**1.** **¿Qué es un computador?**  
Un computador es un dispositivo electrónico que procesa datos a través de instrucciones programadas, permitiendo realizar tareas como cálculos, almacenamiento de información y comunicación. Está compuesto por hardware (partes físicas) y software (programas que controlan el hardware). Los computadores pueden variar en tamaño, desde dispositivos portátiles hasta grandes servidores, y están diseñados para ejecutar una amplia gama de aplicaciones.

**2.** **¿Qué es un programa?**  
Un programa es un conjunto de instrucciones escritas en un lenguaje comprensible para las computadoras, que permite ejecutar una tarea específica. Estas instrucciones le indican al computador cómo realizar una operación o conjunto de operaciones. Un programa puede ser desde una simple calculadora hasta sistemas complejos como videojuegos o plataformas de redes sociales, funcionando bajo la lógica de codificación.

**3.** **Lógica de programación**  
La lógica de programación se refiere al pensamiento estructurado necesario para desarrollar soluciones a problemas mediante programación. Implica la planificación y organización de las instrucciones de manera lógica y eficiente, considerando los pasos que la computadora debe seguir para alcanzar un resultado. Esto incluye decidir el orden de operaciones, identificar condiciones y iteraciones, y manejar la información de forma correcta.

**4.** **Algoritmos**  
Un algoritmo es una secuencia de pasos o instrucciones que resuelven un problema específico. Es una solución general y estructurada que se puede aplicar a diferentes casos, y puede expresarse en forma de pseudocódigo, diagramas de flujo o lenguajes de programación. Los algoritmos son fundamentales en la programación, pues definen cómo debe proceder el sistema para obtener los resultados deseados.

**5.** **Variables**  
Las variables son contenedores de datos que se utilizan en programación para almacenar valores temporales que pueden cambiar durante la ejecución del programa. Dependiendo del tipo de datos que almacenan, las variables pueden ser de tipo entero (números sin decimales), decimal (números con decimales), o texto (cadenas de caracteres). Se les asigna un nombre para referirse a ellas y se usan para manipular datos dentro del programa.

**6.** **Tipos de datos**  
Los tipos de datos en programación definen qué tipo de valores puede almacenar una variable. Los tres tipos principales son: números, que pueden ser enteros (como 5) o flotantes (como 3.14); cadenas de texto, que son secuencias de caracteres (como "hola"); y booleanos, que solo pueden tener dos valores: verdadero (True) o falso (False). Cada tipo de dato tiene propiedades y operaciones específicas que pueden realizarse sobre él.

**7.** **Operadores**  
Los operadores son símbolos utilizados para realizar operaciones sobre variables y valores. Los operadores aritméticos incluyen la suma (+), resta (-), multiplicación (\*), y división (/), permitiendo realizar cálculos. Los operadores lógicos, como AND, OR y NOT, se utilizan para combinar expresiones booleanas y tomar decisiones en base a condiciones, cruciales para la creación de estructuras condicionales en programación.

**8.** **Estructuras de control**  
Las estructuras de control son elementos fundamentales en programación que permiten dirigir el flujo de ejecución del programa según ciertas condiciones. Un ejemplo básico son las estructuras condicionales como if y else, que permiten ejecutar diferentes bloques de código dependiendo de si una condición se cumple o no. Estas estructuras facilitan la toma de decisiones dentro del programa, haciendo que se comporten de forma dinámica.

**9.** **Bucles**  
Un bucle es una estructura que permite repetir un bloque de código varias veces bajo ciertas condiciones. Los tipos más comunes son el bucle for, que repite un número fijo de veces, y el bucle while, que se ejecuta mientras se cumpla una condición. Los bucles son esenciales para manejar tareas repetitivas, como recorrer listas de datos o realizar cálculos en iteraciones sucesivas.

**10.** **Entrada y salida**  
La entrada y salida son procesos que permiten la interacción de un programa con el usuario o con otros sistemas. La entrada (input) es cuando un programa recibe datos del usuario, como al pedirle su nombre o una opción de menú. La salida (output) es cuando el programa muestra resultados o mensajes al usuario, como un cálculo o un mensaje de error. Estas interacciones son clave para la utilidad de un programa.

