Estructura de Datos L Ma J

Prof. Lic. Sonia Salazar León
Materia: Estructura de Datos

Temario:

UNIDAD I: Medios de Almacenamiento

1. Memoria Primaria
2. Memoria secondaria

UNIDAD II: Estructura de Datos Elementales

1. Representación de Números en computadoras
2. Representación de Caracteres
3. Representación de Arreglos

UNIDAD III Listas Ligadas y Doblemente Ligadas

1. Almacenamiento contiguo y Ligado
2. Pila
3. Cola
4. Cola Doble
5. Lista circular
6. Lista Doblemente Ligada

UNIDAD IV Recursividad y Arboles

1. El concepto de Recursividad
2. Algoritmos Recursivos
3. Arboles
4. Arboles binarios

UNIDAD V Tecnicas de Ordenamiento y Busqueda

1. Ordenamiento Interno
2. Ordenamiento Externo
3. Busqueda por Comparación
4. Busqueda por Transformación de llaves

1 5° Parcial } Recopilación Apuntes

Bibliografia

1. Sedgewick
   Algoritms in C++
   Ed Addison Wesley
   E.U.A., 1992

2. Weiss Mark Allen
   Data structuras and algorithm analysis
   Ed Addison Wesley
   E.U.A. 1992

3. Krise, R.L, Leung, B.P., Tondo C.L
   Data structuras and program Desing in C

4. Tenebaum AM. Langsam
   Estructura de Datos en C
   ED. Prentice Hall
   E.U.A 1993

5. Kruse Robert L.
   Programing with data structures
   Ed Prentice Hall
   E.U.A. 1989

6. Lipschutz, Seymour
   Estructura de Datos
   Ed Mc Graw Hill
   México 1987

7. Wirth Niklaus
   Algoritmos y Estructura de Datos
   ED Prentice Hall
   Mex 1987

CALIF

http://orbita.starmedia.com

http://www.itlp.edu.mx

Evaluación:
Examen 50%
Programas 30%
Tareas, Participación, Asistencia } 20%

Pasar los apuntes a PC.

* Entregas Programas en C

* Tareas y Programas

Definiciones Conceptuales y Entregables

GPO :
Nombre:
Nº Tarea
Fecha
Titulo
Ref. Bibliografica

Desarrollo

GPO
Nombre
Nº Prog. Ejemplo.c
Titulo:
Fecha

Codigo Fuente

Prog.exe

corrida

y se entrega Diskette

Unidad I Medios de Almacenamiento

Introducción:

Para el estudio de las estructuras de datos se requiere del conocimiento y algunos aspectos relevantes de la organización y operación de la computadora que nos proporciona un marco de referencia.

La unidad comprende la revisión de los componentes generales de una computadora de ahí se parte hacia aquellos que son relevantes en el diseño y operación de las estructuras de datos, como son los medios de almacenamiento (Memoria).

Estos se han dividido para su estudio en medios de almacenamiento primario y secundario (D.D), en ambos se realiza una revisión de la organización física y lógica y se definen los conceptos asociados a las estructuras de datos.

Para el estudio de la programación de una computadora es necesario conocer los elementos físicos que la integran y sus características asociadas a desarrollo de programas.

Una buena programación generalmente esta asociada al conocimiento de operación del eq.

los elementos de una computadora.

Lo constituyen basicamente el hardware (Física) software (Programas) y datos. El hardware esta constituído por las partes electronicas y mecanicas q' le permiten a la pc realizar operaciones y el software o conjunto de instrucciones q' permiten indicar y secuenciar las operaciones sobre los datos.

componentes físicos de una computadora.

3 unidades funcionales

a) Entrada y Salida
b) Memoria
c) Unidad Central de Proceso

Dispositivos De Entrada	Memoria Primaria	Unidad Aritmetica y Logica
Dispositivos de salida	Memoria Secundaria	Registros y Control
Dispositivos E/S	Memoria	Unidad central de Proceso

Unidades Funcionales de una computadora.

7-Feb-05

* Investigar los 3 componentes Función, clasificación y Elementos.

8-Feb-05

Int

Diagrama

Dispositivos E/S

MEMORIA

UCP

Los dispositivos de E y S son los q' permiten q' la computadora la comunicación con el exterior. Es atraves de ellos. q' los usuarios proporcionan información de la computadora o reciben información. de ella.
Dentro del grupo de los dispositivos de E encontramos el teclado, mouse, scanner, lapiz optico, camara.
Dentro del gpo. de los dispositivos de salida encontramos: Monitor, impresoras, bocinas.
La memoria es la Unidad de Almacenamiento de instrucciones y datos. Para lograr esta función se utiliza una gran variedad de equipos, entre los cuales podemos distinguir los dispositivos de almacenamiento primaria (M. Primaria o memoria principal) y dispositivos de almacenamiento secundaria (mem. Secundaria).
Los dispositivos de almacenamiento primario son capaces de operar a velocidades electronicas y en ellos se encuentra las instrucciones y datos, al momento de la ejecución. Tipicamente consiste en circuitos, semiconductores y nucleos magneticos.
Los dispositivos de almacenamiento secundario operan a velocidades electromecanicas esto es a velocidades mucho menores que la memoria primaria, en ellos se encuentra instrucciones y datos no muy proximos a ser ejecutados. Ejm: D.D. CD, Flopy.
La unidad central de proceso es el dispositivo que ejecuta las instrucciones proporcionada por los usuarios. La mayoria de las operaciones indicadas en las instrucciones se practican en ALU y el secuenciamiento para la instrucción de estas operaciones es coordinada por la unidad de control.

Programas

Se llama programa a las instrucciones que proporciona el usuario a una computadora para realizar una tarea especifica. dentro de los programas que generalmente se manejan en una computadora encontramos.

8-Feb-05

Programas del sistema q' facilitan a los usuarios su interaccion con la maquina. Estos son los q' generalmente proporcionan el fabricante cuando se adquiere un equipo, como son:

- S.O
- Compiladores e Interpretes
- Ensambladores y Macro Ensambladores
- Cargadores y Ligadores
- Editores
- Prog. de Comunicación
- Etc.

Programas de usuario: Son aquellos que escriben los usuarios para aplicaciones particulares:

- Solución de un sist. Ecuaciones
- Sist. control de Asistencia
- Sist. nomina
- Modelado de Sistema.

Una PC acepta información del exterior haciendo uso de los Disp. Entrada. Esta información puede consistir en instrucciones o datos la cual es almacenada en la memoria primaria para ser procesada por U.C.P. y los resultados son enviados al exterior atraves de los dispositivos de salida.

Memoria primaria

La Memoria primaria se divide en 2 partes: la organización fisica q' se refiere a la organización de los componentes fisicos y la organización logica q' se refiere a la forma de organización de la memoria.

Organización Fisica

La UCP toma instrucciones y datos de la memoria primaria en grupos de n bits, llamados palabras de computadoras.

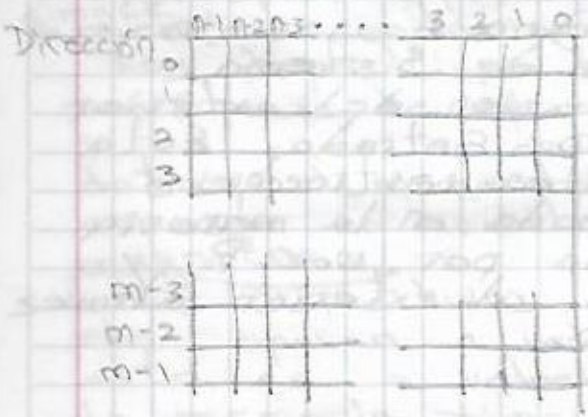
Un bit es un digito binario el cual solo puede valer 0 y 1

La longitud de una palabra es el numero de bits que la componen, en la mayoria de los computadoras esta longitud oscila ≈ 8 y 64 bits.

