Exposición de arte", "Conferencia")

2.25. Análisis financiero con **kwargs:

Consigna: Escribe una función que reciba diferentes tipos de ingresos y gastos como **kwargs y calcule el balance final. La función debe manejar ingresos como positivos y gastos como negativos.

analizar_finanzas(sueldo=2000, renta=-800, transporte=-150, comida=-300, freelance=500)

2.26. Registro de empleados con tuplas y **kwargs:

Consigna: Escribe una función que reciba el nombre, edad, y salario de un empleado como parámetros obligatorios, y otros datos como dirección, número de teléfono, etc., como **kwargs. La función debe devolver un diccionario con toda la información del empleado.

registro_empleado("Ana", 30, 3000, direccion="Calle Falsa 123", telefono="123456789")

2.27. Estadísticas de ventas con arrays:

Consigna: Escribe una función que reciba un array con las ventas de cada mes y devuelva un diccionario con el total de ventas, el promedio mensual, y el mes con mayores ventas.

ventas_mensuales = [2000, 2500, 3000, 2800, 3500, 4000, 4200, 3800, 3600, 3900, 4100, 4500]

2.28. Organización de una biblioteca con diccionarios:

Consigna: Una biblioteca registra sus libros en un diccionario donde la clave es el título del libro y el valor es otro diccionario con la información del autor, año de publicación, y género. Escribe una función que reciba este diccionario y devuelva una lista de todos los libros publicados después del año 2000.

```
biblioteca = {
   "El señor de los anillos": {"autor": "J.R.R. Tolkien", "año": 1954, "género": "Fantasía"},
   "Cien años de soledad": {"autor": "Gabriel García Márquez", "año": 1967, "género": "Realismo mágico"},
   "El código Da Vinci": {"autor": "Dan Brown", "año": 2003, "género": "Suspenso"}
}
```

2.29. Registro de notas con tuplas y arrays:

Consigna: Escribe una función que reciba una lista de tuplas donde cada tupla contiene el nombre de un estudiante y sus calificaciones en un array. La función debe devolver un diccionario con el nombre del estudiante como clave y su promedio de calificaciones como valor.

```
notas_estudiantes = [
   ("Ana", [85, 90, 78]),
   ("Luis", [88, 92, 80]),
   ("María", [75, 85, 70])
]
```

2.30. Configuración de perfiles de usuario con **kwargs y arrays:

Consigna: Escribe una función que reciba una lista de usuarios y sus preferencias de configuración como `**kwargs`. La función debe devolver un diccionario donde la clave es el nombre del usuario y el valor es un array con las configuraciones aplicadas.

usuarios = ["Ana", "Luis", "María"]
configurar_perfiles(usuarios, idioma="es", modo_oscuro=True, notificaciones=False)

2.31. Análisis de rendimiento académico con diccionarios y arrays:

Consigna: Una universidad lleva un registro de las calificaciones de los estudiantes en diferentes materias. Cada estudiante tiene un ID único y su información se almacena en un diccionario donde la clave es el ID y el valor es otro diccionario con las materias y sus respectivas calificaciones (arrays). Escribe una función que reciba este diccionario y devuelva un ranking de estudiantes basado en su promedio general.

```
estudiantes = {
   101: {"matemáticas": [85, 90, 78], "ciencias": [88, 85, 80]},
   102: {"matemáticas": [92, 88, 84], "ciencias": [75, 80, 85]},
   103: {"matemáticas": [78, 85, 88], "ciencias": [90, 95, 92]}
}
```

2.32. Gestión de una red social con **kwargs y arrays:

Consigna: Escribe una función que administre publicaciones de una red social. La función debe recibir el nombre del usuario, el texto de la publicación y un número variable de etiquetas usando **kwargs y arrays. Además, debe manejar opciones adicionales como visibilidad pública o privada. La función debe devolver un diccionario con todos los detalles de la publicación.

publicar("Juan", "Mi primer post!", etiquetas=["#hola", "#primerPost"], visibilidad="publica", likes=100)

2.33. Simulación de ventas con tuplas, arrays, y *args:

Consigna: Una empresa quiere simular las ventas de diferentes productos. Escribe una función que reciba un número variable de ventas (tuplas) donde cada tupla contiene el producto, la cantidad vendida, y el precio por unidad. La función debe devolver el total de ingresos generados por las ventas.

