Python 2024 - Práctica 1

Objetivos

- Preparar el entorno de trabajo que utilizarán a lo largo de la materia instalando las herramientas básicas.
- Aprender a escribir nuestro primer programa con **Python** y ejecutarlo.
- Crear nuestro primer repositorio local de código y sincronizar el mismo a un repositorio remoto (GitHub).

Introducción

Python

Parte principal de esta actividad es tener su entorno de trabajo preparado para trabajar durante la cursada.
Por este motivo es fundamental que comenzemos por instalar `python` en su máquina.
Este año utilizaremos la versión de **Python 3.11.X**. Para instalar **Python** en su máquina les proveemos esta guía de instalación.

Git

Como ya vimos **Git** es una herramienta muy buena que nos permite manejar versiones de nuestro código de manera distribuida con nuestro equipo de trabajo.

Para poder realizar esto es necesario contar con un **servidor** de **Git** además de tener la herramienta instalada en su máquina **local**. En esta oportunidad vamos a analizar un poco GitHub que es el **servidor** de **Git** más popular actualmente.

Recuerde que se encuentra publicada la guía básica de Git donde se realizan varias de las tareas que solicitamos para realizar esta entrega.

Actividades introductorias

1. Desarrolla un programa que solicite al usuario que ingrese su edad y luego calcule y muestre cuántos años le faltan para alcanzar los 100 años.
2. Haz un programa que pida al usuario que ingrese una temperatura en grados Celsius y luego convierta esa temperatura a grados Fahrenheit, mostrando el resultado.
3. Crea un programa que calcule la suma de los primeros 100 números naturales utilizando un bucle for.
4. Cree un programa que dada una lista de números imprima sólo los que son pares. Nota: utilice la sentencia `continue` donde haga falta.

5. Implementa un programa que solicite al usuario que ingrese una lista de números. Luego, imprime la lista pero detén la impresión si encuentras un número negativo. Nota: utilice la sentencia `break` cuando haga falta.
6. Modifique el ejercicio 4 para que dada la lista de número genere dos nuevas listas, una con los número pares y otras con los que son impares. Imprima las listas al terminar de procesarlas.
7. Escribe un programa que tome una lista de números enteros como entrada del usuario. Luego, convierte cada número en la lista a string y únelos en una sola cadena, separados por guiones ('-'). Sin embargo, excluye cualquier número que sea múltiplo de 3 de la cadena final.

Entrega

1. Realizar la instalación de Python **3.11.X**.
2. Instale **git** en su sistema operativo.
3. Escriba en un archivo llamado `game.py` el siguiente programa en Python.

```
import random

# Lista de palabras posibles
words = ["python", "programación", "computadora", "código", "desarrollo",
"inteligencia"]

# Elegir una palabra al azar
secret_word = random.choice(words)
# Número máximo de intentos permitidos
max_attempts = 10
# Lista para almacenar las letras adivinadas
guessed_letters = []

print("¡Bienvenido al juego de adivinanzas!")
print("Estoy pensando en una palabra. ¿Puedes adivinar cuál es?")

word_displayed = "_" * len(secret_word)
# Mostrarla palabra parcialmente adivinada
print(f"Palabra: {word_displayed}")

for i in range(max_attempts):
    # Pedir al jugador que ingrese una letra
    letter = input("Ingresa una letra: ").lower()

    # Verificar si la letra ya ha sido adivinada
    if letter in guessed_letters:
        print("Ya has intentado con esa letra. Intenta con otra.")
        continue

    # Agregar la letra a la lista de letras adivinadas
```

```
    guessed_letters.append(letter)

    # Verificar si la letra está en la palabra secreta
    if letter in secret_word:
        print("¡Bien hecho! La letra está en la palabra.")
    else:
        print("Lo siento, la letra no está en la palabra.")

    # Mostrar la palabra parcialmente adivinada
    letters = []
    for letter in secret_word:
        if letter in guessed_letters:
            letters.append(letter)
        else:
            letters.append("_")

    word_displayed = "".join(letters)
    print(f"Palabra: {word_displayed}")
    # Verificar si se ha adivinado la palabra completa
    if word_displayed == secret_word:
        print(f"¡Felicidades! Has adivinado la palabra secreta:
{secret_word}")
        break
else:
    print(f"¡Oh no! Has agotado tus {max_attempts} intentos.")
    print(f"La palabra secreta era: {secret_word}")
```

4. Comience un repositorio **local** y agregue el archivo recientemente creado.
5. Crea tu propio repositorio **remoto** en Github y suba el archivo al repositorio remoto.
6. Agrega el `README.md` con tu nombre y número de estudiante.
7. Modifique el programa anterior con las siguientes funcionalidades:
    - El juego tiene un bug. Si no se inserta ninguna letra para adivinar, el valor de la letra es un string vacío "". Para este caso, el juego marca como que es un acierto. Modifica el mismo para que indique que es un error en este caso.
    - Modifique el juego para que en lugar de tener un número intentos se tenga un número de fallos. En este caso el usuario pierde cuando el número de fallos es alcanzado.
    - Agregue al juego niveles de dificultad. La variación de la dificultad sería:
        - **Fácil**: En la palabra a adivinar se muestran todas las vocales por defecto.
        - **Media**: Se muestra la primer y la última letra de la palabra.
        - **Difícil**: No se muestra ninguna letra de la palabra.

Nota: Por cada funcionalidad agregada se debe realizar al menos un commit que identifique el cambio.

Pautas

- **Puntos:** 10.
- **Fecha límite de entrega:** Viernes, 15 de marzo de 2024, 23:59
- **Modalidad de entrega:** Copie el enlace de su repositorio remoto con la resolución en la tarea de Cátedras.

Actividad Extra

Creación de un programa básico de gestión de inventario.

Desarrollar un programa en Python que permita gestionar un inventario simple de productos, incluyendo funciones básicas como agregar productos, eliminar productos y mostrar el inventario.

El programa debe tener un menú interactivo que permita al usuario seleccionar las siguientes operaciones:

- Agregar un nuevo producto al inventario, solicitando al usuario el nombre y la cantidad inicial del producto.
- Eliminar un producto existente del inventario, solicitando al usuario el nombre del producto a eliminar.
- Mostrar el inventario actual, que incluya el nombre y la cantidad de cada producto.
- Salir del programa.

El inventario puede ser almacenado utilizando un diccionario simple, donde las claves sean los nombres de los productos y los valores sean las cantidades.
Se deben manejar situaciones simples como la introducción de productos duplicados o la eliminación de productos inexistentes.