

UNIDAD 1

CONCEPTOS BÁSICOS.

SEMINARIO DE ENCUENTRO NÚMERO 1.

PRESEMINARIO

Con el fin de apropiarse de los conceptos básicos introductorios al curso, el estudiante debe realizar la siguiente lectura y complementarla con la bibliografía, o los enlaces sugeridos, buscando temas relacionados con la unidad en estudio.

I. PANORAMA GENERAL DE UN SISTEMA DE CÓMPUTO.

Los sistemas de cómputo se presentan en diferentes formas y tamaños dependiendo de su aplicación; pero independiente de esto, todos se caracterizan por estar compuestos de cuatro partes:

1. HARDWARE.

El término Hardware se refiere a todos los componentes

Anotaciones

físicos del sistema de cómputo, está compuesto por dispositivos electrónicos interconectados que se clasifican en cuatro grupos así:

1.1. PERIFÉRICOS DE ENTRADA: Los computadores necesitan recibir datos e instrucciones para resolver cualquier problema. Los periféricos o dispositivos de entrada consisten en uno o más elementos de hardware que permiten el ingreso de datos al sistema. El teclado y el “mouse” (ratón) son los dispositivos de entrada más utilizados

1.2. UNIDAD DEL SISTEMA: Conjunto de dispositivos electrónicos para el procesamiento de datos. Es lo que erróneamente muchas personas denominan CPU. Se dice erróneamente, porque la CPU es un dispositivo interno de esta unidad. Comúnmente en el lenguaje popular se conoce como la torre. Ésta se compone de:

1.2.1. Fuente de Poder. Es la encargada de transformar la corriente eléctrica alterna en continua, suministrando la corriente imprescindible y necesaria para funcionamiento y protección de los diversos dispositivos

1.2.2. Cables: Se clasifican en: cables de poder, de corriente y de datos.

1.2.3. Tarjetas Auxiliares. Algunas son: De video, de sonido, aceleradoras, en otras

1.2.4. Main Board o Tarjeta Principal: En ella se encuentran diversos elementos y algunos de los más importantes:

- **Procesador o CPU:** (Central Processing Unit) Unidad de proceso central. Es considerado el cerebro del computador, es el encargado de interpretar las instrucciones en un programa, y las ejecuta una por una. Consiste de tres unidades principales:

- Unidad de Control: controla y dirige la transferencia de instrucciones y datos entre varias unidades.
- Unidad Aritmética/Lógica: Realiza operaciones

Anotaciones

	Anotaciones
<p>aritméticas (suma +, resta -, multiplicación *, exponenciación ^ y división /), lógicas (AND, OR, NOT) y relacionales (menor que <, mayor que >, menor igual <=, mayor igual >=, igual = y diferente ≠).</p> <ul style="list-style-type: none"> – Registros: Utilizados para almacenar de manera temporal instrucciones y datos para su uso por el procesador. • Memoria Principal: La CPU utiliza la memoria del computador para guardar información mientras trabaja con ella; mientras esta información permanezca en memoria, el computador puede tener acceso a ella en forma directa. Esta memoria construida internamente se llama memoria de acceso aleatorio (RAM). <p>La memoria principal consta de dos áreas de memoria:</p> <ul style="list-style-type: none"> – Memoria RAM (Random Access Memory). Memoria de acceso aleatorio. En ella se almacena información solo mientras el computador está encendido. Cuando se apaga, la información se pierde, por lo que se dice que la memoria RAM es una memoria volátil. – Memoria ROM (Read Only Memory). Memoria de solo lectura. Es una memoria estática que no puede cambiar; el computador puede leer los datos almacenados en ella, pero no introducir información o cambiar los datos que ahí se encuentran, por lo que se dice que esta memoria es de solo lectura. Los datos de la memoria ROM están grabados en forma permanente y son introducidos por el fabricante de la computadora. Pese a lo anterior, existen nuevas tecnologías que permiten la programación, borrado o modificación de datos por parte del usuario, tales como la PROM¹, EPROM² y EEPROM³. • BIOS. Responde a las siglas de “Basic Input/Output System”. Un sistema básico de entrada y salida, que localiza y reconoce los elementos necesarios para cargar la memoria RAM. En otras palabras, es una combinación 	

¹ PROM Program+nable Read-Only Memory (ROM programmable)

² EPROM Erasable Programmable Read-Only Memory (ROM programable borrrable).