**11.** **Funciones básicas**  
Las funciones son bloques de código que realizan una tarea específica y pueden ser reutilizados en diferentes partes de un programa. Esto permite modularizar el código, hacerlo más eficiente y fácil de mantener. Las funciones pueden recibir parámetros y devolver resultados, lo que las hace flexibles para resolver una variedad de problemas de forma sencilla.

**12.** **Primer lenguaje**  
Python es un lenguaje de programación fácil de aprender y muy popular entre los principiantes. Su sintaxis es clara y legible, lo que lo hace ideal para principiantes. Las estructuras básicas incluyen variables, operadores, estructuras de control como if y for, y funciones. Python también tiene una amplia variedad de bibliotecas para facilitar tareas complejas como el análisis de datos, desarrollo web, y más.

**13.** **Comentarios en el código**  
Los comentarios en el código son anotaciones que los programadores incluyen dentro del código fuente para explicar su funcionamiento o hacer notas importantes. Los comentarios no afectan la ejecución del programa, ya que son ignorados por la computadora. Son esenciales para la documentación y comprensión del código, especialmente cuando se trabaja en equipo o se necesita mantener el código a largo plazo.

**14.** **Errores comunes**  
Los errores en programación se dividen principalmente en dos categorías: errores de sintaxis (syntax errors) y errores en tiempo de ejecución (runtime errors). Los primeros ocurren cuando hay un problema en la forma del código, como una falta de paréntesis o un mal uso de palabras clave. Los segundos ocurren cuando el programa se ejecuta, pero no puede continuar debido a un problema, como la división por cero o intentar acceder a una variable que no existe.

**15.** **Depuración**  
La depuración es el proceso de identificar y corregir errores en el código. Los programadores utilizan herramientas como depuradores o imprimen mensajes de print en el código para rastrear el flujo de ejecución y los valores de las variables. Esto ayuda a detectar qué parte del código está causando un error y cómo solucionarlo. La depuración es una habilidad esencial para mejorar la calidad del software.

**16.** **Compiladores vs. intérpretes**  
La diferencia principal entre compiladores e intérpretes radica en cómo traducen el código fuente a instrucciones ejecutables. Los compiladores traducen todo el código a un archivo ejecutable antes de que se ejecute, lo que puede mejorar la velocidad del programa. Los intérpretes, por otro lado, traducen el código línea por línea en tiempo de ejecución, lo que hace que el desarrollo sea más rápido pero puede ser más lento en ejecución.

**17.** **Cadenas de texto**  
Las cadenas de texto son secuencias de caracteres, como palabras o frases, que se almacenan en una variable. En programación, se pueden manipular mediante operaciones como concatenación (unir cadenas), obtener la longitud de la cadena, o buscar subcadenas dentro de ellas. Estas operaciones son fundamentales para manejar información textual en cualquier aplicación.

**18.** **Hardware básico**  
El hardware básico de un computador incluye la CPU (Unidad Central de Procesamiento), que ejecuta las instrucciones; la memoria RAM, que almacena temporalmente datos y programas activos; y el disco duro, que guarda la información de forma permanente. Estos componentes trabajan juntos para realizar tareas de procesamiento, almacenamiento y recuperación de datos.

**19.** **Software**  
El software se divide en dos categorías principales: el sistema operativo y las aplicaciones. El sistema operativo, como Windows o Linux, gestiona los recursos del hardware y proporciona servicios básicos para ejecutar programas. Las aplicaciones son programas diseñados para tareas específicas, como navegadores web, procesadores de texto o programas de edición de imágenes.

**20.** **Sistemas operativos**  
Un sistema operativo es el software fundamental que gestiona el hardware y proporciona servicios a las aplicaciones. Ejemplos comunes son Windows, Linux y macOS. Cada uno tiene características propias, como la interfaz de usuario, la gestión de archivos y la seguridad. Windows es conocido por su facilidad de uso, mientras que Linux es preferido por desarrolladores y servidores debido a su flexibilidad y control.

**21.** **Archivos y carpetas**  
En un sistema operativo, los archivos son unidades de almacenamiento que contienen datos, como documentos, imágenes o programas. Los archivos se organizan dentro de carpetas (o directorios), que permiten agruparlos de manera lógica. La estructura jerárquica de carpetas facilita la organización y búsqueda de información en el sistema. Las carpetas pueden contener otros archivos o subcarpetas, creando un sistema de organización.