Bit bit { 0 Apagado 1 Encendido

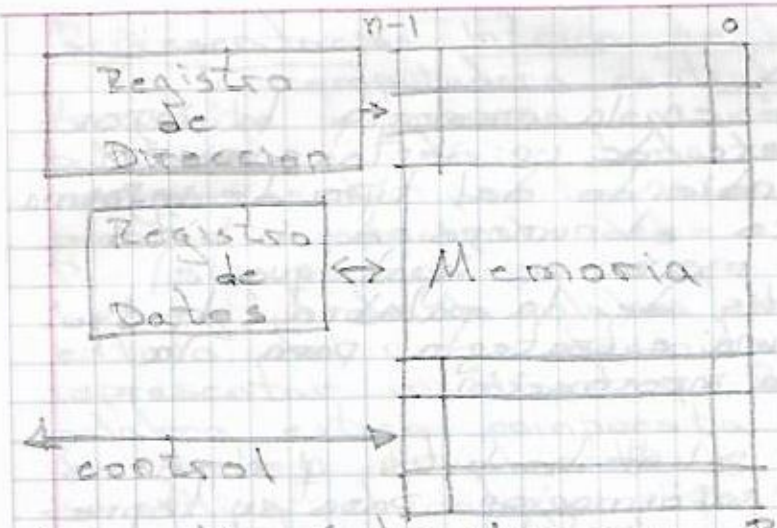
byte { 8 bits 1 caracter

Un bit byte representa 2 objetos distintos de aqui que una palabra de long. n, 2^n objetos distintos. La memoria fisica esta constituida por un conjunto de m palabras de long. n en donde a cada palabra se le asocia una dirección, que es un numero unico entre 0 y m-1 llamado la dirección de la palabra.



Organización Física de la Memoria

Para escribir (almacenar) o leer (recuperar) información desde la UCP existen asociado a la memoria dispositivos como el registro de dirección, el registro de datos y lineas de control, mismos que permiten la realización de estas operaciones.



Conexión de la Memoria Primaria y la UCP

El registro de dirección es un conjunto de k bits en el cual es posible mantener cualquier dirección entre 0 y n-1. Este registro es utilizado para seleccionar la dirección de la palabra con la q' se desea operar.

El registro de datos es un cjto de n bits en los cuales se tiene el contenido de la palabra direccionada al finalizar una operación de lectura. En una operación de escritura, el registro mantiene la información que será depositada en la palabra direccionada.

El control permite indicar a la memoria que operación se va a realizar (Escritura o lectura. Através del control la memoria indica el termino de la operación.

Para leer o escribir en la memoria la Unidad central carga los registros de dirección y de datos con los valores apropiados y envia la señal p' q' se inicie la operación.

El tiempo entre la señal de inicio de la operación y el termino de esta es conocida como tiempo de acceso.

El tiempo para leer o escribir en cualquier localidad de la memoria primaria es constante, esta es una característica de la memoria de acceso directo (RAM).

Organización lógica

Para representar en la memoria la información esta requiere de un cierto numero de b bits dependiendo del tipo de información de q' se trata el numero de b bits podra ser mayor, menor o igual que el numero de n bits de la palabra, de aqui que debe seguirse una estrategia para la representación de la información.

1er Caso: Sea b el # de bits q' requiere cierta información para su representación. Y $b \leq n$. esto significa q'c de una palabra de computadora pueden tomarse b bits continuos para representar la información

Informatica Basica

Eduardo Alcantara
Miguel Garcia
2ª Edición
Ed McGraw Hill

Prox. Lunes
No entregar

Representación de:

- Numeros Enteros
- Numeros Flotantes
- coma o punto Fijo
- coma flotante
- Pag. 103 bus U&S cap 3
- Como se representa la información U3
- Componente U. control, ALU
- Direccion, Intercambio y selector de Memoria
- U Funcional nivel logico
- Memoria central

Pag 528

14-Feb-05

Representación Interna de la Información:

En la computadora central la información se transmite y procesa en unidades denominadas palabras, es decir toda la información, para ser procesada por la ALU o transferida a memoria principal debe estructurarse en palabras Por esta razón para aprobechar bien la memoria la longitud de la palabra debe ser un multiplo entero del numero de bits utilizados para representar un caracter. Por consiguiente, una palabra estara compuesta por un numero entero de bytes (Usualmente 8, 16, 32, 64) cuando se utiliza el código: EBCDIC y ASCII.

Los datos se introducen inicialmente en la computadora, según su codigo E/S.

Los de tipo numerico se utilizan normalmente para operar aritmeticamente con ellos, la representación obtenida en codigo E/S no resulta adecuada para realizar este tipo de operaciones. Para mostar este hecho, consideremos la representación en codigo ASCII extendido del Numero 176:

$$(176)_{10} = (00110001\ 00110111\ 00110110)_{ASCII}$$

mientras que la representación en binario natural de dicho numero es:

$$(176)_2 = (10110000)_2$$

Como la representación del codigo en el 1er ejm es demasiado larga, resulta mas adecuado utilizar una representación basada en binario natural para realizar operaciones.

La representación interna esta basada en binario natural x q' permite un mejor aprovechamiento de la memoria principal y mayor velocidad en los calculos numericos.

Representación Interna de los Distintos Tipos de Datos.

Los tipos de datos más elementales q' puede soportar un computadora son:

Logico,
Caracter
Entero
Real

Los Datos de tipo lógico pueden tomar dos valores (0, 1) que representan un valor de verdad (1) o de falsedad (0).
La representación interna de este tipo de datos es muy variable.

Los datos de tipo caracter representan sencillamente cadena de caracteres ensamblados en las palabras del ordenador y representados según el código E/S (codigo BCD), EBCDIC, ASCII o ASCII extendido los cuales son codigos que asocian a cada caracter una determinada secuencia de bit's).

En este caso, no es necesario efectuar una representación de conversión.

Cada palabra de memoria puede contener tantos datos o caracteres como indique su longitud (por ello, el numero de bit de la palabra es un multiplo entero del numero de bit's con el que se representa un caracter)

Para leer un caracter de la memoria, es necesario leer la palabra de memoria que lo contiene.

Los datos de tipo entero y real son los mas utilizados por ello su representación se analiza con más detalle.

Alumno
UCP

Examen Temas U. control, U. A.L,

*** T. Tipos de estructura de Datos
Estaticas } Dinamicas } Rango de Tipos de Datos Valores en C
T3 T2

Prox. clase: Metodos de Direccionamiento

*** Exposición: De el Disco Flexible, Almacenamiento.

Ejecución de Instrucciones

Unidad de Control.

Es el centro nervioso de la computadora ya que desde ella se controlan y gobiernan todas las operaciones.

Consta de los sig elementos.

1. Contador de programa:
   Denominado tambien registro de control de secuencia, contiene permanentemente la dirección de memoria de la siguiente instrucción a ejecutar. Al iniciar la ejecución de un programa toma la dirección de su primera instrucción. Incrementa su valor en 1 de forma automatica, cada vez que se concluye una instrucción, salvo si la instrucción que se esta ejecutando es de salto o de ruptura de secuencia, en cullo caso el contador de programa tomara la dirección de instrucción q se tenga q ejecutar a continuación., esta dirección esta en la propia instrucción en curso.

2. Registro de instrucción; RI
   Se encarga de ~~extraer~~ el contiene la instrucción que se esta ejecutando en cada momento. Esta instrucción llevara consigo el codigo de operación y en su caso los operandos o las direcciones de memoria de los mismos.

3. Decodificador.
   Se encarga de extraer el código de operación de la instrucción que se encuentra en la RI, lo analiza y emite las señales necesarias al resto de elementos para su ejecución atravez del secuenciador.

4. Reloj.
   Proporciona una sucesión de pulsos eléctricos o ciclos a intervalos constantes (Frecuencia

constante q marca los instantes en q han de comenzar los distintos pasos de q consta c/instrucción.

