Posibles Preguntas examen teórico

- **Diccionarios**

Estructura de datos alterable, almacenan pares de **clave** (siempre texto, tipo str) - **valor** (pueden ser el tipo de dato que queramos).

Definir un diccionario →

Dict1 = {"clave": "valor", OJO!!! Si nuestro valor es numérico, no hace falta ponerlo entre comillas.

- **Listas**

Estructura de datos alterable de elementos.

- Definir una lista →

Lista = ["**elemento1**", **numero**, "elemento2].

- Anidamiento de listas → lista_anidada = [["elemento1"], ["elemento2"].
- Podemos acceder a cada elemento de la lista por el índice numérico;

print (Lista[2]). → se imprimirá por pantalla el **numero** de la lista.

- ¿Como comprobamos que un elemento existe en nuestra lista?

If "**elemento1**" is in Lista:
    Print ("Existe").

- **Len ()** → nº de elementos que tiene la lista.
- Formas de añadir elementos

append() → añade al final de la lista.
Insert () → añade en el lugar que queramos de la lista.

- Formas de eliminar elementos.

Remove () → eliminas el elemento que tu quieras.
Pop () y del → elimina algo especificado.
Celar () → vacía la lista

- Copiar listas.

Lista 2 = lista1. ///// copy () o list ().

- Unir listas. + o extend ()
- Ordenar listas: **sort ()** por orden alfabético y **reverse ()** para un orden inverso.

- **Tuplas**

(sus reglas básicas)

- try except

Definiciones básicas

- Programación → proceso de diseño y construcción de programas informáticos.
- Codificar → escribir códigos de un idioma a otro.
- Algoritmo → conjunto de instrucciones o reglas definidas y ordenadas que tienen un fin.
- **Variable** → espacio de memoria con información conocida o desconocida.

Declarar una variable ¿?

- Son alterables, cada variable recibe instrucciones de los OBJETOS. (Las variables son objetos).
- Case sensitive.
- Carácteres alfanuméricos y guiones bajos. NO PUEDEN COMENZAR POR UN NÚMERO .
- type () → obtenemos el tipo de dato.

- **Función** → proporción o bloque de código reutilizable y que realiza una tarea determinada.
  - bloque de código con un nombre asociado.
  - Reciben parámetros de entrada.

Definir funciones ¿?
- def (nombre de la función):
- SENTENCIAS: bloque de instrucciones que realizará la función.
- RETURN: sentencia de devolución de la función (devuelve la el valor de la función).
- No ARGUMENTOS = Nº PARÁMETROS (Si hay uno de menos = error).

Ventajas de usar funciones ¿?
- Modularización → segmenta una programa difícil en partes más fáciles.
- Reutilización → reutilizar una misma función en distintos programas.

- Objeto → unidad dentro de un programa con un estado (atributo) y un comportamiento (métodos). SE CREAN INSTANCIANDO CLASES.
- Clases → representación de una entidad o concepto.

Excepciones. → MIRAR EN TEORÍA.
Definicíón de IDE → Entorno de desarrollo intregrado seguro. Git hub →control de versiones.

ESTRUCTURAS DE CONTROL MUY IMPORTANTES !!!!!

- Modifican el flujo de ejecución de las instrucciones de un programa.
- 3 tipos; if, whiile y for.

CONDICIONALES.
If → cuando se cumple la condición. (True → se ejecuta la sentencia IF , False → no se ejecuta, **else**).
Elif → necesita un if previo. Es una condición adicional.
Identación → necesaria para que se ejecute correctamente Python. Delimita bloques de código.

BUCLES O CICLOS
Secuencias que se repiten n veces hasta que la condición asignada al bucle se deja de cumplir.
While, for y do-while.
While → se repite mientras la condición sea verdadera. (Si es false, se deja de ejecutar).
For → recorre secuencias de valores.
Break → se para el bucle antes de que finalicen los elementos iterables.
¿Que es un elemento iterable? → elemento que se puede recorrer uno por uno
Continue → se para la iteración del bucle y se continua con el siguiente.

EXCEPCIONES !!!
Ríase → lanzamos una excepción.
Ocurren durante la ejecución de un programa, interrumpe el flujo normal de ejecución.
Error = excepción → usamos **try except**
Controla las excepciones.
El código que genera una excepción se coloca dentro de try.
El código que se ejecuta en caso de que se produzca la excepción en except. Si no se encuentra genera un error.