PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL

FORMATO GUÍA DE APRENDIZAJE

1. IDENTIFICACIÓN DE LA GUIA DE APRENDIZAJE (GFPI-F03-AP03-G22)-1 de 2

- **Denominación del Programa de Formación**: Análisis y Desarrollo de Software
- **Código del Programa de Formación**: 228118
- **Nombre del Proyecto Formativo**: Desarrollo de Software para Integrar Tecnologías Orientadas a Servicios
- **Fase del Proyecto**: Ejecución
- **Actividad de Proyecto Formativo**: Construir los componentes de la solución informática de acuerdo a las características funcionales y de calidad del diseño
- **Competencia:** Desarrollar la solución de software de acuerdo con el diseño y metodologías de desarrollo.
- **Resultados de Aprendizaje**: codificar el software de acuerdo con el diseño establecido
- **Duración de la Guía de Aprendizaje** (horas): 135 (110 directas +25 indirectas)

2. PRESENTACIÓN


https://acortar.link/gqyJ

Hoy, te invitamos a embarcarte en un emocionante viaje hacia el fascinante mundo de Python. Este no es solo un conjunto de actividades, sino una oportunidad para transformar tu relación con la programación y alcanzar nuevas alturas. En cada desafío, descubrirás un espacio donde tus habilidades y conocimientos florecerán, llevándote a la excelencia en el desarrollo de sistemas de información.

Renueva tu compromiso contigo mismo. Estás a punto de emprender un camino que no solo mejorará tu destreza técnica, sino que te convertirá en un desarrollador y analista deseado por las empresas. No subestimes el poder de este viaje; es tu oportunidad de modelar, asegurar y aplicar calidad de una manera que te distinga en el mundo profesional.

Asume el papel principal en tu aprendizaje. Planifica, diseña y ejecuta tu proyecto formativo con la certeza de que cada esfuerzo te acerca a ser el mejor en lo que haces. No estamos aquí solo para enseñarte, sino para despertar en ti la chispa de la mejora continua, para que cada día te despiertes con la determinación de ser mejor que ayer.

Déjame animarte a sumergirte activamente en este proceso. Cada actividad es una oportunidad para crecer y sobresalir. Olvida la rutina y abraza el desafío; cada línea de código es una pieza que contribuye a la creación de tu obra maestra digital.

Motívate, porque este camino te transformará. Observa cómo cada pequeño paso te lleva más cerca de tus metas. Recuerda, este viaje es tuyo, y cada esfuerzo cuenta. Adelante, descubre tu potencial, abraza el aprendizaje y conquista este emocionante trayecto en Python. ¡Te esperan logros extraordinarios!

3. FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE

Hablar de "Python" es tal vez hablar de algo que, como el ingenio del programador, no tiene una frontera definida. Es así como día tras día, surgen nuevos elementos, desafíos y soluciones que apuntan a la excelencia en el desarrollo, implementación y optimización de código. No es suficiente tener conocimiento o ser experto en el manejo de un framework o paradigma de programación. Cada enfoque existente aporta su granito de arena en la búsqueda de ese alto nivel de "Eficiencia" y "Elegancia" en el mundo de Python.

3.1 Actividades de reflexión inicial:

3.1.1 Actividad : Fortalece tu pensamiento crítico y tu capacidad de análisis.

Observa el video "Si tuviera que empezar de nuevo, ¿qué aprendería?", comparte tu opinión en plenaria y escucha los aportes de tus compañeros.

Luego observa el Shorts "Que es Python", y crea una infografía donde expliques (qué es Python, características, ventajas y desventajas, comparación con Java y PHP y la importancia de Python en el desarrollo de soluciones software). Tu entrega será la infografía en formato digital

Ambiente requerido: Ambiente de aprendizaje SENA/ambiente virtual

Estrategias o técnicas didácticas activas: Plenaria, aprendizaje colaborativo, creación de infografías

Materiales de formación: Equipo de cómputo (PC) - Conexión a Internet

Material de apoyo: Video: "Si tuviera que empezar de nuevo, ¿qué aprendería?" Enlace: YouTube, Presentación en Línea: "Qué es Python" Enlace: YouTube Shorts, Software y herramientas en línea para la creación de infografías.

Duración de la actividad: 4horas.