**22.** **Terminal o consola**  
La terminal o consola es una herramienta que permite interactuar con el sistema operativo a través de comandos de texto. A través de ella, los usuarios pueden realizar tareas como navegar por el sistema de archivos, ejecutar programas y gestionar configuraciones. Algunos comandos básicos incluyen cd (para cambiar directorios), dir o ls (para listar archivos), y mkdir (para crear carpetas), que permiten manipular el entorno de manera más rápida y eficiente.

**23.** **Fundamentos de Desarrollo de Software**  
El desarrollo de software involucra el proceso de crear programas o aplicaciones mediante la programación. Este proceso generalmente incluye la planificación del proyecto, el diseño de la arquitectura del software, la codificación del programa, y la posterior prueba y mantenimiento. Es crucial para asegurar que el software funcione correctamente y cumpla con las necesidades del usuario o del negocio.

**24.** **Ciclo de vida del software**  
El ciclo de vida del software describe las fases que atraviesa un programa desde su concepción hasta su retirada. En general, incluye planificación, análisis de requisitos, diseño, desarrollo, pruebas, despliegue y mantenimiento. Un enfoque estructurado durante este ciclo garantiza que el software sea de alta calidad y que pueda evolucionar con el tiempo para cumplir con las nuevas necesidades de los usuarios.

**25.** **Requisitos**  
Los requisitos son las necesidades o expectativas que un sistema o software debe cumplir. Pueden ser funcionales (lo que el software debe hacer) o no funcionales (como rendimiento o seguridad). Es crucial entender los requisitos del usuario desde el inicio para diseñar una solución que satisfaga sus necesidades y expectativas, evitando cambios costosos en etapas avanzadas del desarrollo.

**26.** **Prototipos**  
Un prototipo es una versión preliminar de un sistema, utilizada para visualizar y probar ideas antes de desarrollar el producto final. Los prototipos permiten a los desarrolladores obtener retroalimentación temprana de los usuarios, identificar problemas o áreas de mejora, y ajustar el diseño o funcionalidades antes de la implementación completa.

**27.** **Interfaz de usuario**  
La interfaz de usuario (UI) es el punto de interacción entre el usuario y el software. Su diseño incluye la disposición de los elementos gráficos, botones, menús y otros componentes que facilitan el uso del sistema. Una buena interfaz debe ser intuitiva y fácil de navegar, proporcionando una experiencia positiva al usuario y facilitando la interacción con el software.

**28.** **Pruebas**  
Las pruebas son una parte esencial del desarrollo de software, ya que garantizan que el programa funcione correctamente y cumpla con los requisitos especificados. Durante las pruebas, se identifican errores (bugs) y se verifica la fiabilidad, rendimiento y seguridad del software. Las pruebas pueden incluir pruebas unitarias, pruebas de integración, y pruebas de aceptación de usuario.

**29.** **¿Qué es una base de datos?**  
Una base de datos es un sistema organizado para almacenar, gestionar y recuperar información de manera eficiente. Las bases de datos se estructuran en tablas que contienen filas y columnas, donde se almacenan los datos. Utilizan sistemas de gestión como MySQL, PostgreSQL o MongoDB para realizar operaciones de consulta, inserción, actualización y eliminación de datos.

**30.** **Internet**  
Internet es una red global de computadoras interconectadas que permite el intercambio de información. Funciona a través de protocolos como TCP/IP, que definen cómo se envían y reciben los datos a través de la red. Internet ha transformado la forma en que nos comunicamos, aprendemos, compramos y trabajamos, haciendo posible la accesibilidad instantánea a la información desde cualquier lugar.

**31.** **Direcciones IP**  
Las direcciones IP (Protocolo de Internet) son identificadores únicos asignados a cada dispositivo conectado a una red. Estas direcciones permiten que los dispositivos se localicen y se comuniquen entre sí en Internet o en redes locales. Las direcciones IP pueden ser estáticas (fijas) o dinámicas (cambian con el tiempo), y existen dos versiones principales: IPv4 e IPv6.

**32.** **Navegadores**  
Un navegador es un software que permite acceder a sitios web en Internet. Los navegadores interpretan el código HTML, CSS y JavaScript de las páginas web y presentan la información de manera visual y funcional. Algunos navegadores populares son Google Chrome, Mozilla Firefox, Safari y Microsoft Edge. Los navegadores también permiten interactuar con aplicaciones web y acceder a recursos en línea.

