Programación Avanzada

Conceptos Básicos

Semestre 2019-1

Ing. Juan Carlos Sabido Alcántara

Ingeniero Petrolero

Facultad de Ingeniería UNAM


4G
5:15
777-777-7777
CALLING

Conceptos Básicos

- **Características de un programa.**

1. ***Debe de funcionar:*** La característica más importante de un programa es que funcione.
2. ***No debe de tener dificultades:*** Hay que anticipar las situaciones en las que se va a emplear el programa para evitar errores, esto puede ocurrir cuando el usuario introduce información errónea al programa, para esto el programador deberá asegurarse de obligar al usuario a utilizar solo información correcta y coherente con las capacidades y funciones del programa.

Conceptos Básicos

- **Características de un programa.**

3. ***Debe ser bien documentado:*** La documentación es necesaria para ayudar a comprender o a utilizar un programa. Puede realizarse de dos formas:
   - Externa: Lo que comúnmente se conoce como “Manual de referencia”, permite aprender a utilizar el programa.
   - Interna: La que se incluye en muchas aplicaciones como “Ayuda”, permite resolver dudas sobre le funcionamiento del programa.

Para trabajos de mantenimiento y actualización es conveniente tener documentada cada parte de la estructura, con el fin de que esto no sea complicado de realizar.

4. ***El programa debe ser eficiente:*** Esto significa que un programa debe de realizar las funciones para las que fue creado de manera correcta ocupando el mínimo de recursos del equipo posible, además en su programación es deseable que sea fácil de leer y de comprender para su mantenimiento y modificación.

- Para lograr cada uno de estos puntos se recomienda seguir las siguientes etapas durante la programación.

Conceptos Básicos

- **Etapas de programación.**

1. ***Identificación del problema y definición del objetivo:*** En primer lugar se tiene que establecer si un problema realmente existe y se tiene que determinar que es lo que se pretende evaluar a través de la solución de dicho problema.
2. ***Análisis del problema:*** Una vez establecida la existencia y la necesidad de solución es necesario conocer a fondo el problema que se pretende resolver antes de proceder a desarrollar la solución a este.
3. ***Descripción matemática:*** Es necesario el desarrollo de una descripción matemática pues es la única manera en que la computadora será capaz de resolver el problema, deberá incluir todas las justificaciones teóricas y las manipulaciones matemáticas que permitan una mayor simplificación del modelo y así reducir la complejidad de este.

Conceptos Básicos

- **Etapas de programación.**

4. ***Desarrollo de solución:*** Se deben conocer y entender los diferentes métodos que permiten dar solución al problema y así poder elegir adecuadamente el que mejor funciona o poder incluir todas las diferentes formas de solución en la aplicación que se pretende desarrollar. Aquí el análisis numérico juega un papel importante pues dependiendo de la descripción matemática se podrá elegir mejor la herramienta numérica que permita a la computadora resolver el modelo.
5. ***Algoritmo:*** El desarrollo del algoritmo computacional involucra la transformación del modelo matemático en un método numérico de tal forma que la computadora pueda interpretarlo. Se necesita el desarrollo de una organización lógica para su aplicación en la computadora. Para esto se puede hacer uso de los diagramas de flujo que pueden ayudar en dicha organización.

Conceptos Básicos

- **Etapas de programación.**

6. ***Prueba de escritorio:*** Consiste en realizar los pasos de la solución del problema de manera escrita o manual con la finalidad de verificar que la solución propuesta devolverá los resultados deseados.
7. ***Construcción de solución en forma de programa:*** Esta etapa es mecánica, pues consiste en la construcción en forma de un programa, la solución desarrollada en la etapa anterior siguiendo la lógica de programación y respetando las reglas del lenguaje se vaya a utilizar.