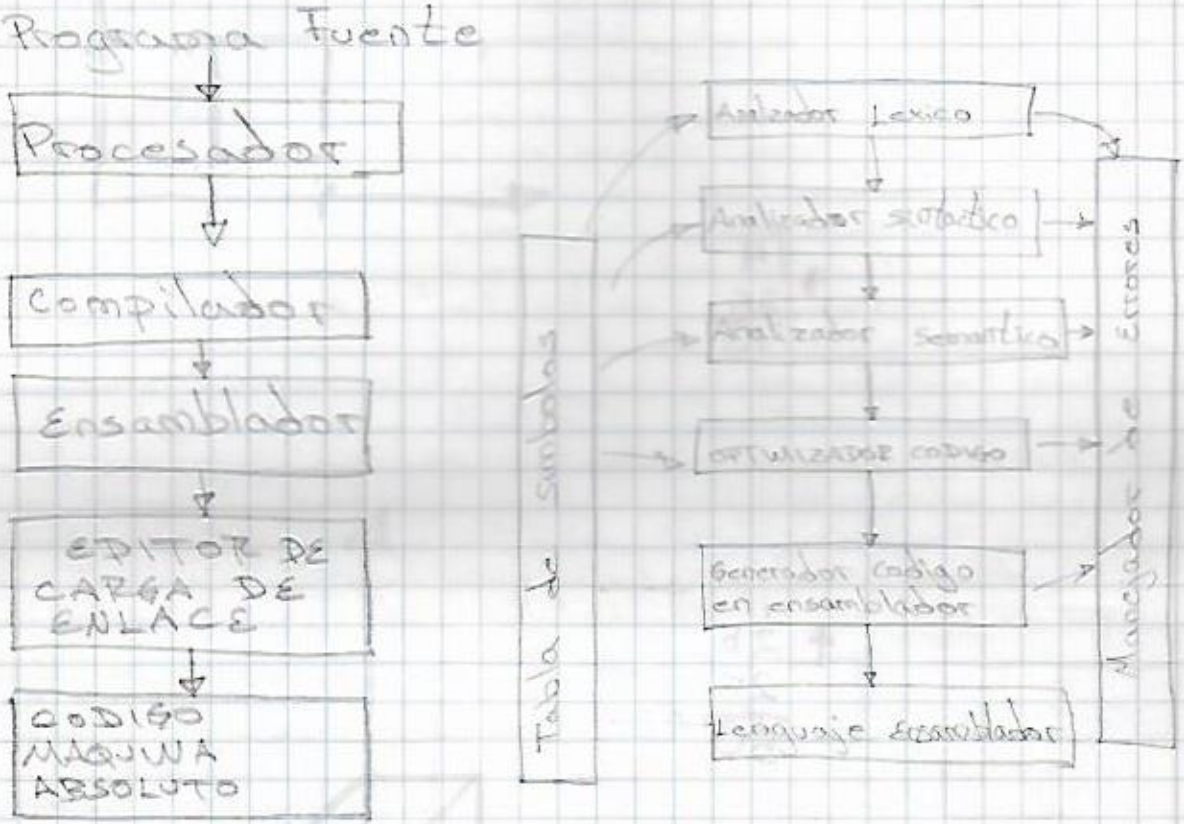
5. Secuenciador (Controlador)
En este dispositivo se generan ordenes muy elementales (microordenes) que, sincronizadas por los impulsos del reloj hacen que se valla ejecutando poco a poco la instrucción que esta cargada en el registro de instrucción.

Ejecución de Instrucciones

El CPU ejecuta instrucciones en una serie de pequeños pasos:

1. El CPU extrae de la memoria la siguiente instrucción y la lleva al registro de instrucciones.
2. Cambia el contador del programa de modo q señala a la siguiente instrucción.
3. Determina el tipo de instrucción que acaba de extraer
4. Verifica si la instrucción requiere de datos de la memoria, si es asi determina donde estan situados
5. Extrae los datos y los carga en los registros internos del CPU
6. Ejecuta la instrucción.
7. Almacena los resultados en el lugar apropiado
8. Regresa al paso 1 para empezar la ejecución de la instrucción siguiente.

A esta secuencia se le denomina Extrae - Decodifica - Ejecuta.



Unidad II

Los tipos de dato simple pueden ser el numero entero el numero real o un caracter.

En muchas situaciones se necesita, procesar una coleccion de valores q estan relacionados entre si por algun metodo. por ejemplo una lista de calificaciones una serie de temperaturas medidas a lo largo de un mes etc. El procesamiento de tales conjuntos de datos, utilizando datos simples puede ser extremadamente dificil y por ello los lenguajes de programación incluyen caracteristicas de estructuras de datos.

Las estructuras de datos basicas que soportan la mayoria de los lenguajes de programación son los arreglos (vector y Matriz).

Un arreglo es una secuencia de posiciones de la memoria central a las que se puede acceder directamente, que contiene datos del mismo tipo y pueden ser seleccionados individualmente mediante el uso de subindices..

Estructura de Datos

1. Coleccion de datos cuya organización se caracteriza por las funciones de acceso q' se usan para almacenar y acceder a elementos individuales de datos.

2. Forma de organizar un conjunto de datos elementales con el objetivo de facilitar la manipulación de los mismos.

3. Forma de organizar la información en determinado lenguaje de programación.

4. Es una coleccion de datos que pueden ser caracterizados por su organización y las operaciones que se definen en ella, c

5. Consiste en una coleccion de nodos o registros que mantienen, importantes relaciones entre si.

Representación de Numeros Enteros.

Las computadoras utilizan 4 métodos p' la representación interna de # enteros

- Enteros, Flotantes } Numericos
- Caracteres → Alfanumericos
  - Codigo ASCII
  - " " Extendido
  - EBCDIC
- Logicos → Comparaciones { AND, OR, NOT

~~Representación Interna de los Tipos de Datos~~

(Positivos y negativos que son) Pag 83

1. Modulo y Signo (MS)
2. Complemento a 1 (C-1)
3. Complemento a 2 (C-2)
4. Exceso a 2^{n-1}

1.- Modulo y Signo

En este sistema de representación el digito que esta situado mas a lequierda representa el signo, y su valor sera cero para el signo positivo y 1 para negativo. El resto de los bits (n-1) representan el modulo del número

7 6 5 4 3 2 1 0

10 → 0 0 0 0 1 0 1 0

Signo Modulo

-10 → 1 0 0 0 1 0 1 0

↑ Signo -

T.4.- Cuales son los sistemas de números q existen.

- Conversiones entre
  - decimal a binario
  - binario — Decimal
  - Decimal — Octal
  - Octal — Decimal
  - Decimal — Hexadecimal
  - Hexadecimal — Decimal
  - Octal — Binario
  - Binario — Octal
  - Octal — Hexadecimal
  - Hexadecimal — Octal

10 de c/u

El rango de representación en un metodo determinado se refiere a los numeros representado en el mismo

$$-2^{n-1}+1 \leq x \leq 2^{n-1}+1$$

n=8 bits $-127 \leq x \leq 127$

n=16 bits $-32767 \leq x \leq 32767$

Complemento a 1

Este sist. de repre

para el Sigo, correspondiendo el Cero para el positivo y el 1 para el Negativo.
Para los números positivos los n-1 bits de la derecha representan el modulo.
El negativo de un número positivo se obtiene complementando todos sus digitos (Cambiando ceros por Unos y viceversa) incluyendo el bit del Signo

del -10 (Complemento 1)

↳ 1 1110101

↑ Signo — Modulo

el rango de representación de éste método $-2^{n-1}+1 \leq X \leq 2^{n-1}+1$

Este sistema de representación posee la ventaja de tener un rango simetrico y la desventaja de tener dos representaciones para el valor cero

de 0 | 0 000 0000

Complemento A2 : este metodo de representación utiliza el bit de + a la izquierda para el signo y para el negativo es 1 y para el positivo es Cero.
En el caso de los numeros positivos los n-1 bits de la derecha representan el modulo.
el negativo de un número se obtiene en dos pasos

1er Paso → Se complementa el No. positivo en todos sus bits (Cambiando) ceros por unos y viceversa, incluido el bit del signo, es decir se realiza el complemento A1

2º Paso → Al resultado Obtenido anteriormente se le suma 1 (en binario) despreciando el último acarreo si existe.

TAREA: Como Hacer el Complemento del No. 45
ó 45,000,000

Ejem. La representación en complemento a 2 de los numeros 10 y -10 al caso de n = 8 bits es:

	+							
10	0	0	0	0	1	0	1	0
C1 -10	1	1	1	1	0	1	0	1
							1	
C-2	1	1	1	1	0	1	1	0
	-	Modulo						

$2^0 = 1$
$2^1 = 2$
$2^2 = 4$
$2^3 = 8$
$2^4 = 16$
$2^5 = 32$
$2^6 = 64$
$2^7 = 128$
$2^8 = 256$
$2^9 = 512$
$2^{10} = 124$

El banco de representación en este caso es asimetrico y se representa por la formula:

$-2^{n-1} \leq X \leq 2^{n-1} - 1$

para $n = 8$ bits

$-128 \leq X \leq 127$

para $n = 16$ bits $\quad -32768 \leq x \leq 32767$

para $n = 32$ bit $\quad -2147483648 \leq x \leq 2147483647$

La principal ventaja es la de tener una unica representación para el número cero.

0 0 0 0 0 0 0 0

1 1 1 1 1 1 1 1 (2)

+ 1

Metodo: Exceso a 2^{n-1}.