3.2 Actividades de contextualización e identificación de conocimientos necesarios para el aprendizaje:

Para cualquier proyecto en el ámbito de la programación con Python, es crucial definir sus características, alcance, tiempos y recursos, entre otros aspectos clave. Asimismo, es esencial anticipar posibles desafíos y dificultades que puedan surgir durante el desarrollo. Establecer una forma efectiva de medir los resultados esperados se vuelve fundamental para mantener un control constante sobre el progreso y asegurar que se alcancen los objetivos planteados. En el vasto mundo de Python, donde la planificación y la anticipación son claves, esta metodología se convierte en un pilar para el éxito de cualquier proyecto de programación.

3.2.1. Actividad Señalar las sentencias de control en algoritmos que domina el aprendiz para la migración al lenguaje Python.

El instructor presentará 6 casos de estudio. Para cada uno, escribe qué sentencia de control (if, while, for, etc.) usarías y por qué.

Al finalizar, escribe en pocas líneas: ¿cómo te sentiste con la actividad y qué debes mejorar para avanzar en Python?

Ambiente requerido: Ambiente de aprendizaje SENA

Estrategias o técnicas didácticas activas: Aprendizaje basado en problemas (ABP).

Materiales de formación: Equipo de cómputo (PC) - Conexión a Internet

Material de apoyo: Casos de estudio, cuadernillo recursos sobre sentencias de control, IDE o editor de texto.

Duración de la actividad: 2 horas.

3.3 Actividades de apropiación:

Desarrolla la solución de software en el lenguaje de programación propuesto de acuerdo a los requerimientos del cliente.

3.3.1 Actividad: Conocer los entorno de trabajo Python e identificar los pasos para su instalación y configuración en los ambientes de desarrollo.

El instructor realiza una explicación guiada del proceso de instalación del lenguaje y del IDE. Posteriormente, orienta un recorrido por la interfaz del editor, mostrando las configuraciones y extensiones clave para programar en Python.

Instala la última versión de Python y el editor de código Visual Studio Code desde los sitios oficiales. Configura las extensiones necesarias en el editor para dejar tu ambiente de trabajo listo. Sitios de descarga: https://www.python.org/downloads/ https://code.visualstudio.com/#alt-downloads

Ambiente requerido: Ambiente de aprendizaje SENA.

Estrategias o técnicas didácticas activas: Aprendizaje guiado, aprendizaje práctico

Materiales de formación: Video beam o Pantalla, Mesas, Sillas, computadores, software requerido

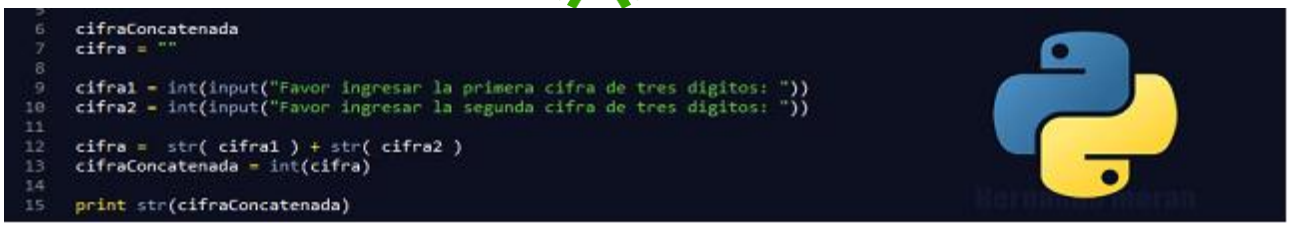
Material de apoyo: Cuadernillo con la guía de Instalación, Sitios web oficiales para descargar Python y https://www.python.org/downloads/, https://code.visualstudio.com/#alt-downloads

Evidencias de aprendizaje: Entorno de trabajo configurado.

Instrumentos de evaluación: Lista de desempeño

Duración de la actividad: 4 horas.

3.3.2 Actividad: Reconocer las estructuras básicas de programación en Python mediante ejercicios guiados.



Exploras los fundamentos esenciales del lenguaje. Cada ejercicio fortalece tu lógica y te acerca al dominio del código.

El instructor desarrolla una sesión explicativa donde aborda los fundamentos de la programación en Python, fortaleciendo el razonamiento lógico mediante ejemplos prácticos.

Temas tratados:

- Tipos de datos y los métodos básicos de formateo.
- Variables.
- Operadores aritméticos, lógicos y relacionales.