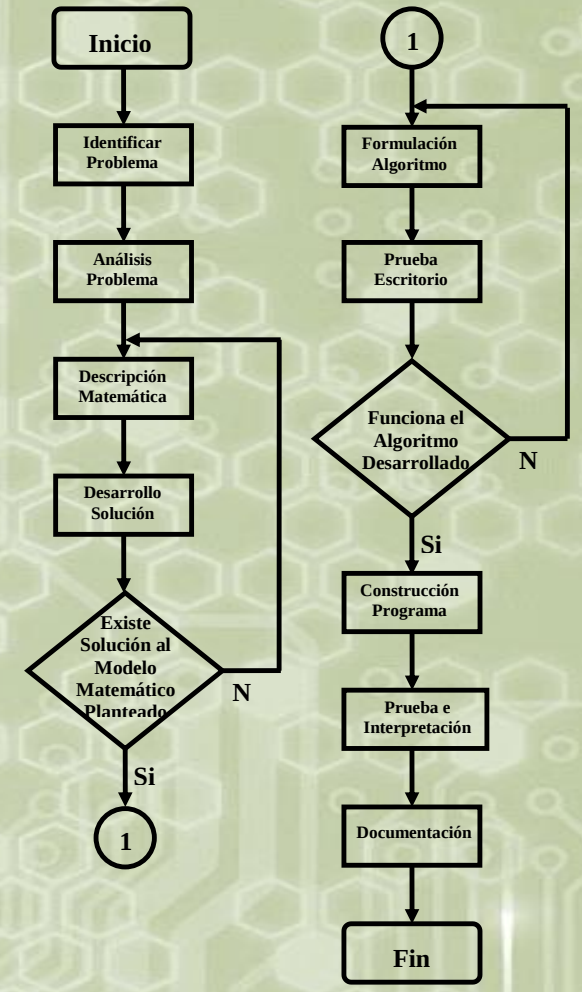
Conceptos Básicos

- **Etapas de programación.**

8. ***Prueba e interpretación:*** Es responsabilidad de los programadores realizar las pruebas necesarias que garanticen el correcto funcionamiento del programa. Estas se deben hacer con los casos más representativos y que simulen cada una de las posibilidades de solución del problema que se busca resolver a fin de comparar los resultados producidos por la computadora mediante el programa realizado con otros obtenidos por medio de alguna herramienta previamente desarrollada.
9. ***Documentación:*** Permite que el usuario final pueda manipular de manera correcta el programa. Además puede ayudar en el mantenimiento del mismo. Es lo que comúnmente se conoce como manual.

Conceptos Básicos

- **Etapas de programación.**
- Estas etapas se resumen en el siguiente diagrama de flujo:



Conceptos Básicos

- **Tipos de programación.**
- En la actualidad existen muchas opciones para desarrollar un programa, la gran variedad de lenguajes de programación ofrecen diversas ventajas y al mismo tiempo desventajas para el programador. Para el objetivo es importante reconocer la existencia de dos corrientes en la programación, primero la ***programación estructurada*** y después la ***programación orientada a objetos***. Hoy en día se tienen algunas más como son la programación de aplicaciones móviles, sitios web, etc.
- La primera es la respuesta al tipo de programación que se realizaba al inicio de la utilización de las computadoras, esta se conocía como ***programación libre*** y hacia uso de algo conocido como ***salto incondicional,*** se trata de una instrucción de programación con la cual, el contador del programa toma un valor nuevo, que el programador indica.

Conceptos Básicos

- **Tipos de programación.**
- Este método se empleó en las primeras técnicas de programación. Por ejemplo algunas líneas de FORTRAN® en programación libre son las mostradas a la derecha.
- Cuando la ejecución del programa llega a la instrucción ***GOTO***, la siguiente sentencia ejecutada será la que se encuentre en: ***100***. El abuso de esta, aparentemente ágil sentencia, da lugar a problemas graves en programas robustos y complejos.

```
PROGRAM PRIMERO
        IMPLICIT NONE
        REAL(8)::A,B,C
        WRITE(*,*)'INDICA A'
        READ(*,*)A
        100 WRITE(*,*)'INDICA B'
        READ(*,*)B
        C=A/B
        WRITE(*,*)C
        IF C=2.5 THEN
        GOTO 100
        END IF
END PROGRAM
```

Conceptos Básicos

- **Tipos de programación.**
- El salto incondicional no es necesario en un lenguaje de programación de alto nivel, ya en los años 60 se dieron cuenta de ello, desarrollándose técnicas de programación que no lo utilizaban, sin embargo los lenguajes aún la tienen como recurso, pero las técnicas así como los expertos recomiendan no usarlo, incluso prohíben su uso.
- En la industria petrolera se utiliza de manera muy importante el lenguaje FORTRAN® que es de programación estructurada, apoyándose en algún otro software como MATLAB® para la parte gráfica que permite interpretar los resultados obtenidos con los programas realizados en FORTRAN®.
- Otros Ingenieros desarrollan habilidades en programación orientada a objetos como es VisualBasic®, lenguaje que sirve como base para programar aplicaciones en Excel®.