³ Electrically - Erasable Programmable Read - Only Memory (ROM programable y borrrable eléctricamente).

de *hardware* y *software* (*firmware*) encargada de Iniciar y probar el *hardware*, dando además las primeras instrucciones a la máquina y cargando un gestor de arranque o inicio. Este *software* opera de forma autónoma, como nuestra respiración, y proporciona una serie de datos informativos para saber si el *hardware* del sistema está configurado correctamente. Su relación directa con el *hardware* y con los controladores de las tarjetas de audio, vídeo, discos rígidos y periféricos hace de la BIOS un elemento clave para el buen funcionamiento del sistema.

- **Chip-Set.** El chipset sirve de centro de información de la *main board*. Por éste, pasa una gran cantidad de los datos que se procesan en el computador.

Dicho de otra forma, la información dentro del PC se envía a través de buses o canales de comunicación (los tres principales buses son el USB, PCI Express y el SATA) y el chip-set hace la administración del flujo de dicha información.

1.2.5. Unidades de Memoria: Utilizadas para almacenar programas y datos de forma permanente. En la jerarquía de memorias se consideran de tipo secundario y su clasificación principal obedece a:

- Memorias magnéticas
- Memoria óptica
- Memorias magneto-ópticas
- Memorias flash

1.3. PERIFÉRICOS DE SALIDA: Son los encargados de entregar la información procesada al usuario o desplegar los resultados que están almacenados en la unidad de memoria. Los más utilizados son el monitor y la impresora.

2. SOFTWARE.

El término *software* se refiere al conjunto de instrucciones electrónicas que le dicen al *hardware* que debe hacer y que a la vez permite aprovechar todos los recursos que el computador tiene, de manera que pueda resolver gran cantidad de problemas. Un computador en

Anotaciones

Anotaciones

si, es sólo un conjunto de elementos electrónicos que por si solos no cumplen ninguna función, es decir el computador es una máquina bruta; el software le da vida a éste, haciendo que sus componentes funcionen de forma ordenada y lo conviertan en la máquina más sorprendente que el hombre haya podido crear.

Este conjunto de instrucciones se conoce como programas y cada uno tiene un fin específico. Se puede decir que las funciones del software son:

- Administrar los recursos de cómputo.
- Proporcionar las herramientas para optimizar estos recursos.
- Actuar como intermediario entre el usuario y la información almacenada

2.1. TIPOS DE SOFTWARE.

2.1.1. Software del sistema. Es un conjunto de programas que administran los recursos del computador tales como la unidad central de proceso, los dispositivos de comunicaciones y dispositivos periféricos entre otros; el software del sistema administra y controla el acceso al hardware.

2.1.2. Software Utilitario: Conocido también como software de aplicaciones; son Programas que son escritos para o por los usuarios y que realizan una tarea específica. Ejemplo: software para procesar un texto, para generar una hoja de cálculo, el software utilitario debe estar sobre el software del sistema para poder operar.

2.1.3. Lenguajes de Programación: Es un conjunto de símbolos, caracteres y reglas (programas) que les permiten a las personas comunicarse con el computador, diseñando software a la medida del usuario.

Los lenguajes de programación tienen un conjunto de instrucciones que nos permiten realizar operaciones de entrada/salida, calculo, manipulación de textos, lógica/comparación y almacenamiento/recuperación.

Los lenguajes de programación se clasifican en:

Anotaciones

- **Lenguajes Máquina:** Son aquellos cuyas instrucciones son directamente entendibles por el computador y no necesitan traducción posterior para que la CPU pueda comprender y ejecutar el programa. Las instrucciones en lenguaje maquina se expresan en términos de la unidad de memoria mas pequeña el bit (dígito binario 0 o 1).

