**CONCEPTUALIZACIÓN SOBRE ALGORITMOS Y PROGRAMACIÓN**

**Presenta: *Juan Pablo Florez Rosario***

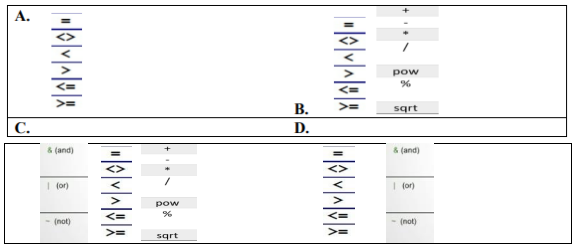
1. El en el contexto de programación las variables cumplen un papel muy importante para el desarrollo del algoritmo, dadas las siguientes definiciones cuál considera usted que es la apropiada para una variable.
2. Es un dato que cambia su valor durante la ejecución del programa, esta debe tener definido un tipo de dato y un nombre.
3. Es un espacio en memoria que guarda cualquier tipo de dato de forma implícita.
4. Es un dato que guarda valores de salida y puede retornar diferentes tipos de valores.
5. Ninguna de las anteriores.
6. Del siguiente contexto identifique la cantidad de variables y su tipo que son necesarias para el desarrollo del algoritmo.

Contexto: Desarrolle un algoritmo que permita leer tres valores y determine cuál de esos valores es el mayor y cuál es el menor, tenga en cuenta que el programa debe validar que los valores proporcionados deben ser diferentes, en caso contrario imprimir un mensaje de alerta donde se indique que debe ingresar valores diferentes.

1. 5, 3 de tipo entero y 2 de tipo entero que guardan el mayor y el menor.
2. 3 de tipo entero
3. 6 porque debe enviar mensaje
4. 3 de cualquier valor

**Este punto está adjuntado como un bloc de notas**

1. Existen tipos de datos que pueden ser utilizados en la programación, entre estos tenemos aritméticos, relacionales y lógicos, identifique cuál de los siguientes conjuntos los representa de forma acertada.

****

1. Relacionales
2. Aritméticos
3. Lógico-Aritméticos
4. Lógicos
5. Es importante en programación conocer la prioridad de los tipos de operadores, dadas las opciones cual considera que es la apropiada.
6. Primero sumas y restas, posteriormente multiplicaciones y divisiones al final paréntesis y corchetes.
7. Paréntesis anidados, potencias, multiplicaciones y divisiones de izquierda a derecha, al igual que sumas y restas.
8. De izquierda a derecha, teniendo en cuenta que los paréntesis deben resolverse primero.
9. Ninguna de las anteriores.
10. Dada la siguiente expresión resolverla −2[−3(−1+(−2))^2]−1(2+(−5))^3

Comenzamos con los paréntesis más internos:

1. −2[−3(−1+(-2))^2]−1(2+(-5))^3

= −2[−3(−1-2)^2]−1(2-5)^3

= −2[−3(−3)^2]−1(−3)^3

1. A continuación, evaluamos las potencias:

= −2[−3(9)]−1(−27)

= −2[−27]−1(−27)

1. Luego, realizamos las multiplicaciones y divisiones de izquierda a derecha:

= −2[-27] - 1(-27)

= 54 +27

1. Por último, realizamos las sumas y restas:

= 81

1. Una forma de representar un algoritmo es un diagrama de flujo, diseñe un diagrama de flujo para el siguiente problema.

**Problemática:** Desarrollar un algoritmo que determine si un número es par o impar

**Este punto está adjuntado como un bloc de notas**

1. Dados los siguientes tipos de datos especifique que significa cada tipo:

En programación, el tipo de dato "int" se utiliza para representar valores numéricos enteros en un rango específico. El rango exacto de valores que se pueden representar depende del lenguaje de programación utilizado y de la arquitectura del sistema en el que se ejecuta el programa. En general, los enteros suelen ocupar una cantidad fija de memoria, lo que permite realizar operaciones matemáticas eficientes con ellos.

En C++:

int edad = 25;

El tipo de dato "string" (cadena de caracteres) se utiliza para representar texto en programación. Una cadena de caracteres es una secuencia de caracteres, como letras, números, símbolos y espacios en blanco, que se pueden combinar para formar palabras, frases, párrafos y cualquier otro tipo de texto.

En la mayoría de los lenguajes de programación, una cadena de caracteres se representa entre comillas dobles ("") o comillas simples (''). Por ejemplo:

En C++:

std::string nombre = "Juan";

El tipo de dato "bool" se refiere a los valores booleanos, que representan la idea de verdadero o falso. El tipo de dato "bool" es una abreviatura de "boolean" en inglés.