Conceptos Básicos

- **Programación Estructurada.**
- La programación estructurada es una técnica en la cual la estructura de un programa se realiza tan claramente como sea posible mediante el uso de tres estructuras lógicas de control:
  - 1. Secuencia: Sucesión simple de dos o mas operaciones.
  - 2. Selección: Es una condicional de una o mas operaciones.
  - 3. Iteración: Repetición de una operación mientras se cumple una condición.
- Esto hace innecesario el uso de la instrucción o instrucciones de transferencia incondicional (GOTO).
- Un programa estructurado esta compuesto de segmentos, los cuales pueden estar constituidos por unas pocas instrucciones o por varias páginas de codificación.

Conceptos Básicos

- **Programación Estructurada.**
- Una característica importante de un programa estructurado es que puede ser leído en secuencia, desde el comienzo hasta el final sin perder la continuidad de la tarea que cumple el programa, lo contrario de lo que ocurre con otros estilos de programación. Esto es importante debido a que, es mucho más fácil comprender completamente el trabajo que realiza una función determinada, si todas las instrucciones que influyen en su acción están físicamente contiguas y encerradas por un bloque.
- La facilidad de lectura, de comienzo a fin, es una consecuencia de utilizar solamente tres estructuras de control y de eliminar la instrucción de desvío de flujo de control, excepto en circunstancias muy especiales tales como la simulación de una estructura lógica de control en un lenguaje de programación que no la posea.

Conceptos Básicos

- **Programación Estructurada.**
- Partes de un programa estructurado: El código de un programa elaborado de forma estructurada cuenta con cuatro secciones principales:
  - 1. Declaración de Variables: En esta sección se declara todas las variables de entrada y salida que se utilizarán a lo largo de todo el programa.
  - 2. Ejecución de las Instrucciones: En esta sección se indica el manejo que se dará a las variables declaradas mediante las operaciones indicadas.
  - 3. Formato de los datos de entrada y salida.
  - 4. Terminación: En esta sección se indica el término del programa.
- Fortran es un lenguaje de programación desarrollado en los años 50 y activamente utilizado desde entonces. Y su nombre lo toma del acrónimo de "Formula Translation". Se utiliza principalmente en aplicaciones científicas y análisis numérico. Desde 1958 ha pasado por varias versiones, entre las que destacan FORTRAN® II, FORTRAN® IV, FORTRAN® 77, FORTRAN® 90, FORTRAN® 95 y FORTRAN® 2003, VISUAL FORTRAN®. Para este curso utilizaremos PLATO® que proporciona una manera sencilla y gratuita de utilizar una versión de FORTRAN®, y que permite compilar códigos fuente escritos en FORTRAN® 90, es decir, programación estructurada.

Conceptos Básicos

- **Programación Orientada a Objetos.**
- La programación orientada a objetos (POO) es una forma de programación que utiliza objetos ligados mediante mensajes para la solución de problemas. Esta corriente surge a partir de la necesidad de volver la interacción del usuario con la computadora más amigable, además del surgimiento de sistemas operativos de tipo gráfico como lo son Windows ® y Mac OS ®.
- **Mecanismos básicos de la POO:** Los mecanismos básicos de la programación orientada a objetos son:
  - Objetos. Un programa tradicional se compone de procedimientos y datos. Un programa orientado a objetos se compone solamente de objetos. Un objeto es una entidad que tiene unos atributos particulares, las propiedades, y unas formas de operar sobre ellos, los métodos. Por ejemplo, una ventana del sistema operativo Windows® es un objeto. El color de fondo de la ventana, el alto, etc., son propiedades. Las rutinas, lógicamente transparentes al usuario, que permiten maximizar la ventana, minimizarla, etc., son métodos.