Este método de representación no utiliza ningún bit para el signo con lo cual todos los bit's representan un módulo o valor. Este valor se corresponde con el numero representado mas el exceso, que para n bit's viene dado por 2^{n-1}

Ejm: para $n = 8$ bit's el exceso es: $2^{8-1} = 2^7 = 128$

con lo cual el # 10 vendria representado por $10 + 128 = 138$

128 64 32 16 8 4 2 1

1 0 0 0 1 0 1 0 → $10 + 128 = 138$

0 1 1 1 0 1 1 0 ⟶ $= 118$

1 0 0 0 0 0 0 0 ⟶ 128

Poner los 8 bit's

El rango de representación en exceso a 2^{n-1} es asimetrico y viene dado por:

$$-2^{n-1} \leq x \leq 2^{n-1}-1$$

Para: n	Rango
8 bits	$-128 \leq x \leq 127$
16 bits	$-32768 \leq x \leq 32767$
32 bits	$-2147483648 \leq x \leq 2147483647$

Ej1 sumar los numeros 10 y -3 en complemento a 1 para n = 8 bits

```
10 -> 00001010          10 00001010
-3 -> 11111100          -3 11111101
      --------              --------
      00000110 + 1           00000111
```

Ej2 Sumar los numeros 100 y 27 en complemento a 1 con n = 8 bits.

```
100 -> 01100100
+27 -> 00011011
       --------
       01111111
```

En la aritmetica de C-2, 2 numeros se suman, de igual forma que en complemento a 1, con la unica diferencia de que se desprecia el ultimo acarreo en caso de existir.

Ej. Sumar los numeros 10 y -3 en C-2 para n = 8 bit's

Ej. En complemento a 2 C-2 de 100 + 27 es el mismo

Conviene tener presente que p' un número determinado de bit's, la suma en C-1 y en C-2 producira desbordamiento (Resultado no representable o fuera de rango) Cuando siendo los 2 sumandos del mismo signo el resultado aparezca con el signo contrario al de los sumandos.

Ejm:
$$\begin{array}{r} \boxed{0}\ 1\ 0\ \ldots \\ +\ \boxed{0}\ 1\ 1\ \ldots \\ \hline 1\ 0\ 1\ \ldots \end{array}$$

bit de signo ⟶ error de desbordamiendo

Ejm + 100 en complemento a 1
+ 28 para n = 8 bit's

Las representaciones de los numeros 100 y 28

100 → 0 1 1 0 0 1 0 0
28 → 0 0 0 1 1 1 0 0
1 0 0 0 0 0 0 0 0
↳ Error se sale de rango

El # 128 no puede ser representado si solo tenemos 8 bit's.

Por tanto

Ejm: Sumar 110 y 30 en C-2 p'n=8b

128 64 32 16 8 4 2 1

$$\begin{array}{r} 0\ 1\ 1\ 0\ 1\ 1\ 1\ 0 \rightarrow 110 \\ +\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0 \rightarrow 30 \\ \hline 1\ 0\ 0\ 0\ 1\ 1\ 0\ 0 \quad 140 \end{array}$$

⟶ Se sale de rango

Al utilizar la computadora metodos de representación interna de datos basados en el sistema binario, es conveniente tener presente que en una particular maquina disponemos de un # determinado de bits para expresar un numero (Generalmente una palabra de 16, 32 o 64 bit's); esto hace que los números reales sufran un determinado truncamiento en su parte decimal. Tambien se presenta problemas al utilizar la coma flotante, debido a q' no todos los numeros tienen representación exacta, por tanto esto se representa de forma aproximada con lo que aparece pequeños errores de

representación. A esto se le denomina precesión finita de la información numerica.
Podemos definir el termino palabra como la cantidad de bit's con la que una computadora esta capacitada para realizar operaciones internas. Esta cantidad queda definida para c/computadora desde el diseño por su propio fabricante, siendo en la actualidad
16, 32, 64
De cuanto se estan fabricando. 64.

Siendo en la actualidad los mas extendidos los 64 bits.

Representación en coma o Punto Fijo.

Su nombre viene de la posición en la que se supone situado el punto decimal que sera Fija.
El punto fijo es utilizado para la representación de No enteros, suponiendose el punto decimal implicitamente a la derecha de los bits.
Existen 3 formas de representar numeros en coma o punto fijo.

1) Binario Puro
Un numero en binario puro se representa utilizando un conjunto de bit's equivalente a una palabra (Para la representación normal denominada simple precisión) o a una doble palabra (Denominada Doble precisión) y la forma de hacerlo es representar su valor en binario.
Si suponemos una computadora con una palabra de 32 bits donde se utiliza el bit de más a la izq. para representar el signo, (0 p'+ y 1 p'-), siendo los 31 bit's restantes los q representan el modulo o valor en complemento a 2 ¿Cual sera el rango de representación y la configuración interna de los No. 0, 10, -10, 21, 47, 48, 3647
47483647

En esta representación el punto decimal puede suponerse situado en cualquier otro lugar distinto de la derecha de los bit's, con lo q' podemos trabajar con números Reales.

2) Decimal Desempaquetado

El sistema de codificación BCD significa (Decimal Codificado en Binario).

En este sistema cada cifra de un numero decimal se representa por un cito de 4 bits, siendo la tabla de equivalencias entre ambos la siguiente.

CIFRA Decimal	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

	1 ↓	5 ↓	
	0001	0101	
	1	9	9 H
	0001	1001	1001 0100

En el sistema de codificación decimal desempaquetado, un No. decimal se representa de tal forma q' cada una de sus cifras ocupa un octeto o un BYTE c/u. de estos octetos. lleva en su cuarteto de la izquierda 4 1's Denominados bit's de zona y en el cuarteto de la derecha la codificación de la cifra en BCD, denominandose a este cuarteto

Exceso a 2^{n-1}

bits de dígito. El cuarteto de la izq. de la última cifra del número (cifra de la derecha representa el signo del número, conteniendo: 1100 + → C
1101 − → D

Se denomina compactación de un código binario a su representación simplificada utilizando los sistemas octal o hexadecimal es decir, representado c/conjunto de 3 o 4 bit's respectivamente por su correspondiente caracter octal o hexadecimal.

1111 → F

1 9 9

1111 0001 1111 1001 11111001
bit zona bit Dígito

4

1100 0100
↓
signo +

1 9 9 4

1111 0001 11111001 11111001 1101, 0100
(−)

La compactación en Hexadecimal

F1 F9 F9 C4
F1 F9 F9 D4

Decimal Empaquetado

En este sistema se representa c/cifra decimal de un cuarteto (se eliminan los bit's de zona) salvo en el 1er octeto de la derecha que en su cuarteto tambien de la derecha lleva el signo con las mismas consideraciones q' el caso anterior.

1	9	9	4	
0001	1001	1001	0100	1100 → signo +
0001	1001	1001	0100	1101

Representación de Números en Coma o Punto Flotante

La coma o punto flotante surge de la necesidad de representar numeros Reales y enteros con un rango de representación mayor del que nos ofreces la representación en punto Fijo y posibilitar a la computadora el tratamiento de numeros muy grandes y muy pequeños. Estas ventajas que nos ofrece la coma flotante traen como contra prestación una disminución (relativamente pequeña) en la precisión de los numeros representados.

En su representación se utiliza la notación cientifica o exponencial matematica en la q' una cantidad se representa de la sig. forma:

$$n = \text{mantisa} * \text{base}^{\text{exponente}}$$

Un numero en esta notación tiene infinitas representaciones, de la que se toma como estandar la denominada "Normalizada", que consiste en que la matisa no tiene parte entera y el primer digito o cifra a la derecha del punto decimal es significativo (Distinto de cero), salvo en la representación

del número cero.

Ejm: 1) Representación del No. decimal 835.4 con base de exponenciación 10.

$835.4 = 8354 * 10^{-1} = 835.4 * 10^{1}$

$8.354 * 10^{2} = \underbrace{0.8354 * 10^{3}}$ ← hasta aquí llegamos

Normalizada

Ejm: 25.4 en notación científica normalizada

R.- 0.254×10^{2}

3. Marzo

En este sistema de codificación de #, se dividen los bits disponibles en la palabra o doble palabra de la computadora. entre la mantisa y el exponente teniendo una base de exponenciación determinada (2 o una potencia de 2) normalmente la definición de la coma flotante de una computadora sigue las sig. reglas

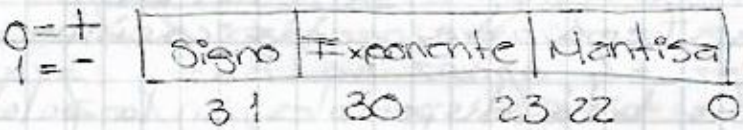
1° El exponente se representa en uno de los sig. sistemas de codificación: modulo y signo o exceso a 2^{n-1}, siendo siempre un # entero. En este sistema de codificación el exponente también recibe el nombre de característica.

2° La mantisa es un # real con el punto decimal implícito a la izquierda de sus bits, representada normalmente en uno de los siguientes sistemas de codificación: modulo y signo, complemento A-1 o complemento A-2

3- La base de exponenciación, también denominada radix es una potencia de dos determinada por el fabricante de la computadora (2, 8, 16) existen muchas formas de representación en coma flotante variando la longitud de la palabra de la computadora.

