**PROGRAMACIÓN**

**SENA**

**CMM**

**INSTRUCTOR**

Yerman avila

**INTEGRANTES**

Michael santiago cardona contreras

David santiago moreno rodriguez

**ACTIVIDAD**

*consultar:*

1. ¿Qué es un lenguaje de programación?

2. ¿Qué diferencias hay entre un intérprete y un compilador?

3. ¿Qué es un programa?

4. Tipos de errores en programación

5. ¿Qué es Debugging?

6. ¿Qué es python?

7. Explique variables, tipos de datos, operaciones y expresiones básicas en Python.

***SOLUCIÓN***

***2.*** Un lenguaje de programación es un conjunto de reglas y símbolos que se utilizan para escribir programas de computadora.

Estos programas son conjuntos de instrucciones que le dicen a una computadora qué hacer, cómo hacerlo y en qué orden hacerlo.

Los lenguajes de programación se utilizan para comunicarse con las computadoras y para escribir software que puede realizar tareas específicas.

Un lenguaje de programación proporciona una forma estructurada y legible para expresar algoritmos y lógica de programación.

Los programadores utilizan lenguajes de programación para crear aplicaciones, juegos, sistemas operativos, software de productividad, aplicaciones web y una amplia variedad de software personalizado.

Algunos ejemplos de lenguajes de programación populares incluyen Python, Java, C++, C#, JavaScript, Ruby, PHP y muchos más.

Cada lenguaje tiene sus propias reglas sintácticas y semánticas, así como su propio conjunto de bibliotecas y herramientas que facilitan el desarrollo de software en ese lenguaje.

Los programadores eligen un lenguaje de programación en función de sus necesidades y preferencias, así como de los requisitos específicos del proyecto en el que están trabajando.

Las diferencias clave entre un intérprete y un compilador radican en la forma en que procesan y ejecutan el código fuente de un programa.

Compilador:

Proceso de Traducción: Un compilador es un programa que traduce todo el código fuente de un programa en un lenguaje de alto nivel a código de máquina (código binario) en un solo paso. Esta traducción se realiza por completo antes de que el programa se ejecute.

Generación de Archivos Binarios: Después de la fase de compilación, se genera un archivo binario (a menudo un archivo ejecutable) que contiene el código de máquina directamente. Este archivo se puede ejecutar sin necesidad del código fuente original.

Eficiencia: Los programas compilados tienden a ser más eficientes en términos de rendimiento, ya que el código de máquina se optimiza durante la compilación.

Errores: Los errores se detectan durante la fase de compilación, lo que significa que el programa no se ejecutará hasta que se hayan corregido todos los errores en el código fuente.

Ejemplos de lenguajes compilados: C, C++, Rust, Go.

Intérprete:

Proceso de Ejecución: Un intérprete es un programa que ejecuta el código fuente línea por línea o instrucción por instrucción. No traduce el código fuente a código de máquina por adelantado; en su lugar, lo interpreta y ejecuta en tiempo real.

No se genera un archivo binario: No se crea un archivo binario separado; el código fuente se interpreta directamente sin una fase de compilación.

Facilidad de depuración: Los errores se detectan y se informan durante la ejecución del programa, lo que facilita la depuración.

Portabilidad: Los programas interpretados suelen ser más portátiles, ya que el código fuente se puede ejecutar en cualquier plataforma con un intérprete compatible.

Ejemplos de lenguajes interpretados: Python, JavaScript, Ruby, PHP.

En resumen, un compilador traduce todo el código fuente a código de máquina de una vez y genera un archivo binario, mientras que un intérprete ejecuta el código fuente línea por línea o instrucción por instrucción sin generar un archivo binario separado. Cada enfoque tiene sus ventajas y desventajas, y la elección entre un compilador y un intérprete a menudo depende de las necesidades y requisitos específicos del proyecto y del lenguaje de programación utilizado.

3. Un programa, en el contexto de la informática y la programación, es un conjunto de instrucciones escritas en un lenguaje de programación que le indican a una computadora cómo realizar una tarea específica. Estas instrucciones se organizan de manera lógica y secuencial para que la computadora las pueda seguir y ejecutar.