Conceptos Básicos

- **Programación Orientada a Objetos.**
  - Mensajes. Cuando se ejecuta un programa orientado a objetos, los objetos están recibiendo, interpretando y respondiendo a mensajes de otros objetos.
  - Métodos. Un mensaje esta asociado con un procedimiento, de tal forma que cuando un objeto recibe un mensaje la respuesta a ese mensaje es ejecutar el procedimiento asociado. Este procedimiento recibe el nombre de método.
- En adición, las propiedades permitirán almacenar información para dicho objeto. Un método puede también enviar mensajes a otros objetos solicitando una acción o información. La estructura más interna de un objeto esta oculta para otros usuarios y la única conexión que tiene con el exterior son los mensajes. Los datos que están dentro de un objeto solamente pueden ser manipulados por los métodos asociados al propio objeto.
- **Propiedades de los objetos:** Cada clase de objeto tiene predefinido un conjunto de propiedades, como título, nombre, color, etc. Las propiedades de un objeto representan todos los datos que por definición están asociadas con ese objeto.

Conceptos Básicos

- **Tarea #1**
- Investigación: Diagramas de Flujo.
  - ¿Qué son?.
  - Razones para el uso de diagramas de flujo.
  - Símbolos utilizados en la construcción de un diagrama de flujo.
  - Reglas para la construcción de diagramas de flujo.
  - Etapas en la construcción de un diagrama de flujo.
  - Realizar el diagrama de flujo para realizar unas deliciosas quesadillas de por lo menos tres guisados diferentes con queso.

Conceptos Básicos

- **Aritmética de las computadoras.**
- Las computadoras no almacenan los números con precisión infinita sino de forma aproximada empleando un número fijo de *bits* o *bytes.* Prácticamente todas las computadoras permiten al programador elegir entre varias representaciones o “tipos de datos”, los diferentes tipos de datos pueden diferir en el número de bits empleados, pero también en cómo el número representado es almacenado en formato entero (integer) o como real (punto flotante).
- De lo anterior es importante mencionar que un Bit es la mínima expresión que puede ser representada por una computadora mientras que un Byte es un conjunto de 8 Bits.

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Sistemas de números.***
- **Sistema decimal:** En el sistema decimal se utilizan los diez dígitos (0, 1, 2, 3, 4, 5, 6, 7, 8, 9) para representar números. La posición de cada dígito es la que determina el valor del número que se quiere representar.
- **Sistema binario:** Este es el sistema que utilizan las computadoras para representar los números, y como el nombre lo indica es un sistema con base 2 y los únicos números que utiliza son los dígitos uno y cero (1 y 0).

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Sistemas de números.***
- **Sistema hexadecimal:** Este sistema es utilizado por algunos tipos de computadoras, la base de este es 16 y utiliza los diez dígitos (0, 1, 2, 3, 4, 5, 6, 7, 8, 9) además de las letras A, B, C, D, E y F, que toman los siguientes valores:

A=10	D=13
B=11	E=14
C=12	F=15

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Sistemas de números.***
- **Ejemplos.**
- Para un entero: $5692=5*10^3+6*10^2+9*10^1+2*10^0$
- Para un real: $6921.368=6*10^3+9*10^2+2*10^1+1*10^0+3*10^{-1}+6*10^{-2}+8*10^{-1}$
- La base del sistema decimal es 10, un número binario a base 10 se desarrolla así:
  - $[10100101]_2=1*2^7+0*2^6+1*2^5+0*2^4+0*2^3+1*2^2+0*2^1+1*2^0=128+32+4+1=[165]_{10}$
- Los números enteros pueden ser representados exactamente por el sistema binario.
- Un número hexadecimal a base 10:
  - $[39]_{16}=3*16^1+9*16^0=57_{10}$
  - $[EA9D]_{16}=14*16^3+10*16^2+9*16^1+13*16^0=57344+2560+144+13=[60061]_{10}$

Conceptos Básicos

- **Tarea #2**
- Convertir los siguientes números a binario, decimal o hexadecimal según sea el caso:
  - $[11001101]_2$ convertir a base 10 y 16.
  - $[1256]_{10}$ convertir a base 2 y 16.
  - $[10011010]_2$ convertir a base 10 y 16.
  - $[1100111111]_2$ convertir a base 10 y 16.
  - $[226]_{10}$ convertir a base 2 y 16.
  - $[666]_{10}$ convertir a base 2 y 16.
  - $[EBD1]_{16}$ convertir a base 2 y 10.
  - $[A2B2]_{16}$ convertir a base 2 y 10.
  - $[11111]_2$ convertir a base 2 y 10.
  - $[9531]_{10}$ convertir a base 2 y 16.