El # de bits reservados para la matisa y para el exponente. El sistema utilizado para representar la mantisa y el exponente. La coma o punto flotante se define particularmente en c/caso las definiciones mas frecuentes son las sigu.

a) Para single presicion (para computación de 32 bits)



Normalizada representación

6.8354×10^{32} → ± exponente { M.S ≡ exceso 2^{n-1}

↓

Base $b^7\ b^6\ b^5\ b^4\ b^3\ b^2\ b^1\ b^0$

128 64 32 16 8 4 2 1

8

4 2 1

8 | 4 | 2 | 2

0 0 0

1 1

64 8 1

1. Subdesbordamiento positivo:

Son los # Pequeños situados en el entorno de cero hacia la parte positiva:

$0 < X < mNP$.

2. Subdesbordamiento Negativo:

Son los # Pequeños situados en el entorno de cero hacia la parte negativa:

$MNN < X < 0$.

3. Desbordamiento Positivo:

Son los # Positivos que exceden la capacidad de representación del punto flotante:

$X > MNP$.

4. Desbordamiento negativo:

Son los números Negativos que excede la capacidad de representación del punto flotante por la parte Negativa

$X < mNN$

Además, no todos los numeros comprendidos entre el mNN y el MNN así como todos los comprendidos el mNP y el MNP tienen representación, es decir en ambos subrangos existen huecos con numeros que la computadora aproxima al mas cercano para trabajar con ellos.

Ejm: Una computadora utiliza el sig. formato para registrar numeros en punto flotante:

1. Los bit's del 23 al 30 se utilizan para representar el exponente en exceso a 2^{n-1}
2. Los bits del 0 al 22 se utilizan para representar la mantiza normalizada en c-1.
3. El bit 31 se utiliza p' representar el signo (0) p' (+) y (1) p' (-).
4. La base de exponenciación es 2.

a) Con lo anterior representar el # 12

b) Representar el # -12

c) Representar el # 0

d) Sacar el rango de representación.

$12 \times 2^0 = 2^3 = 8$ $\quad 2^{-2}$

$.12\ 10^{-2}$ $\quad .75 \times 2^4 = 12$

$12 = 0.75 \times 2^4$

Mantisa $\quad$ EXP

128 64 32 16 8 4 2 1

12 ← 0000.1100 → 4 exceso 2^{n-1}

75 ← 01001011 $\quad$ 10000100

75 c-1 ← 10110100 $\quad$ 30 $\quad$ 23

22

signo 11000000000000000000000

0

0

$* 2^{127}$

.75 * 2 =	1.5
1.5 * 2 =	1 1.0
=	

.11000

Signo	exponente	mantisa

mNN = $-1 * 2^{127} = -2^{127} = -1.70141183460 * 10^{38}$

MNN = $-0.5 * 2^{128} = -2^{-129} = -1.469367938528 * 10^{-39}$

mNP = $0.5 * 2^{-128} = 2^{-129} = 1.469367938528 * 10^{-39}$

MNP = $1 * 2^{127} = 2^{127} = 1.70$

estas características ofrecen soluciones eficaces y efectivas en la solución de (valores) problemas complejos.

Las estructuras de datos dinámicas más comunes son: Las listas (enlazadas, pilas colas), árboles (binarios) árbol-b, de búsqueda binaria) y grafos.

La elección de tipos de estructura idónea a cada aplicación dependerá esencialmente del tipo de aplicación y en menor medida del lenguaje, ya que en aquellos en que no esté incrementada una estructura, deberá ser simulada con el algoritmo adecuado.

Una característica importante que diferencia a los tipos de datos es la siguiente: "Los tipos de datos simple tienen como característica común que cada variable representa a un elemento; los tipos de datos estructurados tienen como característica común que un "identificador" puede representar múltiples datos individuales, pudiendo c/u de éstos ser referenciado independientemente.

ARRAY'S UNIDIMENSIONALES (VECTORES)

Un array (Matriz o Vector) es un conjunto finito y ordenado de elementos homogeneos. La propiedad "ordenado"

significa q° el elemento 1°, 2°, 3°... nesimo de un array puede ser identificado.

Los elementos de un array son homogeneos, es decir, del mismo tipo de datos.

Un array puede estar compuesto por todos sus elementos de tipo cadena, o tb puede tener todos sus elementos de tipo entero.

Los Array tambien se conocen como matrices o tablas.

El tipo mas simple de array es el array unidimensional o vector (Matriz de una sola dimensión)

Ejm:

Un vector de una dimensión denominados notas que consta de n elementos puede representarse de la sig. forma.

Notas(1)	Notas(2)	...	Notas[I]	...	Notas[N]

↑ Vector, Array, Tabla.

El subindice o indice de un elemento [1, 2, 3... i... n] Designa su posición en la ordenación del vector.

otras posibles notaciones son:

A(1), A(2), ... A(i) ... A(n)

Los elemtos del vector se referencian por su subindice o indice, es decir es decir su posicion relativa en el vector.

Como ejemplo de un vector o Array unidimensional, se puede considerar el vector TEMPERATURAS que contiene las tempeaturas horarias registradas en una ciudad durante las 24 horas del dia. Este vector constara de 24 elementos de tipo real.

Ya que las temperaturas normalmente no serán enteras siempre.

El valor mínimo permitido de un vector se denomina límite inferior del vector, y

El valor máximo permitido; límite superior.

Ejm:

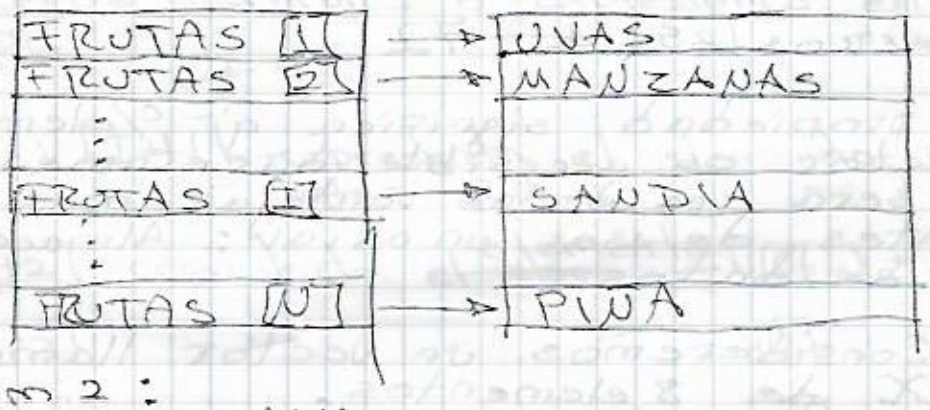
TEMPERATURAS [I] donde $1 \leq I \leq 24$

I = 1 TO 24 Elementos.

El No. de elementos de un vector se denomina rango del vector.

Los vectores pueden contener datos no numéricos, es decir de tipo caracter.

Ejm 1:



Ejm 2:

Los vectores se almacenan en memoria central de la computadora en orden adyacente. Un vector de SO ~~II~~ denominado numéros se representa gráficamente

con 50 posiciones.
Ejm:

Cada elemento de un vector se puede procesar como si fuese una variable simple al ocupar una posición en memoria.
Ejm:

NUMEROS [25] ← 72

Esta propiedad significa q' c/elemento de un vector es accesible directamente.
Esta será una de las ventajas más importantes de usar un array: Almacenar un cjto de datos.

Tarea: Consideremos un vector llamado X de 8 elementos.

X [1]	14.0	Elem. 1°
X [2]	12.0	" 2°
X [3]	6.0	
X [4]	7.0	
X [5]	6.41	
X [6]	5.23	
X [7]	6.15	
X [8]	7.25	" 8°

Realizar las sig operaciones:

1. Escribir (X [1])
2. X [4] ← 45
3. SUMAR X [1] + X [3] Almacenar en SUMA

4. SUMAR SUMA + X[4] → Almacenar en SUMA
5. SUMAR X[5] + 3.5 → Almacenar en X[5]
6 Almacenar en X[6] La suma X[1] + X[2]

10-3-5

Supongamos un vector V de 8 elementos

V[1]	12
V[2]	5
V[3]	-7
V[4]	14.5
V[5]	2.0
V[6]	1.5
V[7]	2.5
V[8]	-10

Los subindices de un vector pueden ser enteros variables o expresiones enteras ejm:

I ← 4

V[I+1] representa el elemento V[5] de valor 2.0

V[I+2] representa ~~el valor~~ elemento V[6] de valor 1.5

V[I-2] " " " V[2] " 5

V[I+3] " " " V[7] " 2.5

Un array unidimensional necesita ser dimensionado previamente a su uso dentro de un programa.