Resuelve los ejercicios propuestos en el cuadernillo entregado por el instructor "**Fundamentos del Lenguaje"**. Solicita apoyo cuando lo requieras; cada avance fortalece tu lógica y dominio del código.

Ambiente requerido: Ambiente de aprendizaje SENA.

Estrategias o técnicas didácticas activas: Aprendizaje Basado en Proyectos (Project-Based Learning) y Retroalimentación

Materiales de formación: Equipo de cómputo (PC) - Conexión a Internet - Software Especializado (Python, pip, IDLE y editor de código Visual studio Code o similares)

Material de apoyo: cuadernillo "**Fundamentos del Lenguaje**" con ejercicios prácticos.

Evidencias de aprendizaje: Presentación cuadernillo "**Fundamentos del Lenguaje**" resuelto.

Instrumentos de evaluación: Lista de chequeo de Desempeño y producto.

Duración de la actividad: 5 horas - 2 indirectas.

3.3.3 Actividad: Reconocer la sintaxis de las estructuras de control y repetición en Python mediante ejemplos guiados.

Aplica tu conocimiento previo en algoritmia para adaptar la lógica de las estructuras de control y repetición a la sintaxis de Python.

El instructor realiza una sesión guiada para explicar a través de ejemplos, la creación de algoritmos usando el lenguaje de programación Python en el marco de los siguientes temas:

- Control de flujo if-elif-else.
- Bucles while y for.

Resuelve los ejercicios propuestos por el instructor en el cuadernillo **"Estructuras de control y Ciclos"** teniendo en cuenta las orientaciones dadas. Presenta las soluciones implementadas a la clase. El instructor aclara dudas y retroalimentara el proceso.

Ambiente requerido: Ambiente de aprendizaje SENA.

Estrategias o técnicas didácticas activas: Aprendizaje Basado en Proyectos (Project-Based Learning) Aprendizaje Práctico.

Materiales de formación: Equipo de cómputo (PC) - Conexión a Internet - Software Especializado (Python, pip, IDLE y editor de código Visual studio Code o similares)

Material de apoyo: Cuadernillo**"Estructuras de control y Ciclos"** y ejercicios

Evidencias de aprendizaje: Presentación en clase de los ejercicios propuestos en el Cuadernillo **"Estructuras de control y Ciclos"**. Prueba de conocimiento Fundamentos de Python

Instrumentos de evaluación: lista de chequeo de desempeño, cuestionario

Duración de la actividad: 15 horas - 4 indirectas

3.3.4 Actividad: Reconocer la sintaxis de las colecciones de datos en Python mediante ejemplos guiados por el instructor

Estas colecciones nos brindarán un poder único para organizar, almacenar y manipular datos. Descubriremos cómo utilizar listas para secuencias dinámicas, cómo las tuplas nos ofrecen datos inmutables y ordenados, cómo los diccionarios establecen conexiones clave-valor, y cómo los conjuntos gestionan colecciones únicas y sin orden aparente. Utiliza las diferentes colecciones de datos para organizar, almacenar y manipular información de manera eficiente. Desarrolla los ejercicios propuestos por el instructor para afianzar el manejo de cada tipo de colección.

El instructor dirige una sesión guiada sobre colecciones en Python abordando los siguientes temas:

- Definición y uso de listas.
- Definición y uso de tuplas.
- Definición y uso de diccionarios.
- Definición y uso de conjuntos.

El instructor planteará ejercicios teniendo en cuenta los ejemplos de la sesión. Deberás resolver los ejercicios planteados en los cuadernillos entregados por el instructor “Colecciones en Python”, utilizando la estructura indicada para cada caso, contando con la asesoría y orientación del instructor. Finalizando el instructor aplica una prueba de conocimiento de la sintaxis Python aplicada hasta este momento en las soluciones implementadas en los cuadernillos de trabajo.

Ambiente requerido: Ambiente de aprendizaje SENA

Estrategias o técnicas didácticas activas: Aprendizaje Basado en Proyectos (Project-Based Learning) Aprendizaje Práctico

Materiales de formación: Equipo de cómputo (PC) - Conexión a Internet - Software Especializado (Python, pip, IDLE y editor de código Visual studio Code o similares)

Material de apoyo: Cuadernillo Colecciones en Python con ejercicios prácticos

Evidencias de aprendizaje: Socialización ejercicios Cuadernillo Colecciones en Python y Prueba de conocimiento.