En la mayoría de los lenguajes de programación, el tipo de dato "bool" solo puede tener dos posibles valores: "true" (verdadero) o "false" (falso). Estos valores booleanos se utilizan para tomar decisiones y controlar el flujo de un programa.

El tipo de dato "char" se utiliza para representar caracteres individuales en programación. "Char" es una abreviatura de "character" en inglés.

En la mayoría de los lenguajes de programación, un valor de tipo "char" representa un solo carácter y se define entre comillas simples (''). Por ejemplo:

En C++:

char letra = 'A';

El tipo de dato "float" se utiliza para representar números de punto flotante, es decir, números con parte decimal en programación. "Float" es una abreviatura de "floating-point" en inglés.

En la mayoría de los lenguajes de programación, el tipo de dato "float" se utiliza para representar números reales y se declara utilizando la palabra clave correspondiente. Por ejemplo:

En C++:

float numero = 3.14;

1. Un algoritmo tiene unas etapas de desarrollo, menciones dichas etapas.
2. Análisis del problema: En esta etapa, se comprende completamente el problema que se desea resolver. Se definen los requisitos y se identifican las restricciones o limitaciones del problema.
3. Diseño del algoritmo: En esta etapa, se desarrolla una solución de alto nivel para el problema identificado. Se selecciona una estructura de datos apropiada y se determina la lógica del algoritmo. Esta etapa puede involucrar la creación de diagramas de flujo u otros modelos para representar el algoritmo.
4. Codificación: En esta etapa, se traduce el diseño del algoritmo en un lenguaje de programación específico. Se escriben las instrucciones detalladas que permitirán implementar el algoritmo en un entorno computacional.
5. Depuración (debugging): Una vez que el algoritmo se ha codificado, se lleva a cabo la depuración para identificar y corregir cualquier error o defecto en el código. Se prueban diferentes casos y se realiza un seguimiento para asegurarse de que el algoritmo funcione correctamente en diferentes situaciones.
6. Pruebas y evaluación: En esta etapa, se realizan pruebas exhaustivas del algoritmo para verificar su funcionamiento y evaluar su desempeño. Se utilizan diferentes conjuntos de datos de prueba para verificar la precisión y eficiencia del algoritmo. Si es necesario, se realizan ajustes o mejoras en el algoritmo.
7. Implementación: Una vez que el algoritmo ha sido probado y evaluado satisfactoriamente, se implementa en un entorno operativo.
8. Resuelva el ejercicio utilizando código C++. Realice un programa que calcule la nota de un estudiante, el estudiante durante el semestre presenta tres exámenes con el mismo ponderado. El programa debe imprimir la nota junto con un mensaje de aprobado (Si el promedio de las 3 notas es mayor o igual a 3.5) o no aprobado (Si el promedio de las 3 notas es menor a 3.5). A continuación, organice la estructura de la solución propuesta.

**Este punto está adjuntado como un bloc de notas**

1. Proponga una problemática que usted desee solucionar mediante un algoritmo.

un algoritmo que ayude a la gestión eficiente de inventario en empresas y cadenas de suministro. El manejo inadecuado del inventario puede llevar a problemas como la falta de productos en stock, exceso de inventario, obsolescencia, altos costos de almacenamiento y pérdida de ventas. Un algoritmo inteligente podría analizar datos históricos de ventas, pronósticos de demanda, plazos de entrega de proveedores y otros factores relevantes para optimizar la gestión del inventario. Algunas posibles soluciones podrían incluir:

1. Pronóstico de demanda: el algoritmo podría utilizar técnicas de análisis predictivo para predecir la demanda futura de cada producto. Esto ayudaría a determinar los niveles de inventario adecuados y evitar la falta o el exceso de stock.
2. Reabastecimiento automatizado: el algoritmo podría generar órdenes de compra automáticas cuando los niveles de inventario alcancen ciertos umbrales predefinidos. Esto aseguraría que los productos se reabastezcan a tiempo, evitando la falta de stock y reduciendo los costos de almacenamiento.
3. Optimización de la distribución: el algoritmo podría analizar la ubicación geográfica de los clientes, los puntos de distribución y los tiempos de entrega para optimizar la distribución de productos. Esto reduciría los costos de transporte y mejoraría la eficiencia en la entrega.
4. Gestión de obsolescencia: el algoritmo podría identificar productos obsoletos o de bajo rendimiento y recomendar acciones como promociones, liquidaciones o ajustes en la producción para minimizar las pérdidas asociadas.

Estas son solo algunas ideas iniciales, pero un algoritmo de gestión de inventario eficiente podría ayudar a las empresas a reducir costos, mejorar el servicio al cliente y maximizar las ganancias al mantener los niveles de inventario óptimos en todo momento.