Los programas pueden abordar una amplia variedad de tareas y objetivos, desde simples cálculos matemáticos hasta operaciones complejas, como la gestión de bases de datos, la creación de interfaces de usuario, el procesamiento de imágenes, la ejecución de algoritmos, la automatización de tareas, la simulación de sistemas, y mucho más.

Un programa puede ser tan breve como unas pocas líneas de código o tan extenso como un sistema operativo completo. Dependiendo de su complejidad y propósito, los programas pueden ser escritos en diferentes lenguajes de programación, como C++, Python, Java, JavaScript, entre otros.

En resumen, un programa informático es un conjunto de instrucciones escritas por un programador para que una computadora realice una tarea específica o resuelva un problema determinado. Los programas son fundamentales en la informática y son la base de todo el software que utilizamos en nuestras computadoras y dispositivos electrónicos.

4. En programación, existen varios tipos de errores que pueden ocurrir durante la escritura, compilación o ejecución de un programa.

Estos errores se dividen generalmente en tres categorías principales: errores de sintaxis, errores de tiempo de ejecución y errores lógicos.

Errores de Sintaxis:

Descripción: Los errores de sintaxis son errores que ocurren cuando el código no sigue la estructura gramatical o las reglas del lenguaje de programación. Estos errores son detectados por el compilador o el intérprete durante la fase de compilación o interpretación.

Ejemplos: Falta de paréntesis o llaves, uso incorrecto de operadores, declaraciones incompletas, nombres de variables no válidos, entre otros.

Efecto: El programa no se compila o no se ejecuta hasta que se corrijan los errores de sintaxis.

Errores de Tiempo de Ejecución:

Descripción: Los errores de tiempo de ejecución ocurren durante la ejecución del programa y a menudo están relacionados con problemas en los datos de entrada o condiciones inesperadas.

Estos errores pueden no ser detectados durante la compilación, pero hacen que el programa falle durante la ejecución.

Ejemplos: División por cero, acceso a un índice fuera de los límites de un arreglo, falta de memoria, entre otros.

Efecto: El programa se detiene o arroja una excepción durante la ejecución. Estos errores a menudo requieren una gestión adecuada para evitar que el programa se bloquee.

Errores Lógicos:

Descripción: Los errores lógicos, a veces llamados "bugs", son errores en la lógica del programa. El código se ejecuta sin problemas, pero no produce el resultado deseado debido a un error en el algoritmo o la secuencia de instrucciones.

Ejemplos: Calcular una fórmula incorrecta, condiciones mal planteadas en una declaración "if", algoritmo ineficiente, entre otros.

Efecto: El programa se ejecuta, pero no produce el resultado esperado. Estos errores a menudo requieren la depuración y la revisión del código para encontrar y corregir el problema.

Es importante notar que los errores de sintaxis se detectan antes de la ejecución del programa, mientras que los errores de tiempo de ejecución y los errores lógicos se manifiestan durante la ejecución. La depuración es el proceso de identificar y corregir errores de tiempo de ejecución y errores lógicos en el código. La depuración es una parte fundamental del desarrollo de software para garantizar que un programa funcione según lo previsto.

5. El "debugging" es el proceso de identificar, analizar y corregir errores o fallos en un programa de computadora. Los errores en un programa se denominan "bugs", y el acto de eliminarlos se conoce comúnmente como "depuración" o "debugging".

El debugging es una parte esencial del desarrollo de software, ya que ningún programa es perfecto desde la primera vez que se escribe. Los programadores utilizan técnicas de depuración para encontrar y solucionar diversos tipos de problemas, como errores de sintaxis, errores de tiempo de ejecución y errores lógicos que pueden provocar un comportamiento incorrecto del programa. Aquí hay algunas actividades comunes en el proceso de debugging:

Identificación del error: El primer paso es detectar el error o el comportamiento inesperado en el programa. Esto puede incluir la revisión de mensajes de error, registros de eventos y pruebas del programa en busca de problemas.