Instrumentos de evaluación: Lista de chequeo de desempeño y de producto. Cuestionario.

Duración de la actividad: 20 horas - 4 indirectas

3.3.5 Actividad: Reconocer la sintaxis de las funciones, errores y excepciones en Python mediante una sesión explicativa

El instructor desarrolla una sesión explicativa sobre funciones y captura de errores en Python, abordando los siguientes temas:

- Definición y utilización de funciones.
- Control de errores usando Try-Except.

Posteriormente, el instructor comparte el cuadernillo con los ejercicios prácticos basados en los ejemplos presentados durante la sesión.

Define y utiliza funciones para modularizar tu código. Implementa mecanismos de control de errores para gestionar excepciones y hacer tus programas más robustos. Resuelve los ejercicios prácticos del cuadernillo “Funciones, Errores y Excepciones“ con la asesoría del instructor. Durante este proceso, cuentas con la asesoría del instructor para despejar cualquier duda que surja.

Ambiente requerido: Ambiente de aprendizaje SENA.

Estrategias o técnicas didácticas activas: Aprendizaje Basado en Proyectos (Project-Based Learning) Aprendizaje Práctico

Materiales de formación: Equipo de cómputo (PC) - Conexión a Internet - Software Especializado (Python, pip, IDLE y editor de código Visual studio Code o similares).

Material de apoyo: Cuadernillo “Funciones, Errores y Excepciones“ con ejercicios prácticos

Evidencias de aprendizaje: Presentación de los ejercicios cuadernillo “Funciones, Errores y Excepciones“

Instrumentos de evaluación: Lista de chequeo de desempeño y de producto

Duración de la actividad: 5 horas 1 indirectas.

3.3.6 Actividad: Explorar los conceptos de módulos, paquetes, PyPi, PIP, distribuibles y ejecutables en Python mediante exposiciones

En esta actividad, explora los conceptos fundamentales de módulos, paquetes, PyPi (Python Package Index), PIP (Python Package Installer), distribuibles y ejecutables en el contexto de Python.
El instructor organiza los equipos y asigna los temas a exponer:

- Introducción a módulos y paquetes.
- Uso de PyPi y PIP para la gestión de paquetes.
- Creación de distribuibles y ejecutables en Python.

Prepara y realiza una exposición sobre los conceptos asignados, presenta ejemplos prácticos y fomenta la resolución de ejercicios para reforzar el aprendizaje de tus compañeros. Durante la actividad, el instructor actúa como moderador, complementa la información y orienta el debate técnico.

A continuación, resuelve los ejercicios entregados por el instructor en el cuadernillo “Módulos y paquetes” y socializa con tus compañeros las soluciones implementadas, el instructor acompañara tu proceso despejando las dudas.

Ambiente requerido: Ambiente de aprendizaje SENA.
Estrategias o técnicas didácticas activas: Aprendizaje Basado en Proyectos (Project-Based Learning) Aprendizaje Práctico
Materiales de formación: Equipo de cómputo (PC) - Conexión a Internet - Software Especializado (Python, pip, IDLE y editor de código Visual studio Code o similares)
Material de apoyo: Cuadernillo Módulos y paquetes con ejercicios prácticos
Evidencias de aprendizaje: Presentación en clase de las exposiciones y cuadernillo Módulos y paquetes resuelto.
Instrumentos de evaluación: Lista de chequeo de desempeño y de producto
Duración de la actividad: 10 horas + 2 indirectas.

3.3.7 Actividad: POO – Explorar la programación orientada a objetos en Python mediante un proyecto colaborativo

La programación orientada a objetos (Object Oriented Programming o OOP) es un paradigma de programación organizado por objetos, que consisten en datos y funciones. Entre estos objetos, se pueden establecer relaciones como herencia, cohesión, abstracción, polimorfismo y encapsulamiento. Esto proporciona una gran flexibilidad, permitiendo la creación de objetos que pueden heredarse y transmitirse sin la necesidad de modificaciones constantes.

El instructor socializara los conceptos principales del paradigma orientado a objetos como son:

- Clase y Métodos
- Relaciones (Herencia, herencia múltiple, asociación, agregación, composición)

Organízate en equipos de 3 personas y asume un rol (diseñador, programador, documentador). Desarrolla el proyecto asignado aplicando los conceptos de la POO. Presenta los avances periódicos y recibe el feedback de los otros equipos.