- **Lenguaje de Bajo Nivel (Ensamblador):** En este lenguaje las instrucciones se escriben en códigos alfabéticos conocidos como mnemotécnicos para las operaciones y direcciones simbólicas.

- **Lenguaje de Alto Nivel:** Los lenguajes de programación de alto nivel (BASIC, pascal, cobol, fortran, etc.) son aquellos en los que las instrucciones o sentencias al computador son escritas con palabras similares a los lenguajes humanos (en general en ingles), lo que facilita la escritura y comprensión del programa.

- **Lenguajes Algorítmicos:**

Son una serie de símbolos y reglas que se utilizan para describir de manera explícita un proceso para la resolución de un problema.

Básicamente existen dos tipos:

- **Gráficos:** Es la representación gráfica de las operaciones que realiza un algoritmo (diagrama de flujo, diagramas N.S).
- **No Gráficos:** Representan en forma descriptiva las operaciones que debe realizar un algoritmo (pseudocódigo).

3. DATOS.

Los datos se refieren a los elementos crudos que el computador puede manipular. Pueden consistir en letras números, sonidos, imágenes; pero sin importa que clase de datos son introducidos al computador, estos siempre serán convertidos en números. Lo anterior indica que los datos computarizados son digitales. Ya en el computador los datos son organizados en archivos.

Anotaciones

4. USUARIOS.

La última parte de un sistema de cómputo son las personas que hacen uso de éste.

II. REPRESENTACIÓN INTERNA DE DATOS

1. UNIDADES DE INFORMACION

Bits y Bytes: Los datos en el computador se representan utilizando únicamente '0' y '1'. Estos son llamados BITS ("Binary digiT'S"), por consiguiente, un "BIT" es la unidad mínima de información que se puede representar. Un conjunto de 8 bits se denomina Byte (octeto en español), y un Byte almacena un caracter. Con n bits, pueden representarse 2^n .

Ejemplo

Con dos bits podemos representar 4 cadenas:
00, 01, 10, 11.

El código ASCII (American Standards Code for Information Interchange) se utiliza para representar cada caracter. El código ASCII incluye códigos para las letras del alfabeto (minúsculas y mayúsculas), los dígitos decimales, 32 caracteres especiales y otros códigos para los llamados caracteres de control.

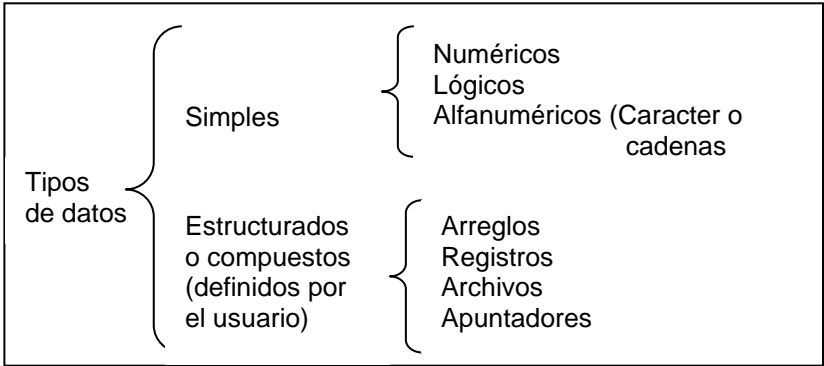
2. TIPOS DE DATOS

Todos los datos tienen un tipo asociado con ellos. Un dato puede ser un simple carácter, tal como 'b', un valor entero tal como 35, o una serie de datos ya definidos y relacionados o agrupados. El tipo de dato determina la naturaleza del conjunto de valores que puede tomar una

variable. La clasificación básica de los tipos de datos es:

simple y estructurados, pero ésta en detalle se puede observar en la figura 3.

