Buenos Aires aprende
Agencia de Habilidades para el Futuro

Talento Tech

Iniciación con Python

Clase 12 - “Funciones y Diccionarios”

Buenos Aires Ciudad

¡Les damos la bienvenida!

 Vamos a comenzar a grabar la clase

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Buenos Aires Ciudad

Talento Tech

Clase 11.

Funciones

1. Funciones en Python: definición y uso.
2. Argumentos y retorno de valores.
3. Funciones que llaman a funciones.

Clase 12.

Ruta de avance

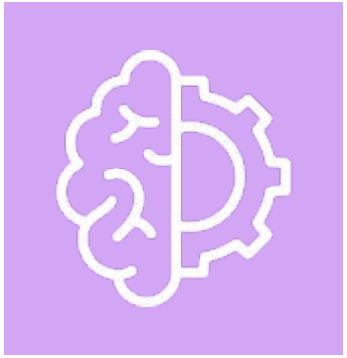
1. Creamos las funciones necesarias para el Proyecto Integrador

Clase 13.

Bases de Datos

1. Concepto y utilidad de los módulos en Python
2. Introducción a Bases de Datos.
3. Idea de tabla, campo, índice, clave, etc.

Pero antes...



¡Resolvamos los “**Ejercicios prácticos**” de la clase anterior!

Objetivos de la clase

En esta clase vamos a visualizar cómo podés darle vida a tu Proyecto Final Integrador. Crearemos un sistema básico para gestionar un inventario de productos, usando funciones y un diccionario, que será nuestra estructura principal de datos. A lo largo de la clase vamos a:

- Crear un conjunto de funciones para gestionar un inventario.
- Utilizar diccionarios para almacenar datos temporalmente.
- Prepararnos para la implementación del Proyecto Final Integrador (PFI).

¿Porqué usamos funciones?

Recordemos que una función es una forma de organizar y reutilizar nuestro código, son casi indispensables para nuestro proyecto. Entre otros motivos, porque modularizan y ordenan nuestro código, haciendo más fácil su escritura y mantenimiento posterior. ¡No pueden faltar en nuestro PFI!

Definición de una función:

```
# La función se define con def y comienza luego de los ":"
def saludar():
    print("Hola, ¡bienvenido!")
```

Diccionario para el inventario

Podemos emplear un diccionario como estructura de datos para el inventario de productos. Usamos una clave para cada dato que queremos almacenar. Por ejemplo:

Diccionario “inventario”:

```
inventario = {
    1: {"nombre": "Manzana", "descripcion": "Fruta fresca",
        "cantidad": 50, "precio": 0.5, "categoria": "Frutas"},
    2: {"nombre": "Pan", "descripcion": "Pan casero",
        "cantidad": 20, "precio": 1.0, "categoria": "Panadería"}
}
```

Diccionario para el inventario

Detalle de la posible estructura del diccionario:

- **nombre**: Es el nombre del producto. Es una cadena de caracteres que lo describe de manera general.
- **descripcion**: Descripción más detallada del producto. Es también un texto.
- **cantidad**: Cantidad disponible del producto en el inventario. Es un número entero.
- **precio**: Representa el precio del producto. Es un número decimal (tipo float).
- **categoria**: Indica la categoría a la que pertenece el producto. Esto nos permite organizar los productos en grupos según su tipo.

Registro de productos

Necesitamos una función **registrar_producto()** que se encargue de agregar los productos en el diccionario **inventario** con un código único y sus respectivos datos. La función pedirá que se ingrese los detalles del producto y los almacenará en el diccionario.

Usaremos una variable que actúe como un contador para los códigos de los productos, así cada vez que se registra un producto, se le asigna automáticamente un código nuevo.

El diccionario inventario usará el código del producto como clave, mientras que los valores asociados a esa clave serán otro diccionario que contendrá toda la información del producto.

Listado de productos

Nuestro proyecto necesita una función que se encargue de mostrar una lista todos los productos que están almacenados en el diccionario inventario. La hemos llamado **mostrar_productos().**

El código de esta función debe recorrer todo el inventario y mostrar la información de cada producto de manera clara, incluyendo su código, nombre, descripción, cantidad, precio y categoría. Para ello puedes usar un bucle.