Al finalizar la actividad presenta tu proyecto terminado, explica cómo implementaste los conceptos de POO, el instructor acompañara permanentemente la sesión resolviendo preguntas y despejando las dudas que se generen.

Estrategias o técnicas didácticas activas: Aprendizaje Basado en Proyectos (Project-Based Learning) Trabajo en Equipo

Materiales de formación: Equipo de cómputo (PC) - Conexión a Internet - Software Especializado (Python, pip, IDLE y editor de código Visual studio Code o similares)

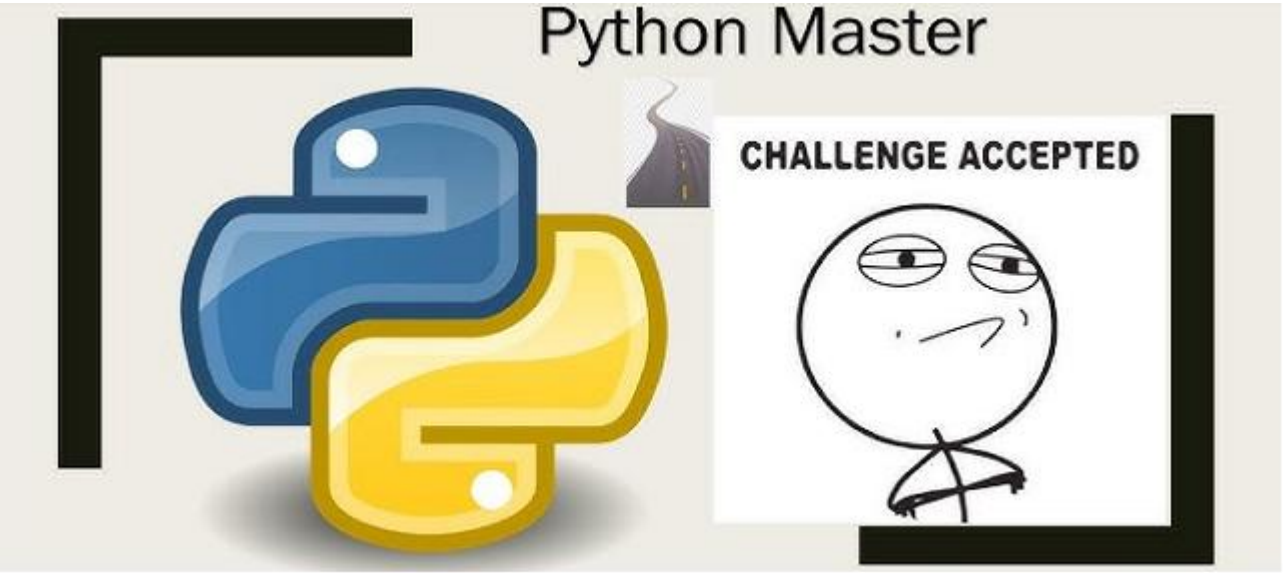
Material de apoyo: Cuadernillo de trabajo con ejercicios prácticos

Evidencias de aprendizaje: Presentación del proyecto colaborativo y código desarrollado

Instrumentos de evaluación: Lista de desempeño y producto

Duración de la actividad: 15 horas - 4 indirectas

3.3.8 Actividad: Comprender el paradigma funcional en Python mediante la resolución de retos.



Python es un lenguaje de programación Multiparadigma.

La Programación Funcional (PF) es un paradigma de programación donde las protagonistas son las funciones. Las funciones se pueden almacenar en variables, pasar como parámetros a otras funciones, ser retornadas por otras funciones.

El instructor introduce los conceptos del paradigma funcional mediante ejemplos y ejercicios prácticos. Los temas por tratar incluyen:

- Composición de Funciones, funciones anidadas y closures.
- Funciones como variables, parámetros y valores de retorno.
- Decoradores, generadores, y comprensión de listas.
- Funciones Lambda y de orden Superior

Resuelve en equipo una serie de retos de programación, utilizando y combinando los conceptos de la programación funcional en Python para encontrar soluciones eficientes.
Después de cada reto, los equipos recibirán feedback detallado del instructor sobre sus soluciones, destacando las fortalezas y áreas de mejora. El instructor proporcionará apoyo continuo, actuando como mentor y facilitador durante el proceso de resolución de los retos.