Reproducción del error: Los programadores intentan reproducir el error de manera consistente para comprender las circunstancias que lo causan. Esto implica proporcionar los mismos datos de entrada y condiciones que llevaron al error.

Análisis de código: El programador examina el código en busca de posibles fuentes de error. Esto puede incluir la revisión de las partes relevantes del código y la búsqueda de problemas de sintaxis, lógica o algoritmos incorrectos.

Uso de herramientas de depuración: Las herramientas de depuración son software diseñado específicamente para ayudar a los programadores a encontrar y corregir errores. Estas herramientas permiten examinar el estado del programa en tiempo de ejecución, establecer puntos de interrupción, inspeccionar variables y ejecutar el código paso a paso.

Modificación y corrección del código: Una vez que se identifica el error, el programador realiza cambios en el código para corregirlo. Esto puede implicar ajustar variables, modificar algoritmos o reescribir secciones de código.

Prueba y verificación: Después de realizar las correcciones, se prueban nuevamente el programa para asegurarse de que el error se haya resuelto sin introducir nuevos problemas.

Documentación: Es importante llevar un registro de los errores encontrados, las soluciones aplicadas y cualquier lección aprendida durante el proceso de debugging. Esto puede ser útil para futuras referencias y para colaboración con otros miembros del equipo.

La depuración es una habilidad esencial para cualquier programador, ya que permite mantener y mejorar la calidad del software. A medida que los programas se vuelven más complejos, la depuración se convierte en una parte integral del ciclo de desarrollo de software.

6. Python es un lenguaje de programación de alto nivel, multipropósito, interpretado y de código abierto. Fue creado por Guido van Rossum y lanzado por primera vez en 1991. Python se ha vuelto extremadamente popular en la comunidad de desarrollo de software debido a su sintaxis clara y legible, su facilidad de uso y su versatilidad. Algunas de las características clave de Python incluyen:

Sintaxis legible: Python se enorgullece de tener una sintaxis clara y fácil de leer, que utiliza espaciado en lugar de llaves o paréntesis para definir bloques de código. Esto facilita la escritura y lectura de código.

Multipropósito: Python es un lenguaje versátil que se utiliza en una amplia gama de aplicaciones, desde desarrollo web y aplicaciones de escritorio hasta ciencia de datos, inteligencia artificial, automatización de tareas y más.

Interpretado: Python es un lenguaje interpretado, lo que significa que el código fuente se ejecuta línea por línea en tiempo real por un intérprete en lugar de ser compilado a código de máquina antes de la ejecución. Esto facilita la experimentación y la depuración.

Código abierto: Python es de código abierto, lo que significa que su código fuente es accesible y modificable por la comunidad de programadores. Esto ha llevado al desarrollo de una amplia variedad de bibliotecas y módulos de terceros que amplían las capacidades del lenguaje.

Gran comunidad y bibliotecas: Python tiene una comunidad activa y un ecosistema de bibliotecas y marcos de trabajo robusto. Esto facilita el desarrollo de aplicaciones y la reutilización de código.

Portabilidad: Python es compatible con múltiples plataformas, lo que significa que el mismo código fuente puede ejecutarse en diferentes sistemas operativos sin modificaciones significativas.

Python se ha convertido en un lenguaje popular en áreas como la ciencia de datos, aprendizaje automático, desarrollo web, automatización, desarrollo de juegos y más. Es conocido por su enfoque en la legibilidad del código, lo que lo hace accesible para programadores novatos y experimentados por igual. Además, Python cuenta con una amplia base de usuarios y una gran cantidad de recursos en línea, lo que lo convierte en una excelente opción para aquellos que deseen aprender a programar.

7. En Python, las variables, los tipos de datos, las operaciones y las expresiones son conceptos fundamentales que se utilizan para almacenar, manipular y procesar información en programas.

Variables:

Una variable es un nombre que se utiliza para almacenar datos en la memoria de un programa.