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto fijo (Enteros o INTEGER).***
- Las computadoras almacenan los números con un número finito de bits, por lo tanto, el número de bits limita el valor del número que se puede representar.

3 bits como máximo	$(111)_2$	=4+2+1=7	$=2^3-1$
4 bits como máximo	$(1111)_2$	=8+4+2+1=15	$=2^4-1$
5 bits como máximo	$(11111)_2$	=16+8+4+2+1=31	$=2^5-1$
N bits como máximo			$=2^n-1$

- Por ejemplo, una computadora que trabaje con 16 bits, los enteros están comprendidos entre:

$$-(2^{16} - 1) \text{ y } 2^{16} - 1$$

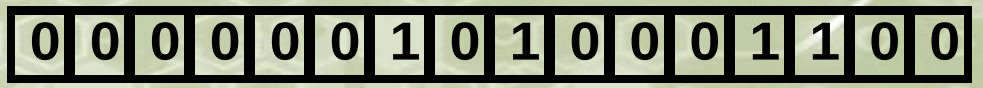
- Se tendrá un intervalo de: [-32768,32767]

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto fijo (Enteros o INTEGER).***
- Un número representado en formato entero es “exacto” por lo tanto las operaciones aritméticas entre números enteros son también “exactas” siempre y cuando:
  - La solución no esté fuera del rango del número entero más grande o más pequeño que se puede representar (generalmente con signo). En estos casos se dice que se comete un error de desbordamiento por exceso o por defecto (en inglés: *Overflow* y *Underflow*).
  - La división da lugar a un número entero, despreciando la parte después del punto decimal.
- Por estos motivos, la aritmética de punto fijo se emplea muy raramente en cálculos de alto nivel.

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto fijo (Enteros o INTEGER).***
- Los números enteros se almacenan como un número binario por medio de una cadena de bits unidos. Si se trabaja con 16 bits se contaría con quince posiciones para colocar el número y una para el signo (sign bit), el signo se representa con 0 para el positivo (+) y 1 para el negativo (-). Por Ejemplo:
  - $[652]_{10}=[1010001100]_2$



  - $[-761]_{10}=[1000001011111001]_2$

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto fijo (Enteros o INTEGER).***
- De igual forma se trabajaría con 32 bits, se tendrían treinta y una posiciones para el número y una para el signo, si se tratara de 64 bits tendría sesenta y tres posiciones para el número y una para el signo.

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto flotante (Reales).***
- **Notación científica normalizada.**
- En el sistema decimal, cualquier número real puede expresarse mediante notación científica normalizada. Para expresar un número en notación científica normalizada se multiplica o divide por 10 tantas veces como sea necesario para que todos los dígitos aparezcan a la derecha del punto decimal de modo que el primer dígito después del punto no sea cero. Por ejemplo:

$$563.2365=0.5632365*10^{3}$$

$$-0.006954=-0.6945*10^{-2}$$

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto flotante (Reales).***
- **Notación científica normalizada.**
- Un número real *x* distinto de cero, se representa en notación científica normalizada en la forma:

$$x = \pm r * 10^{n}$$

- En donde *r* es un número tal que $1/10 \leq r < 1$ y *n* es un entero (positivo, negativo o cero). Del mismo modo se puede utilizar la notación científica en el sistema *binario*. En este caso:
- $$x = \pm q * 2^{m}$$
- En donde *m* es un número entero, *q* se denomina *mantisa* y el entero *m* *exponente*.

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto flotante (Reales).***
- **Notación científica normalizada.**
- En una computadora *q* y *m* están representados como números en base 2. Puesto que la mantisa *q* está normalizada, en la representación binaria empleada se cumplirá que:

$$\frac{1}{2} \leq |q| < 1$$

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto flotante (Reales).***
- **Representación de números con punto flotante.**
- En una computadora los números en punto flotante se representan de la manera descrita anteriormente, pero con ciertas restricciones sobre el número de dígitos de *q* y *m* impuestas por la cantidad de bits disponibles y que se van a emplear para almacenar un número. Existen dos formatos definidos como precisión sencilla y doble precisión que trabajan con 32 bits y 64 bits respectivamente.
- De igual forma que con un número entero el signo se representa por un cero cuando es positivo (+) y con uno si el signo es negativo (-).