Ten en cuenta que si el inventario está vacío, la función debería informar que aún no han ingresado productos

Actualización de productos

En aplicaciones como la que estamos desarrollando, es necesario contar con una opción que permita actualizar los datos almacenados. Para ello, escribiremos la función **actualizar_producto().**

Esta función debería solicitar que se ingrese el código del producto a actualizar, y verificar si existe en nuestro diccionario. En caso afirmativo se piden el o los datos a actualizar y se efectúa el reemplazo de los valores en el diccionario. Si el producto cuyo código hemos ingresado no existe, se puede mostrar un mensaje explicando la situación antes de salir de la función.

Por supuesto, ¡puedes agregar todas las funcionalidades extra que consideres necesario!

Eliminación de productos

En algún momento vas a necesitar quitar elementos de tu lista de productos. Productos obsoletos, o que no desees comercializar más, deberían ser quitados del diccionario para que no ocupen lugar o demoren innecesariamente las búsquedas.

eliminar_producto() debe pedir el código del producto a borrar, buscarlo en el diccionario, y si lo encuentra, quitarlo de él. Si no lo encuentra, podemos notificar a la usuaria o usuario de esta situación.

TIP: Es posible que puedas utilizar en esta función un algoritmo similar el que usaste en **actualizar_producto()**

Búsqueda de productos

Frecuentemente necesitamos conocer datos de un único producto. Si bien la función **mostrar_productos()** que escribiste antes hace esto, lo cierto es que si tenemos muchos productos el listado puede ser demasiado extenso.

Podés crear una función más especializada (a la que llamamos **buscar_producto()**) que le pida a la persona que opera el programa ingresar el código del producto que está buscando, y si el producto existe en el inventario, mostrar la información de ese único producto, con un formato claro.

TIP: Si el código que se ingresa no está registrado, podemos avisar que no se encontró el producto.

Reporte de stock bajo

En muchos proyectos nos interesa conocer qué productos de nuestro inventario poseen pocas unidades. Esta información nos facilita organizar las compras y reposición de stock.

Para eso, podemos crear **reporte_bajo_stock()**, una función que se encargue de indicar que se ingrese un límite de stock, y luego busque en el diccionario todos los productos cuya cantidad sea igual o inferior a ese límite. Finalmente, debería mostrar todos esos productos en pantalla.

TIP: Validá la entrada del usuario o usuaria, para evitar que se ingresen valores negativos o que no sean coherentes con la lógica de tu programa.

Menú principal

En clases anteriores hemos analizado la utilidad que tiene dotar a nuestra aplicación de un menú que permita a la persona que lo utiliza acceder a las funciones que hemos desarrollado.

Por ejemplo, nuestro menú principal podría mostrar las distintas acciones disponibles (registrar productos, mostrar el inventario, actualizar productos, eliminarlos, buscarlos y generar reportes de bajo stock). Se seleccionará la acción escribiendo el número de la opción que se desea accionar y el programa entonces ejecutaría la función correspondiente.

TIP: ¡ El menú principal también puede ser una función!

¡Vamos a la práctica! 

Ejercicios prácticos

Optativos | No entregables

Registro de productos con validaciones

Escribí una **función** que permita registrar un nuevo producto en el inventario, pero con una condición: la cantidad de productos debe ser mayor que 0 y el precio también debe ser un valor positivo.

Al ingresar una cantidad o precio no válido, debe mostrar un mensaje de error y pedir los datos nuevamente hasta que sean correctos.

Ejercicios prácticos

Optativos | No entregables

Visualización personalizada de productos

Agregá una función al sistema explicado en la clase, que permita simular la venta de un producto.
El usuario o usuaria deberá ingresar el código del producto y la cantidad a vender. Si la cantidad en stock es suficiente, la función debe restar esa cantidad del inventario. Si la cantidad solicitada es mayor a la disponible, debe mostrar un mensaje de error.

¡NUEVO CUESTIONARIO EN CAMPUS!

La resolución del cuestionario es de carácter obligatorio para poder avanzar en la cursada.

Buenos Aires aprende
Agencia de Habilidades para el Futuro

BA Buenos Aires Ciudad

Talento Tech