Figura 3. Clasificaciones básicas de los datos



Los tipos de datos simples se clasifican en:

2.1. DATOS NUMÉRICOS. Permiten representar valores escalares de forma numérica, esto incluye a los números enteros y los reales. Este tipo de datos permiten realizar operaciones aritméticas comunes.

2.2. DATOS LÓGICOS. Son aquellos que solo pueden tener dos valores (cierto o falso) ya que representan el resultado de una comparación entre otros datos (numéricos o alfanuméricos).

2.3. DATOS ALFANUMÉRICOS. (caracteres o cadenas). Obedecen a un carácter o una secuencia de caracteres alfanuméricos que permiten representar valores identificables de forma descriptiva, esto incluye nombres de personas, direcciones, una simple letra o símbolo, etc.

Es posible representar números como alfanuméricos, pero estos pierden su propiedad matemática, es decir no es posible hacer operaciones con ellos. Este tipo de datos se representan encerrados entre comillas.

Ejemplo:

"A"	un dato caracter
"Calle 10 # 20 - 50"	un dato cadena
"2000"	un dato cadena

Anotaciones

Anotaciones

Anotaciones

Muchos autores hablan de los datos alfanuméricos como uno solo, es decir no los clasifican en caracteres y cadenas, llamándolos simplemente string, pero es importante que se establezca esta diferencia por la representación que estos deben de tener en el computador.

Los tipos de datos estructurados o compuestos no se analizan en este libro, pero para tener una visión clara de estos, en el seminario central se presenta su mapa conceptual.

3. IDENTIFICADORES.

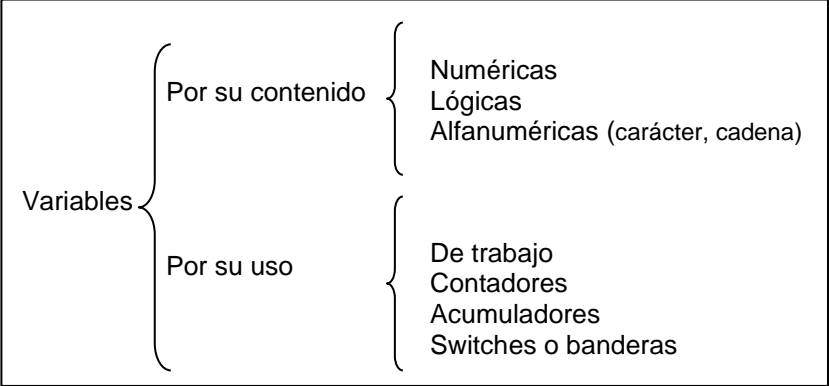
Para representar los datos de un programa en la memoria del computador se utilizan los identificadores (constantes, y variables). Un identificador es una secuencia de caracteres que sirve para identificar una posición en la memoria del computador, y que nos permite acceder a su contenido. Para poder reconocer un identificador en la memoria del computador, es necesario darle un nombre con el cual podamos identificarlo dentro de un algoritmo.

Ejemplo: Nombre
 Num_hrs
 Calif2

Reglas para formar un Identificador

- Debe comenzar con una letra (A a Z, mayúsculas o minúsculas).
- no deben contener espacios en blanco.
- Letras, dígitos y caracteres como la subraya (_) están permitidos después del primer carácter.
- Su nombre debe ser significativo a lo que representa

3.1. CONSTANTE. Es un espacio en la memoria del computador que corresponde a un dato numérico o alfanumérico que no cambia durante la ejecución del programa.