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto flotante (Reales).***
- **Representación de números con punto flotante.**
- La manera en que se distribuyen cada uno de los bits es la siguiente:

32 Bits	
Signo del número real x:	1 bit
Signo del exponente m:	1 bit
Exponente (entero |m|):	7 bits
Mantisa (número real |q|):	23 bits

64 Bits	
Signo del número real x:	1 bit
Signo del exponente m:	1 bit
Exponente (entero |m|):	10 bits
Mantisa (número real |q|):	52 bits

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto flotante (Reales).***
- **Representación de números con punto flotante.**
- En la mayoría de los cálculos en punto flotante las mantisas se normalizan, es decir, se toman de forma que el bit más significativo (el primer bit) sea siempre 1. Por lo tanto, la mantisa q cumple siempre:

$$\frac{1}{2} \leq |q| < 1$$

- Como la mantisa siempre se representa normalizada, el primer bit en q es siempre 1, por lo que no es necesario almacenarlo proporcionando un bit significativo adicional.

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto flotante (Reales).***
- **Representación de números con punto flotante.**
- Se dice que un número real expresado como aparece en la ecuación

$$x = \pm q * 2^m$$

y que satisface

$$\frac{1}{2} \leq |q| < 1$$

Tiene la forma de punto flotante normalizado. Si además puede representarse exactamente con $|m|$ ocupando 7 bits y $|q|$ ocupando 24 bits, entonces es un número de computadora de 32 bits.

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto flotante (Reales).***
- **Representación de números con punto flotante.**
- La restricción de que $|m|$ no requiera más de 7 bits significa que:

$$|m| \leq (1111111)_2 = 2^7 - 1 = 127$$

- Ya que $2^{127} \approx 10^{38}$ con 32 bits se puede manejar números tan pequeños como 10^{-38} y tan grandes como 10^{38}. Este no es un intervalo de valores suficientemente grande, por lo que en muchos casos se deben utilizar programas escritos en doble precisión.

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto flotante (Reales).***
- **Representación de números con punto flotante.**
- Como *q* debe representarse empleando no más de 24 bits significa que los números de computadora tienen una precisión limitada cercana a las siete cifras decimales, ya que el bit menos significativo de la mantisa representa unidades de:

$$2^{-24} \approx 10^{-7}$$

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Números de punto flotante (Reales).***
- **Representación de números con punto flotante.**
- Por lo tanto, los números expresados mediante más de siete dígitos decimales serán aproximados cuando se almacenen en la memoria de la computadora. Esto genera un tipo de error conocido como error de redondeo, este debe considerarse pues aunque para cálculos pequeños pueden no ser significativos para programas más robustos en los cuales resultados previos se utilicen en iteraciones posteriores para la solución del problema al que están dirigidos podrían llegar a ser relevantes, además existe otro tipo de error importante conocido como error de truncamiento.
- Ambos tipos de error se tratarán mas adelante con mayor profundidad.

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Machine epsilon.***
- El Machine Epsilon es el número decimal más pequeño que sumado a 1, la computadora arroja un valor diferente de uno, es decir, el número no es redondeado.
- Representa la exactitud relativa de la aritmética de la computadora, es decir, define la precisión que tiene la computadora y depende de las características de esta, por ejemplo: procesador, memoria RAM, velocidad, etc.

Conceptos Básicos

- **Aritmética de las computadoras.**
- ***Machine epsilon.***
- El siguiente código realizado en FORTRAN® permite conocer el Machine Epsilon de una maquina:

```
MACH1=2.0
MACH=1.0
I=0.0
DO WHILE (MACH1>1.0)
        MACH=MACH/2.0
        MACH1=MACH+1.0
         I=I+1.0
        WRITE(20,*)I,'    ',MACH
END DO
```

GRACIAS

Ing. Juan Carlos Sabido Alcántara

Ingeniero Petrolero

Facultad de Ingeniería UNAM