OPERACIONES CON VECTORES

Un arreglo es una secuencia ordenada de elementos y el limite inferior no tiene por que empezar forzosamente en 1.

X[1], X[2], ..., X[n]

L[0], L[1], L[2], L[3], L[4], L[5]

El arreglo L contiene 6 elementos en el que el primer elemento comienza en cero el siguiente vector llamado P tiene un rango de 7 elementos y su limite inferior y superior son: -3 y 3.

P[-3], P[-2], P[-1], P[0], P[1], P[2], P[3]

Las operaciones que se pueden realizar con vectores durante el proceso de resolución de un problema son:

1) Asignación
2) Lectura/Escritura
3) Recorrido (Acceso Secuencial)
4) Actualizar (Añadir, Borrar, insertar)
5) Ordenación
6) Busqueda

En general las operaciones con vectores implican el procesamiento o tratamiento de los elementos individuales del vector.

Notación Algoritmica

Tipo
array [Dimensión] de <Tipo de Dato>: <Nombre del tipo_array>

Ejm:
1)
Tipo
array [1...10] de caracter: nombres

Var
nombres: n

② tipo
array ['A' .. '2'] de real : lista
var
lista : l

③ tipo
array [0...102] de entero : numero
var
numero : nu

1. Asignación
La asignación de valores a ~~todos~~ un ~~los~~ elementos de un vector. se realizara con instrucción de asignación.

Ejm: A[29] ← 5

si deseamos asignar valores a todos los elementos del vector, hacemos uso de las estructuras de control repetitivas: Desde Mientras o repetir.

2. Lectura/Escritura de Datos.

La lectura escritura de datos en Array u operaciones de entrada salida tambien se realizan con estructuras repetitivas las instrucciones simples Escritura/Lectura se representaran como sigue.

leer [A] → lectura del vector A
escribir [A] → Escritura del vector A
leer (V[5] → Leer el elemento V(5) del vector V

3. Acceso secuencial al vector (Recorrido)
se puede acceseder a los elementos de un vector para introducir datos (leer) en el o bien para visualizar su contenido (Escribir). A la operación de efectuar una acción gral. sobre todos las elementos de un vector se le denomina recorrido del vector., tambien se utiliza

estructuras repetitivas, el incremento del contador del bucle, producira el tratamiento sucesivo de los elemento del vector.

Ejs. Realizar el algoritmo que efectue la lectura de 20 valores enteros de un vector denominado F

1. Array [1..20] de Integer : F
2. Desde i = 0 hasta i ≤ 19
3. Leer F[i]
4. regresa ' con i = i+1
5. Fin de Pseudo código

Prog. 1

1er Programa

```
tipo
   array [1..20] de entero : finial
var
   final: f
inicio
   Desde i ← 1 hasta 20 hacer
      leer (f[i])
   fin desde
fin
```

Prog. 2

Prog 2: Realizar un programa que declare un arreglo llamados puntos., el limite superior del rango se fijara mediante la constante limite a la cual se le asignara el valor de 40, El limite inferior comenzara en 1

a) Leer el arreglo
b) Calcular la suma de los valores del arreglo
c) Calcular la media de los valores

15-Mar-05

Actualización de un Vector

La operación de actualizar un vector puede constar de 3 elementos: Añadir, insertar o borrar elementos.

Añadir: Se refiere agregar datos a un vector o un nuevo elemento al final del vector. la única condición necesaria para esta operación consistira de la comprobación de espacio de memoria suficiente para el nuevo elemento (Que el vector no contenga todos los elementos suficientes con que fue definido al principio)

Ejm:

INSERTAR

La operación de insertar un elemento, consiste en introducir dicho elemento en el interior del vector. En este caso se necesita un desplazamiento previo hacia abajo para colocar el elemento nuevo en su posición relativa.

Ejm: Se tiene un agreglo coche con 9 elementos definidos inicialmente, de los cuales 7 se encuentran ocupados con marcas de automoviles en orden alfabetico y se desean insertar 2 nuevas marcas. OPEL CITROEN

Como OPEL esta comprendido entre LANCIA y RENAULT se deberan desplazar hacia abajo los elementos 5 y 6 que deberan ocupar la posc. 6 y 7 y posteriormente

Debe realizarse la operación con CITROEN que ocupara la posición 2.

a) Coches

1	Alfa Romeo
2	Fiat
3	Ford
4	Lancia
5	Renault
6	seat
7	
8	
9	

b) Insertar OPEL

1	Alfa Romeo
2	Fiat
3	Ford
4	Lancia
5	OPEL ←
6	Renault
7	seat
8	
9	

c) Insertar CITROEN

1	Alfa Romeo
2	citroen ←
3	Fiat
4	Ford
5	Lancia
6	OPEL
7	Renault
8	seat
9	

1° se recorren 7 - 4 = 3

Prog. 3

Vector Para Desplazar e Insertar Elementos

```
NUEVO = OPEL
Ult-Pos = 6
Pos = 5
FOR I = Ult-Pos to Pos
    coches (I+1) = coches (I)
END FOR
Coches (Pos) = Nuevo
Imprimir Vector
```

Si se desean realizar mas inserciones, habra q- incluir una estructura de decisión si entonces para preguntar si se van a realizar mas inserciones.

La operación de borrar un elemento del interior del vector, proboca el movimiento hacia arriba de los elementos inferiores a el para reorganizar el vector

Prog. 4 Realizar un borrado de registro y se recorran los registros

MARZO 17, 2005

ARRAYS DE VARIAS DIMENSIONES

Existen grupos de datos que son representadas mejor en forma de tabla o matriz con 2 o más subindices, ejemplos tipicos son:

- Tablas de distancias km entre ciudades
- Cuadros horarios de trenes o aviones
- Informes de ventas periodicas (mes/unidades Vend)
  (mes/ventas totales)

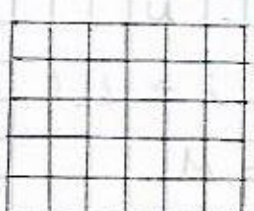
Se pueden definir tablas o matrices como array multidimensionales, cuyos elementos se pueden referenciar por 2 o más subindices, estos tipos de arrays se dividen en:

- Array Bidimensionales → 2 Dimensiones
- Array Multidimensionales → 3 o más Dimensiones

ARRAY BIDIMENSIONAL (TABLAS/MATRICES)

El array bidimensional se puede considerar como un vector de vectores. Es por eso consiguiente, un conjunto de elementos todos del mismo tipos en el cual el orden de los componentes es significativo y en el que se necesita especificar dos subindices para poder identificar cada elemento del array.

Si se visualiza un array unidimensional se puede considerar como una colomna de datos; un array bidimensional es un trozo de columnas ilustradas a continuación.

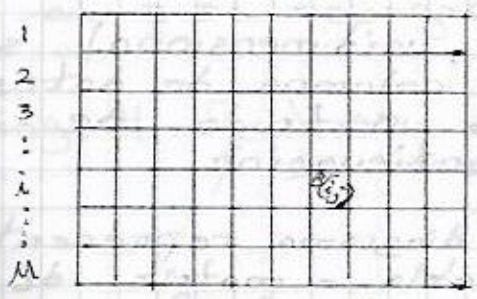


El diagrama representa una tabla o matriz de 30 elementos (5x6) con 5 filas y 6 columnas.

Como un vector de 30 elementos, cada uno de ellos tiene el mismo nombre, sin embargo, un subindice no es suficiente para especificar un elemento de un array bidimensional; por ejemplo si el nombre del array es M no se puede identificar M[3] ya que no sabemos si el 3er elemento de la 1ª fila o de la 1ª columna. Para esto un array bidimensional se referencia con 2 subindices el 1er subindice se refiere a la fila y el 2º se refiere a la columna. Por consiguiente: M[2,3] se refiere al elemento de la 2ª fila y la 3ª columna.

Un array bidimensional M, tambien denominado matriz o tabla se considera que tiene dos dimensiones. (Una dimensión por cada subindice) y necesita un valor por cada subindice y poder identificar un elemento individual. En notación estandar normalmente el primer subindice se refiere a la fila del array mientras que el segundo subíndice se refiere a la columna del array.

Ej. El arreglo $b_{i,j} = b[i,j]$ es el elemento b que ocupa la fila i y la columna j.



Elemento B[i,j] del Array B

Donde:

$i = 1, \ldots, M$

$j = 1, \ldots, N$

$1 \leq i \leq M$

$1 \leq j \leq N$

El array b se dice que tiene mxn elementos. Existen N elementos en cada fila y M elementos en cada columna (MxN).