Anotaciones	
Anotaciones	
<p>Ejemplo: Pi = 3.1416</p>	
<p>3.2. VARIABLE. Es un espacio en la memoria del computador que permite almacenar temporalmente un dato durante la ejecución de un proceso, su contenido puede cambia durante la ejecución del programa.</p>	
<p>Ejemplo: Area = PI * radio ^ 2 Las variables son: radio, Area y la constate es PI</p>	
<p>3.2.1. Clasificación de las Variables. En la figura 4 se observa la clasificación básica de las variables, la cual se describe a continuación:</p>	
<p>Figura 4. Clasificación básica de las variables</p>  <pre> graph LR V[Variables] --- C1{ } C1 --- PC[Por su contenido] C1 --- PU[Por su uso] PC --- N[Numéricas] PC --- L[Lógicas] PC --- A[Alfanuméricas (carácter, cadena)] PU --- DT[De trabajo] PU --- C[Contadores] PU --- AC[Acumuladores] PU --- SB[Switches o banderas] </pre>	
<ul style="list-style-type: none"> • Por su Contenido: Numéricas Lógicas Alfanuméricas (carácter o cadena) 	
<ul style="list-style-type: none"> – Variable Numéricas: Son aquellas en las cuales se almacenan valores numéricos, positivos o negativos, es decir almacenan números del 0 al 9, signos (+ y -) 	

y el punto decimal.

Ejemplo:

lva = 0.15 pi = 3.1416 costo = 2500

Variables Lógicas: Son aquellas que solo pueden tener dos valores (Verdadero o falso) estos representan el resultado de una comparación entre otros datos

- Variables Alfanuméricas: Corresponden a aquellas cuyo contenido está formado por un caracter alfanumérico (letra, número o caracteres especiales) y recibe el nombre de variable carácter; o por un conjunto de caracteres y recibe el nombre de variable cadena.

Ejemplo:

Letra = "a"	Carácter
Apellido = "González"	Cadena
direccion = "Carrera 48 # 7-110"	Cadena

- **Por su Uso:** De Trabajo,
Contadores
Acumuladores
Switches o banderas
- Variables de Trabajo: Variables que reciben el resultado de una operación matemática completa y que se usan normalmente dentro de un programa.

Ejemplo: suma=a+b/c

- Contadores: Se utilizan para llevar el control del número de ocasiones en que se realiza una operación o se cumple una condición. Con los incrementos generalmente de uno en uno.

Contador = Contador + <incremento>

- Acumuladores: Forma que toma una variable y que

Anotaciones

Anotaciones

sirve para llevar la suma acumulativa de una serie de valores que se van leyendo o calculando

progresivamente

Acumulador = Acumulador + <variable a acumular>

- Switches o banderas. Una variable tipo switch o bandera es aquella que almacena generalmente uno de dos valores excluyentes entre sí. Los valores los define el programador y son normalmente 0 y 1, S' y 'N' 'F' y 'V' "SI" y "NO" "VERDADERO" y "FALSO"

Los switches o banderas se utilizan para:

- a. controlar un ciclo cuando no se puede controlar con otro tipo de variable.
- b. para indicar que ya se consiguió u ocurrió algo que se esperaba en el algoritmo.
- c. para alternar los caminos dentro de un ciclo.

4. EXPRESIONES.

Las expresiones son combinaciones de constantes, variables, valores, símbolos de operación, paréntesis y nombres de funciones especiales.

Ejemplo:

$$A=(b + 3)/c$$

Cada expresión toma un valor que se determina tomando los valores de las variables y constantes implicadas y la ejecución de las operaciones indicadas.

Una expresión consta de operandos y operadores. Según sea el tipo de datos que manipulan, se clasifican las expresiones en: Aritméticas, relacionales y lógicas; lo cual se da por resultado de los tipos de operadores implicados en la expresión