Ambiente requerido: Ambiente de aprendizaje SENA
Estrategias o técnicas didácticas activas: Aprendizaje Basado en Retos (Challenge-Based Learning) Trabajo en Equipo.
Materiales de formación: Equipo de cómputo (PC) - Conexión a Internet - Software Especializado (Python, pip, IDLE y editor de código Visual studio Code o similares)
Material de apoyo: Cuadernillo de trabajo con ejercicios prácticos
Evidencias de aprendizaje: Presentación de la solución Reto y código desarrollado
Instrumentos de evaluación: Lista de Chequeo de Desempeño y Producto
Duración de la actividad: 15 horas - 4 indirectas

3.4 Actividades de Transferencia el Conocimiento:

Implementar la solución de software en el lenguaje de programación propuesto que permitan satisfacer los requerimientos del cliente

3.4.1 Actividad: Aplicar conceptos de migración a Python para la construcción de software mediante ejercicio práctico

Aplica los conocimientos adquiridos para desarrollar un módulo de administración de usuarios que realice las cuatro operaciones CRUD sobre la base de datos del proyecto formativo. Escribe los algoritmos en archivos independientes con extensión .py y socializa tu solución en clase.

El instructor define los requerimientos del módulo CRUD. Durante el desarrollo, acompaña la actividad despejando dudas y generando opciones de mejora. Finalmente, fija una fecha límite de entrega.

Ambiente requerido: Ambiente de aprendizaje SENA.

Estrategias o técnicas didácticas activas: Aprendizaje Basado en Proyectos (Project-Based Learning) Trabajo en Equipo, Interacción y Soporte

Materiales de formación: Equipo de cómputo (PC, Tablet o Móvil, etc.) - Conexión a Internet - Software Especializado (Python, pip, IDLE y editor de código Visual studio Code o similares)

Material de apoyo: Cuadernillo de trabajo con ejercicios prácticos

Evidencias de aprendizaje: Presentación en clase módulo de administración de usuarios.

Instrumentos de evaluación: Lista de Chequeo de Desempeño y Producto

Duración de la actividad: 15 horas - 4 indirectas

4. PLANTEAMIENTO DE EVIDENCIAS DE APRENDIZAJE PARA LA EVALUACIÓN EN EL PROCESO FORMATIVO.

Fase del proyecto formativo	Actividad del proyecto formativo	Actividad de Aprendizaje	Evidencias de Aprendizaje	Criterios de Evaluación	Técnicas e Instrumentos de Evaluación
Ejecución	Construir los componentes de la solución informática de acuerdo a las características funcionales y de calidad del diseño	Desarrolla la solución de software en el lenguaje de programación propuesto de acuerdo a los requerimientos del cliente. Implementar la solución de software en el lenguaje de programación propuesto que permitan	**Evidencias de Conocimiento** Prueba de conocimientos **Evidencias de Desempeño:** Entorno de trabajo configurado Socialización de los cuadernillos Fundamentos del Lenguaje, Estructuras de control y Ciclos, Colecciones en Python, Funciones, Errores y	Codifica los módulos del software stand-alone, web y móvil, de acuerdo con las Especificaciones del diseño y el estándar de codificación. Crea servicios web para disponer de	Formulación de preguntas /Cuestionario. Observación. Listas de chequeo desempeño. Valoración del producto

		satisfacer los requerimientos del cliente	Excepciones, Módulos y paquetes Exposiciones Presentación del proyecto colaborativo Presentación de la solución Reto Presentación en clase módulo de administración de usuarios. **Evidencias de Producto:** Cuadernillo "Fundamentos del Lenguaje" Cuadernillo "Colecciones en Python" Cuadernillo "Funciones, Errores y Excepciones" Cuadernillo "Módulos y paquetes" Proyecto colaborativo Reto Python Módulo de administración de usuarios.	métodos reutilizables en el software. Incorpora tecnologías emergentes y disruptivas de acuerdo con los propósitos del software.	Lista de chequeo Producto

5. GLOSARIO DE TÉRMINOS

Archivo: Una colección de datos almacenados en una computadora. En Python, los archivos pueden ser manipulados para leer, escribir y modificar datos.