simular_ventas(("Producto A", 10, 15.0), ("Producto B", 5, 25.0), ("Producto C", 3, 50.0))

2.34. Sistema de reservas con tuplas y diccionarios:

Consigna: Un hotel maneja sus reservas utilizando un diccionario donde la clave es la fecha y el valor es una lista de tuplas, cada tupla contiene el nombre del huésped, la habitación asignada y el precio. Escribe una función que permita hacer una nueva reserva verificando primero si la habitación está disponible en la fecha seleccionada.

```
reservas = {
   "2024-08-15": [("Juan", 101, 150), ("Ana", 102, 180)],
   "2024-08-16": [("Luis", 101, 150)]
}
```

2.35. Análisis de resultados de encuestas con diccionarios y arrays:

Consigna: Una empresa realiza encuestas de satisfacción y registra las respuestas en un diccionario donde la clave es la pregunta y el valor es un array con las respuestas recibidas. Escribe una función que calcule la frecuencia de cada respuesta para cada pregunta y devuelva un diccionario con estos resultados.

```
encuestas = {
   "¿Cómo califica el servicio?": [5, 4, 5, 3, 5, 4],
   "¿Recomendaría nuestro producto?": [1, 1, 0, 1, 1, 0]
}
```

2.36. Optimización de rutas con arrays y tuplas:

Consigna: Una empresa de logística necesita optimizar sus rutas de entrega. Cada ruta se representa como una tupla (origen, destino, distancia). Escribe una función que reciba una lista de rutas y un array con las distancias máximas permitidas para cada ruta. La función debe devolver las rutas que cumplen con las restricciones.

```
rutas = [("Madrid", "Barcelona", 620), ("Madrid", "Valencia", 350), ("Barcelona",
"Valencia", 350)]
distancias_max = [600, 400, 500]
```

2.37. Gestión de inventarios en múltiples tiendas con diccionarios y **kwargs:

Consigna: Escribe una función que gestione el inventario de una cadena de tiendas. La función debe recibir el nombre de la tienda, el producto y la cantidad a actualizar usando **kwargs. Debe manejar un diccionario donde la clave es el nombre de la tienda y el valor es otro diccionario con los productos y sus cantidades. La función debe actualizar el inventario y devolver el estado actual.

```
inventario = {
   "Tienda A": {"producto_1": 50, "producto_2": 30},
   "Tienda B": {"producto_1": 20, "producto_2": 40}
}
actualizar_inventario(tienda="Tienda A", producto_1=10, producto_2=-5)
```

2.38. Análisis de tendencias en redes sociales con arrays y tuplas:

Consigna: Una empresa de marketing digital desea analizar las tendencias de hashtags en las redes sociales. Escribe una función que reciba un array de hashtags y una lista de tuplas donde cada tupla contiene un hashtag y su frecuencia de uso. La función debe devolver los hashtags que han sido mencionados más de una cierta cantidad de veces.

```
hashtags = ["#verano", "#moda", "#viajes", "#verano", "#moda", "#tecnologia"]
tendencias = [("#verano", 120), ("#moda", 80), ("#tecnologia", 150)]
```

2.39. Administración de suscripciones con diccionarios, arrays, y **kwargs:

Consigna: Escribe una función que gestione las suscripciones a un servicio en línea. La función debe recibir el nombre del usuario, el tipo de suscripción (mensual, anual), y cualquier otra opción adicional usando **kwargs. La función debe actualizar un diccionario que almacene el historial de suscripciones de los usuarios y devolver el estado actualizado.

```
suscripciones = {
   "Jose": ["mensual", "anual"],
   "Ana": ["mensual"]
}
actualizar_suscripcion(usuario="Luis", suscripcion="mensual", auto_renovacion=True)
```

2.40. Simulación de mercado bursátil con arrays y tuplas:

Consigna: Escribe una función que simule el comportamiento de acciones en un mercado bursátil. La función debe recibir un array con los precios diarios de una acción y una lista de tuplas donde cada tupla contiene un día y un precio de compra o venta. La función debe devolver el beneficio o pérdida total si las acciones se hubieran comprado y vendido en los días especificados.

```
precios_diarios = [100, 105, 102, 110, 108]
operaciones = [("compra", 0), ("venta", 3), ("compra", 2), ("venta", 4)]
```