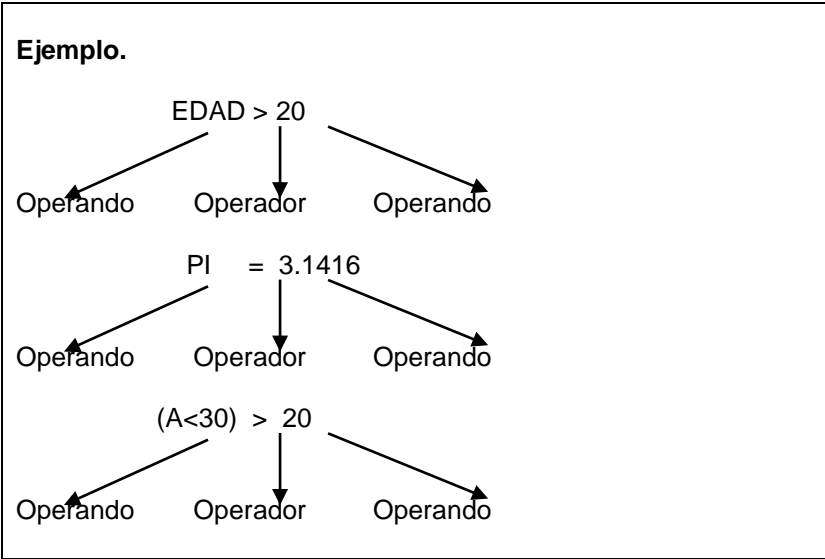
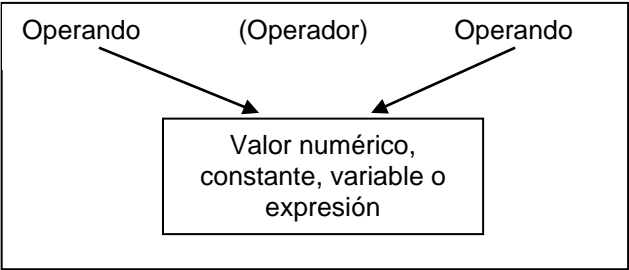
Anotaciones

Anotaciones

4.1. OPERANDOS y OPERADORES

4.1.1. Operandos. Obedecen a los diferentes elementos que están relacionados por intermedio de un operador, en

una expresión. Pueden ser valores, variables, constantes e incluso expresiones.



4.1.2. Operadores. Son elementos que relacionan de forma diferente, los valores de una o más variables y/o constantes. Es decir, los operadores nos permiten manipular valores.

Tipos de Operadores { Aritméticos
Relacionales
Lógicos

- **Operadores Aritméticos:** Los operadores aritméticos permiten la realización de operaciones matemáticas con

los valores, variables y constantes. Éstos pueden ser utilizados con tipos de datos enteros o reales. Si ambos son enteros, el resultado es entero; si alguno de ellos es real, el resultado es real.

– Tipos de operadores aritméticos

^	Exponenciación
+	Suma
-	Resta
*	Multiplicación
/	División
Mod	Módulo (residuo de la división entera)
Div	Cociente de la división

– Prioridad de los Operadores Aritméticos

a) Todas las expresiones entre paréntesis se evalúan primero. Las expresiones con paréntesis anidados se evalúan de adentro hacia afuera, el paréntesis más interno se evalúa primero.

b) Dentro de una misma expresión los operadores se evalúan en el siguiente orden.

1. ^ Exponenciación.
2. *, /, div, mod Multiplicación, división, cociente, módulo.
3. +, - Suma y resta.

c) Los operadores en una misma expresión con igual nivel de prioridad se evalúan de izquierda a derecha.

Ejemplo:

$$\begin{array}{rcl}
 4 + 2 * 6 & = & 16 \\
 4 + 10 & = & 16 \\
 14 & = & 16 \\
 \\
 23 * 2 / 5 & = & 9.2 \\
 46 / 5 & = & 9.2 \\
 9.2 & = & 9.2 \\
 \\
 3 + 5 * (10 - (2 + 4)) & = & 23
 \end{array}$$

Anotaciones

Anotaciones

Anotaciones

$3 + 5 * (10 - 6)$	$= 23$
$3 + 5 * 4$	$= 23$
$3 + 20$	$= 23$

• **Operadores Relacionales.** Se utilizan para establecer una relación entre dos valores, permitiendo comparar estos entre si, lo cual produce un resultado de certeza o falsedad (verdadero o falso).

Los operadores relacionales comparan valores del mismo tipo (numéricos o cadenas)

Tienen el mismo nivel de prioridad en su evaluación.

Los operadores relacionales tiene menor prioridad que los aritméticos.