En Python, no es necesario declarar explícitamente el tipo de una variable; Python infiere el tipo automáticamente.

Se asigna un valor a una variable mediante el operador "=".

x = 5

nombre = "Juan"

Tipos de Datos:

Los tipos de datos en Python representan el tipo de valor que puede contener una variable. Algunos de los tipos de datos comunes en Python incluyen:

Enteros (int): Números enteros, como 5 o -2.

Flotantes (float): Números decimales, como 3.14 o -0.5.

Cadenas (str): Secuencias de caracteres, como "Hola, mundo".

Listas: Colecciones ordenadas de elementos, como [1, 2, 3].

Tuplas: Colecciones ordenadas inmutables, como (1, 2, 3).

Booleanos (bool): Representan valores verdaderos (True) o falsos (False).

Operaciones:

En Python, puedes realizar una variedad de operaciones matemáticas y lógicas en variables. Algunas operaciones básicas incluyen:

Suma (+): Realiza una adición, por ejemplo, a + b.

Resta (-): Realiza una sustracción, por ejemplo, a - b.

Multiplicación (\*): Realiza una multiplicación, por ejemplo, a \* b.

División (/): Realiza una división, por ejemplo, a / b.

Módulo (%): Devuelve el residuo de una división, por ejemplo, a % b.

Potencia (\*): Realiza una potenciación, por ejemplo, a \* b.

Expresiones Básicas:

Una expresión es una combinación de valores y operadores que produce un resultado. Las expresiones se evalúan para obtener un valor.

suma = 5 + 3 # La expresión se evalúa a 8.

mensaje = "Hola, " + "mundo" # La expresión se evalúa a "Hola, mundo".

Estos conceptos son esenciales para escribir programas en Python. Puedes utilizar variables para almacenar datos, aplicar operaciones para manipular esos datos y construir expresiones para realizar cálculos y operaciones lógicas. La comprensión de estos conceptos es fundamental para trabajar de manera efectiva con Python y otros lenguajes de programación.

***BIBLIOGRAFIA:***

1. [***https://executrain.com.mx/que-es-programacion/#:~:text=La%20programaci%C3%B3n%20es%20la%20actividad,entre%20ordenadores%20y%20los%20usuarios***](https://executrain.com.mx/que-es-programacion/#:~:text=La%20programaci%C3%B3n%20es%20la%20actividad,entre%20ordenadores%20y%20los%20usuarios)***.***
2. [***https://www.freecodecamp.org/espanol/news/lenguajes-compilados-vs-interpretados/#:~:text=Los%20compiladores%20e%20int%C3%A9rpretes%20toman,por%20la%20m%C3%A1quina%20de%20destino***](https://www.freecodecamp.org/espanol/news/lenguajes-compilados-vs-interpretados/#:~:text=Los%20compiladores%20e%20int%C3%A9rpretes%20toman,por%20la%20m%C3%A1quina%20de%20destino)***.***
3. [***https://desarrollarinclusion.cilsa.org/tecnologia-inclusiva/que-es-un-programa/***](https://desarrollarinclusion.cilsa.org/tecnologia-inclusiva/que-es-un-programa/)
4. [***https://rockcontent.com/es/blog/tipos-de-lenguaje-de-programacion/***](https://rockcontent.com/es/blog/tipos-de-lenguaje-de-programacion/)
5. [***https://www.hostgator.mx/blog/que-es-debug-en-programacion/***](https://www.hostgator.mx/blog/que-es-debug-en-programacion/)
6. [***https://aws.amazon.com/es/what-is/python/#:~:text=Python%20es%20un%20lenguaje%20de,ejecutar%20en%20muchas%20plataformas%20diferentes***](https://aws.amazon.com/es/what-is/python/#:~:text=Python%20es%20un%20lenguaje%20de,ejecutar%20en%20muchas%20plataformas%20diferentes)***.***
7. [***https://www.tokioschool.com/noticias/caracteristicas-principales-de-python/***](https://www.tokioschool.com/noticias/caracteristicas-principales-de-python/)