Los array de dos dimensiones son muy frecuentes: las calificaciones de los estudiantes de una clase se almacenan en una tabla llamada notas de 2 dimensiones Notas[20,5] donde el 20 es el número de alumnos y 5 el número de asignatura. El valor del subindice i debe estar entre 1 y 20 y el de j entre 1 y 5

Los subindices pueden ser variables o expresiones numericas, ej:

Notas[M,4]

En general se considera que un array bidimensional comienza sus subindices en cero o en uno (según el lenguaje de programación que se trabaje) pero pueden tener limites seleccionados por el usuario durante la codificación del algoritmo.

En general, el array bidimensional B con su primer subindice, variando desde un limite inferior L (inferior: LOW) y un limite superior U (superior: UP); notación algoritmica.

$$B[L1:U1, L2:U2] = \{B[i,j]\}$$

donde:

$$L1 \leq i \leq U1 \quad ; \quad L2 \leq j \leq U2$$

(El número de elementos de una fila)

Ej. La matriz T representa una tabla de notaciones de saltos de altura (1er salto) donde las filas representan el nombre del atleta y las columnas las diferentes alturas saltadas por cada atleta. Los simbolos almacenados en la tabla son:

x = Salto valido

o = Salto nulo o no intentado

Otro ejemplo tipico de un arreglo bidimensional es un tablero de ajedrez. se puede representar cada posición o casilla mediante un array en el que cada elemento es una casilla y en el que su valor sera código representativo de cada figura del juego y los correspondientes numeros negativos para las piezas negras.

Ej:

	2.00	2.10	2.20	2.30	2.35	2.40
Garcia	x	o	x	x	x	o
Perez	o	x	x	o	x	o
Hernandez	o	o	o	o	o	o
Hinojosa	o	o	o	x	x	x

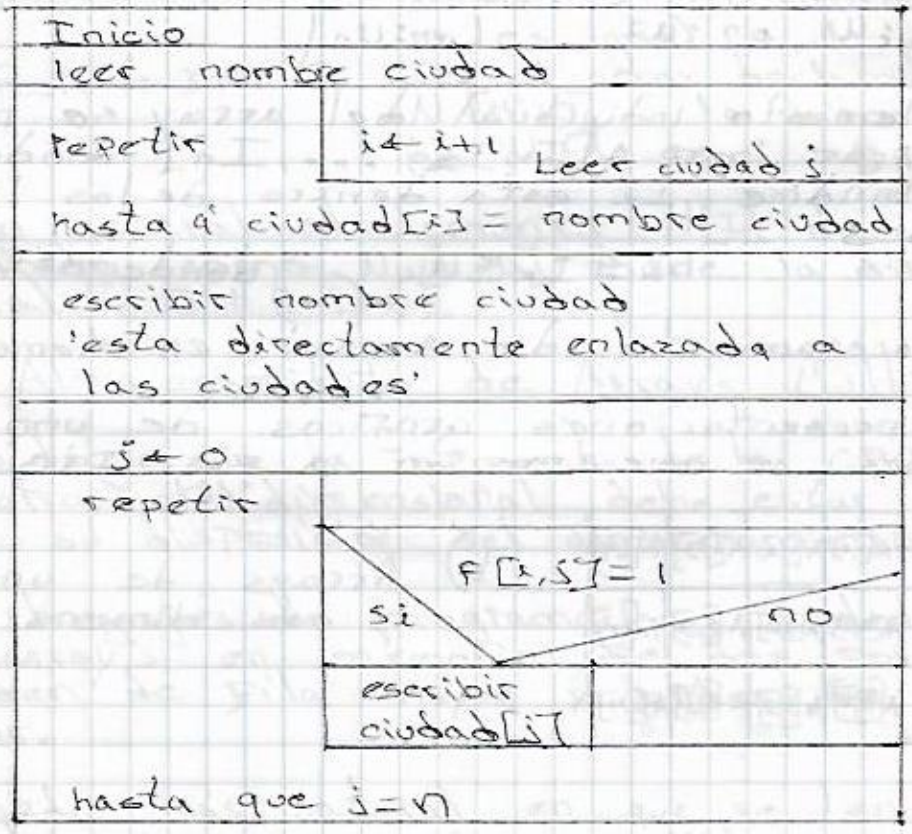
Ssaavedra@yahoo.com

Ote
Ejm: Se supone que se dispone de un mapa de ferrocarriles y los nombres de las estaciones (ciudades). Estan en vector ciudad.

El siguiente diagrama N-S lee un nombre de una ciudad e imprime los nombres de las ciudades con las que tiene colindancia.
El array f puede tener los siguientes valores
f[i,j] = 1 si existe enlace entre las ciudades i, j y f[i,j] = 0 si no existe enlace.

T. 6 Investigar

Nassi Sheiderman y 5 ej.



Tarea: Programa (5): De todos los estados ver con quien tienen colindancia. (32)

Un ejs. de A ndimen.

28 Marzo

Array Multidimensional

Un array puede ser definido de 3 o mas dimensiones.

Los conceptos de rango de subindices y No. de elelementos se pueden ampliar directamente desde array's de Una y dos dimensiones a estos array's de orden mas alto. En general un array de n dimensiones requiere que los valores de n subindices puedan ser especificados a fin de identificar un elemento individual del array. Si 1 componente de un array tiene n subindices, el array se dice que es solo de n-Dimensiones

El Array A de n-dimensiones se puede identificar como:

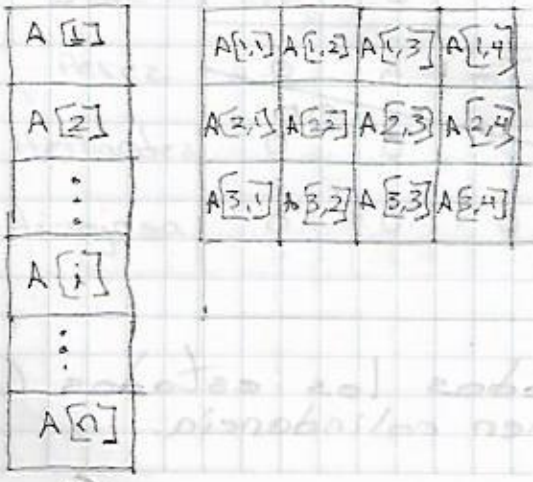
$$A[L1:U1, L2:U2, \ldots Ln:Un]$$

y un elemento individual del array se puede especificar por $A[I_1, I_2, \ldots, I_n]$ donde cada subindice I_K esta dentro de los limites adecuados

$$L_K \leq I_K \leq U_K \quad \text{donde } K=1,2,\ldots,n$$

Almacenamiento de Array's en Memoria.

Las representaciones graficas de uno a dos dimensiones se muestran en la sig. figura.



a)

b) Arrays de una y dos dimensiones

Almacenamiento de un vectos

El Alm. de 1 vec. en memoria se realiza en celdas por sucesiones secuenciales a si como en caso de un vectos A con subindice de rango de 1 → n

Si c/elemento de la array ocupa 's b/Le (8 bits) y B es la direccion inicial de la memoria central de la CPU (Posición o Dirección Base), la direccion inicial del elemento i-esimo seria:

$$B + [I-1) * S$$

Nota: si el limite inferior no es igual a 1, sonsidere-se el array declarado como N[4:10], la dirección inicial de N[6] es: $B+(6-4)*S$

En general el elemento N[I] de un array definido como N[L:U] tiene la dirección inicial $B + (I-L)*L$

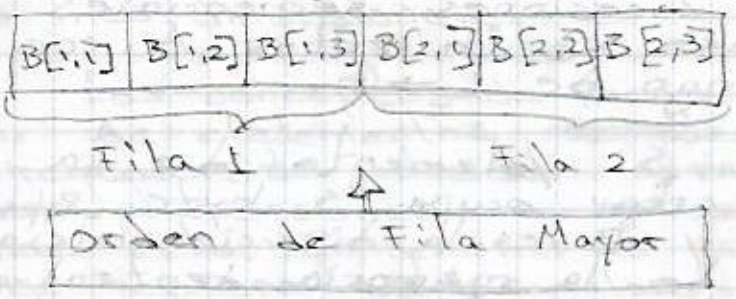
Almacenamiento de Array's Multidimensionales

Debido a que la memoria de la CPU es lineal un array mutidimensional debe estar linealizado para su disposición del almacenamiento.

Los lenguajes de programacion. pueden almacenar los array's en memoria de dos formas:
Orden de fila mayor y orden de columna mayor.