– Tipos de operadores relacionales

>	Mayor que
<=	Menor o igual que
<	Menor que
< >	Diferente
>=	Mayor o igual que
=	Igual

Ejemplo:

Si $a = 20$ $b = 30$ $c = 40$

$a + b$	$>$	c	Falso
$a - b$	$<$	c	Verdadero
$a - b$	$=$	c	Falso
$a * b$	$< >$	c	Verdadero

• **Operadores Lógicos.** Estos operadores se utilizan para establecer relaciones entre valores lógicos y sus valores pueden ser resultado de una expresión relacional.

– Tipos de operadores lógicos

And Y
Or O
Not Negación

Para la evaluación de estos operadores se han establecido las tablas de verdad, las cuales definen el resultado de una operación (**R**), dependiendo de los valores de verdad de dos variables

Operador
AND

A	B	R
V	V	V
V	F	F
F	V	F
F	F	F

Operador
OR

A	B	R
V	V	V
V	F	V
F	V	V
F	F	F

Operador
NOT

A	R
V	F
F	V

- Prioridad de los operadores lógicos

Not
And
Or

Ejemplos:

Si $a = 15$, $b = 17$, $c = 30$

$(a < b)$ **And** $(b < c)$.

$(15 < 17)$ **And** $(17 < 30)$

V And V

.V.

• Prioridad de los Operadores en General

1. ()
2. ^
3. *, /, Mod, Not
4. +, -, And
5. >, <, >=, <=, <>, =, Or

Anotaciones

Anotaciones

4.2. EXPRESIONES ALGORITMICAS

Tradicionalmente las expresiones son representadas algebraicamente, es decir de la siguiente forma:

$$\frac{x + 2}{3x^2 - x^3}$$

A la hora de representar una expresión para ser utilizada por un computador, ésta se representa en forma lineal; razón por la cual la prioridad de los operadores juega un papel de vital importancia y su mal aplicación puede cambiar completamente el resultado de dicha expresión.

La anterior expresión expresada en forma algorítmica sería de la siguiente forma:

$$(X+2) / (3 * x ^ 2 - x ^ 3)$$

Ejemplo.

Para entender su resultado asumamos que X es igual a 2 y sigamos los pasos de solución en orden, así:

Primero se haya el numerador:

$$(x + 2) \rightarrow (2 + 2) \rightarrow 4$$

Para obtener el denominador se resuelven en orden las operaciones aplicando la regla de la prioridad así:

Primero exponenciaciones $X ^ 2 \rightarrow 2 ^ 2 \rightarrow 4$

$$X ^ 3 \rightarrow 2 ^ 3 \rightarrow 8$$

Segundo la multiplicación $3 * 4 \rightarrow 12$

Y por último la resta $12 - 8 \rightarrow 4$

Ya se obtuvo el denominador y se puede dar el resultado final así:

$$4 / 4 = 1$$

Cuando en una expresión algebraica se plantea una función específica, tal como el seno, coseno, raíz cuadrada, etc., ésta se representa en forma narrada, es decir se le asigna un nombre y éste corresponde a una función que será posteriormente definida o usada por el usuario. (el tema de funciones se ve con detalle en la

sección de programación modular.

Ejemplo. Dada la siguiente expresión algebraica obtener su equivalente en expresión algorítmica.

$$3x - \frac{x^2 + 2(x + x^3)}{\sqrt{3x - 2x^2}}$$

$3 * X - ((X ^ 2 * (X + X ^ 3)) / \text{RAIZCUA}(3 * X - 2 * X ^ 2))$

RAIZCUA equivale a una función que haya la raíz cuadrada de lo que este entre paréntesis

III PROGRAMACIÓN Y LENGUAJES DE PROGRAMACIÓN

ACTIVIDADES

Para la clase del día 15 de febrero consultar sobre los siguientes temas, realizar las siguientes actividades y subirlas al Classroom.

1. Paradigmas de la programación
 - Realizar un diagrama conceptual
2. Clasificación de los lenguajes de programación.
 - Realizar un diagrama conceptual
3. Historia de los lenguajes de programación C y C++
 - Realizar un diagrama con la línea de tiempo

Anotaciones