Base de Datos: Un sistema organizado para almacenar, gestionar y recuperar información estructurada. Python puede interactuar con bases de datos utilizando bibliotecas específicas como SQLite y MySQLdb.

Closure: Una función anidada que recuerda el entorno en el cual fue creada. Esto permite que la función anidada acceda a las variables locales de la función externa incluso después de que la función externa haya finalizado su ejecución.

Comprensión de Listas: Una sintaxis compacta para crear listas de Python. Permite generar nuevas listas aplicando una expresión a cada elemento de una secuencia.

CRUD: Acrónimo de Create, Read, Update, Delete. Se refiere a las cuatro operaciones básicas de almacenamiento persistente.

Decorador: Una función que recibe otra función como argumento y devuelve una nueva función mejorada. Los decoradores se utilizan para modificar el comportamiento de las funciones o métodos.

Excepción: Un evento durante la ejecución de un programa que interrumpe el flujo normal de las instrucciones. Las excepciones se manejan mediante bloques try y except en Python.

Generador: Una función que devuelve un iterador que produce una secuencia de valores en lugar de un solo valor, utilizando la palabra clave yield.

Intérprete: Programa que ejecuta código fuente de Python directamente, línea por línea, en lugar de compilarlo previamente a código máquina.

Lambda: Una función anónima en Python, definida usando la palabra clave lambda. Estas funciones pueden tener cualquier número de argumentos pero solo una expresión.

Lenguaje Multiparadigma: Un lenguaje de programación que soporta múltiples estilos de programación, como la programación orientada a objetos, la programación funcional y la programación imperativa.

Módulo: Un archivo de Python que contiene definiciones y declaraciones. Un módulo puede definir funciones, clases y variables, y también puede incluir código ejecutable.

Orientación a Objetos (OOP): Un paradigma de programación que organiza el diseño de software en torno a datos, o "objetos", en lugar de funciones y lógica. Los objetos pueden contener datos, en forma de campos, y código, en forma de procedimientos, conocidos como métodos.

Paquete: Una forma de estructurar los módulos de Python mediante un directorio jerárquico que contiene módulos y un archivo especial __init__.py.

PIP: El instalador de paquetes de Python, que permite instalar y gestionar paquetes y sus dependencias.

Programación Imperativa: Un paradigma de programación que usa declaraciones que cambian el estado de un programa. En este paradigma, un programa se compone de una secuencia de instrucciones que indican al ordenador cómo realizar una tarea.

PyPI (Python Package Index): Un repositorio en línea de software para el lenguaje de programación Python. PyPI ayuda a encontrar y usar software desarrollado por la comunidad Python.

Python: Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma, ya que soporta parcialmente la orientación a objetos, programación imperativa y, en menor medida, programación funcional.

Sintaxis: Las reglas que definen la estructura correcta de las declaraciones y expresiones en un lenguaje de programación.

Tkinter: Una biblioteca estándar de Python para crear interfaces gráficas de usuario (GUI).

Variable: Un nombre que se asigna a un valor y que se puede usar para referirse a ese valor más tarde en el programa.

6. REFERENTES BILBIOGRÁFICOS

Ramírez, Ó, Python a fondo. Marcombo, Octubre 2021, Alphaeditorial. https://www.alphaeditorialcloud.com/library/publication/python-a-fondo

Salazar Perdomo, P.. Empezando a programar en Python. Escuela Colombiana de Ingeniería Julio Garavito, 2024. Digitalia, https://www-digitaliapublishing-com.bdigital.sena.edu.co/a/144677

Sarasa Cabezuelo, A.. Gestión de la información web usando Python. Universitat Oberta de Catalunya, 2017. Digitalia, https://www-digitaliapublishing-com.bdigital.sena.edu.co/a/47344

Documentación Oficial Python. https://www.python.org/

Libreria estandar de python https://docs.python.org/3/library/index.html.

Repositorio oficial de python PyPI Python Package Index

7. CONTROL DEL DOCUMENTO

	Nombre	Cargo	Dependencia	Fecha
Autor (es)	Wilfredo Hernando Moran Gordillo	Instructor	CEAI	Octubre 2025

	Diego Fernando Lenis Z. Olga Patricia Moreno G.			

8. CONTROL DE CAMBIOS (diligenciar únicamente si realiza ajustes a la guía)

	Nombre	Cargo	Dependencia	Fecha	Razón del Cambio
Autor (es)					