**33.** **Cliente y servidor**  
En la arquitectura cliente-servidor, el cliente es un dispositivo o programa que solicita servicios o recursos, mientras que el servidor es el sistema que proporciona esos servicios o recursos. Los servidores pueden almacenar datos, ejecutar aplicaciones o gestionar redes, mientras que los clientes acceden a estos servicios para utilizarlos, como cuando un navegador solicita información a un servidor web.

**34.** **Seguridad inicial**  
La seguridad en tecnología se refiere a proteger los sistemas y datos de accesos no autorizados, daños o alteraciones. El uso de contraseñas seguras es una de las primeras medidas para proteger la privacidad. Es importante elegir contraseñas complejas, evitar compartirlas y usar autenticación en dos pasos para mejorar la seguridad. También se deben tomar precauciones contra malware y ataques cibernéticos.

**35.** **HTML**  
HTML (HyperText Markup Language) es el lenguaje estándar utilizado para crear páginas web. Define la estructura de un documento web mediante etiquetas que indican el contenido, como encabezados, párrafos, enlaces y tablas. Aunque HTML solo describe la estructura, es fundamental para crear contenido web que luego puede ser estilizado y interactuado mediante CSS y JavaScript.

**36.** **CSS**  
CSS (Cascading Style Sheets) es un lenguaje utilizado para describir la presentación de un documento HTML, incluyendo su diseño, colores, fuentes y disposición de los elementos. Permite separar la estructura de la página (HTML) de su apariencia visual, lo que facilita su mantenimiento y la mejora de la experiencia del usuario. Un diseño atractivo y organizado depende en gran medida de CSS.

**37.** **JavaScript introductorio**  
JavaScript es un lenguaje de programación utilizado para crear interactividad en las páginas web. Permite realizar acciones como mostrar alertas, cambiar el contenido de la página sin recargarla, o responder a la interacción del usuario. Con JavaScript, las páginas web se vuelven dinámicas y ofrecen una experiencia más atractiva y funcional.

**38.** **Páginas estáticas**  
Las páginas estáticas son aquellas cuyo contenido no cambia a menos que se edite directamente el archivo. Están creadas principalmente con HTML y CSS, y son ideales para mostrar información que no necesita ser actualizada frecuentemente, como portafolios, blogs o sitios de negocios pequeños. Son simples y rápidas de cargar, pero carecen de interactividad avanzada.

**39.** **Hosting básico**  
El hosting es el servicio que permite almacenar y publicar páginas web en un servidor para que sean accesibles en Internet. Para subir una página web, se necesita contratar un servicio de hosting que proporcione espacio en un servidor. Al subir archivos HTML, CSS y otros necesarios, la página se vuelve accesible desde cualquier navegador web a través de una URL.

**40.** **Editores de código**  
Los editores de código son herramientas que facilitan la escritura y edición de programas informáticos. Un editor de código popular es Visual Studio Code (VS Code), que ofrece funcionalidades como resaltado de sintaxis, autocompletado y depuración, entre otras. Estos editores son esenciales para hacer más eficiente el desarrollo de software y permiten a los programadores escribir código limpio y fácil de entender.

**41. Control de versiones - Qué es Git (concepto inicial)**:  
Git es un sistema de control de versiones que permite llevar un registro de los cambios realizados en el código fuente. Facilita la colaboración entre varios desarrolladores, permite deshacer cambios y gestionar versiones de proyectos a lo largo del tiempo.

**42. Repositorios - Idea de GitHub**:  
GitHub es una plataforma de hospedaje de repositorios basada en Git. Permite a los desarrolladores almacenar y gestionar su código, así como colaborar con otros usuarios a través de "pull requests" y otras funcionalidades.

**43.**  **Línea de comandos - Comandos útiles para programar**:  
La línea de comandos es una interfaz en la que los usuarios pueden escribir comandos para interactuar con el sistema operativo. Comandos como git clone, git commit, python, entre otros, son fundamentales para el desarrollo y la ejecución de programas.

**44. Entornos de desarrollo - Instalación de Python o similar**:  
Un entorno de desarrollo es un conjunto de herramientas que facilitan la programación, como editores de texto, depuradores y compiladores. Python, por ejemplo, es un lenguaje popular, y para desarrollarlo necesitas instalar su intérprete y un editor adecuado.

**45. Metodología ágil - Idea de iteraciones cortas**:  
Las metodologías ágiles, como Scrum, se centran en el desarrollo incremental y la mejora continua. Se dividen proyectos grandes en pequeñas tareas (iteraciones) para adaptarse rápidamente a los cambios y entregar resultados en ciclos cortos.

**46. Documentación - Escribir cómo funciona el código**:  
La documentación es el proceso de escribir explicaciones claras y comprensibles sobre el funcionamiento del código. Es esencial para que otros desarrolladores puedan entender, mantener y modificar el código en el futuro.

**47. Resolución de problemas - Dividir en partes**:  
La resolución de problemas es un enfoque sistemático para encontrar soluciones. Dividir un problema grande en partes más pequeñas hace que sea más fácil de manejar y resolver paso a paso.

**48. Comunicación - Explicar ideas técnicas**:  
La capacidad de comunicar de manera efectiva ideas técnicas es esencial en el desarrollo de software. Esto incluye escribir documentación clara y ser capaz de presentar y explicar conceptos técnicos a otros, ya sea en equipo o en presentaciones.

**49. Pensamiento crítico - Evaluar soluciones**:  
El pensamiento crítico permite analizar y evaluar diferentes enfoques o soluciones a un problema. Implica considerar los pros y los contras, y tomar decisiones basadas en la lógica y la evidencia.

**50. Ética en TI - Uso responsable de la tecnología**:  
La ética en TI se refiere al uso responsable y justo de la tecnología. Esto incluye respetar la privacidad, evitar el uso malintencionado de las herramientas tecnológicas y asegurarse de que las soluciones tecnológicas beneficien a la sociedad.

**51. Privacidad - Proteger datos básicos**:  
La privacidad es fundamental para proteger los datos personales y sensibles de los usuarios. Las buenas prácticas incluyen el uso de cifrado, la protección contra accesos no autorizados y la transparencia en el uso de los datos.

**52. Persistencia - Lidiar con errores y fracasos**:  
La persistencia es la capacidad de seguir adelante a pesar de los errores y fracasos. Es importante en el desarrollo de software, donde los problemas surgen con frecuencia, y la habilidad para aprender de ellos es clave.

**53. Proyecto simple - Calculadora o lista de tareas**:  
Empezar con proyectos simples, como una calculadora o una lista de tareas, es una excelente manera de practicar las habilidades de programación y aplicar los conceptos básicos.

**54. Reutilización de código - Usar funciones ya hechas**:  
Reutilizar código existente ahorra tiempo y esfuerzo. Utilizar funciones, bibliotecas o módulos que ya están probados y documentados mejora la eficiencia y reduce la posibilidad de errores.

**55. Inteligencia artificial - Qué es en términos simples**:  
La inteligencia artificial (IA) es una rama de la informática que permite a las máquinas realizar tareas que normalmente requieren inteligencia humana, como el aprendizaje, la toma de decisiones y el procesamiento del lenguaje natural.

**56. Tipos de archivos: doc, png**:  
Los tipos de archivos definen el formato y la estructura de los datos. Por ejemplo, un archivo .doc es un documento de texto de Microsoft Word, mientras que .png es un formato de imagen sin pérdida de calidad.

**57. Aplicaciones móviles - Ejemplo de su uso**:  
Las aplicaciones móviles son programas diseñados para ejecutarse en dispositivos móviles como smartphones y tabletas. Ejemplos incluyen aplicaciones de redes sociales, mensajería, banca y entretenimiento.

**58. Videojuegos - Introducción al desarrollo básico**:  
El desarrollo de videojuegos involucra la creación de software interactivo para entretenimiento. Comienza con conceptos básicos de programación y diseño gráfico, y puede involucrar programación de inteligencia artificial, gráficos 3D y sonido.

**59. Impacto del software - Cómo cambia el mundo**:  
El software tiene un impacto profundo en la sociedad, desde la automatización de tareas hasta la creación de nuevas industrias. Ha transformado la comunicación, la educación, la salud, la economía y muchos otros aspectos de la vida cotidiana.

**60. Aprendizaje continuo - Importancia de seguir estudiando**:  
La tecnología avanza rápidamente, por lo que es esencial mantener un aprendizaje continuo. La capacitación y la actualización de conocimientos permiten a los profesionales mantenerse competitivos y relevantes en el campo de TI.