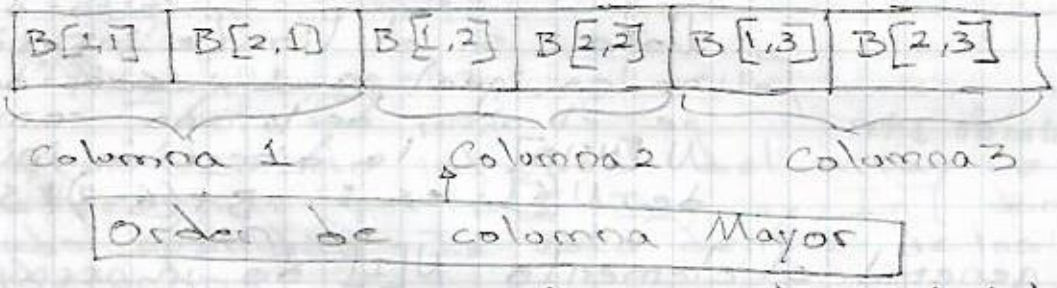
El medio más natural en que se len y almacenan array en la mayoria de compiladores es el denominado orden de fila mayor.

Por ejemplo, si un array es B[2,3] el orden de los elementos en la memoria sera la siguiente

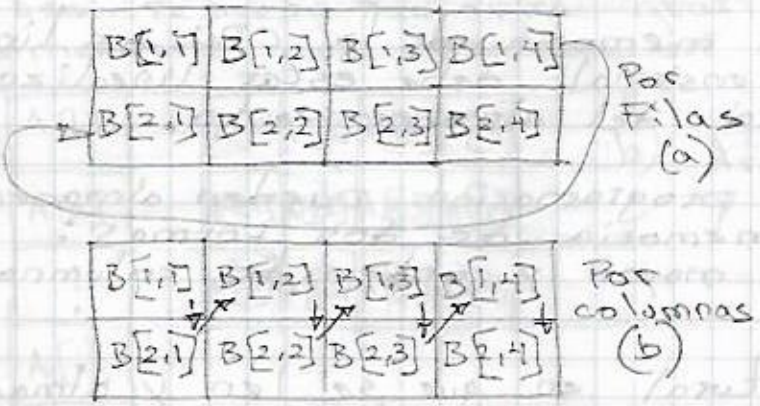


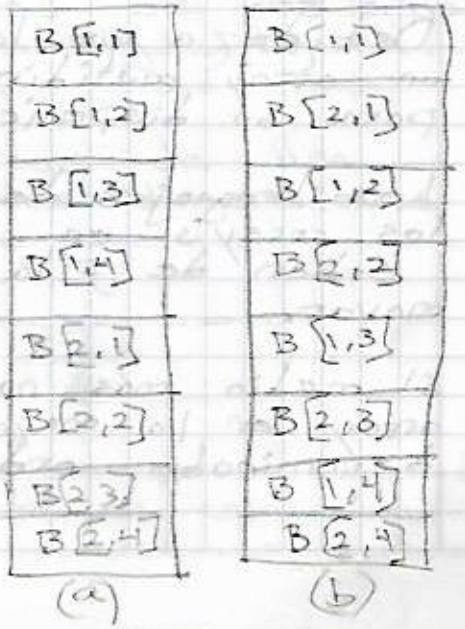
El lenguaje C almacena los elementos de la forma anterior (Orden de Fila Mayor).

El orden de columna mayor es la sig.



De modo general el compilador del lenguaje de alto nivel debe ser capaz de calcular con un indice [i,j] la posición del elemento correspondiente





Escribir un algoritmo q' permita calcular el cuadrado de los 100 # enteros y despues escribir una tabla que contenga 100 # dicuadrados.

T [1] = 1 * 1 = 1

T [2] = 2 * 2 = 4

T [3] = 3 * 3 = 9

=

=

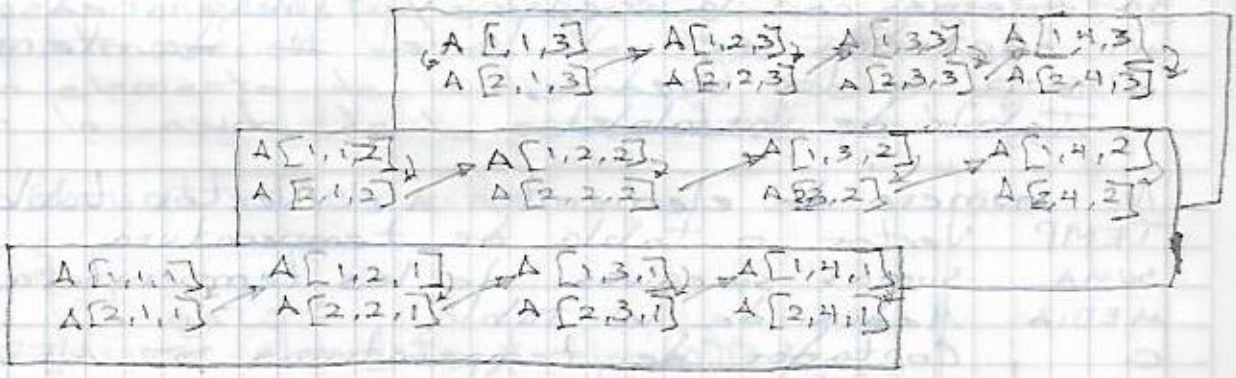
T [100] = 10000

29-Mar-05

En un array de orden fila mayor, cuyo subindices maximos sean m y n (m, filas; n, columnas) la posición ~~del~~ p [i, j] con relación al primer elemento es: $P = n(i-1) + j$ (del elemento).

Para calcular la dirección real del elemento [i, j] se añade p a la posición del primer elemento y se resta 1. La representación gráfica del almacenamiento de una tabla o matriz B [2, 4] y C [2, 4].

En caso de un Array de 3 dimensiones, supongamos un array tridimensional A [2, 4, 3]:



Almacenamiento de una matriz A [2, 4, 3] por columnas

```
Algoritmo cuadrados
tipo
    array [1...100] de entero: tabla
var
    tabla: T
    entero: I, C
inicio
    desde  I ← 1 hasta 100 hacer
           C ← I * I
           T[I] ← C
    Fin desde
    desde I ← 1 hasta 100 hacer
        escribir (T[I])
    fin_desde
Fin   printf("\n\t%d:%d", i, tabla[i]);
```

Ej. Se tienen n temperaturas se desea calcular su media y determinar entre todas ellas cuales son superiores o iguales a esa media.

En un primer momento, se leen los datos y se almacenan en un vector (Array unidimensional), TEMP[1:N]

A continuación se van realizando las sumas sucesivas para obtener la media.

Por último, con un bucle de lectura de la tabla se va comparando c/elemento de la misma con la media y luego mediante un contador se calcula el # de temp = o sup a la media.

Tabla de variables

N	numero de elementos del vector tabla
TEMP	Vector o tabla de temperatura
SUMA	Sumas sucesivas de las temperaturas
MEDIA	Media de la tabla
C	Contador de temperaturas >= MEDIA

Investigar (No se entrega)
- calculo de long. de cadena
- Comparación
- concatenación
- Extraccion de subcadenas
- busqueda de información.

1er Parcial hasta Cadenas El ~~jueves~~ Martes

A 7 lab compo el jueves

CAD 9
Luis Hdz
Alma, Josa, Vero
Erika
Sebastian
Van
Copias

```
Algoritmo temperaturas
tipo
    array [1...N] de Float : tabla
Var
    tabla : Temp
    Real  : MEDIA
    ENTERO: C, I, SUMA, N
inicio
   escribir("Numero de elementos?")
   Lee (N)
   suma = 0
   desde I = 1 hasta hacer
        Lee (N[I])
        SUMA = SUMA + N[I]
   Fin desde
   Media = suma/N
  C = 0
  desde I = 1 hasta N
    Si (N[I] >= MEDIA
      C = C+1
    FIN DEL SI
  FIN DESDE
  Escribir (C)
```

Pag 283-284

% Caracter

Crear una lista enlazada de elementos que almacenen datos de tipo entero
Un elemento de la lista se puede definir con la ayuda de la estructura siguiente

```
struct Elemento
{
int dato;
struct Elemento * siguiente;
};
typedef struct Elemento Nodo;
```

Exponer p' 2° Parcial

Unidad II

1. Pilas 25 y 26
2. Colas 25 y 26
3. Colas Dobles 28
4. Listas 2
5. Listas Circulares 3
6. Listas Doblemente ligadas } Equipo lunes 5 Mayo Beta y Alma

Unidad III 3er Parcial

1.- Concepto de Recursividad
2.- Algoritmos Recursivos
3.- Arboles
4.- Arboles Binarios → Programa Ejp. } Hacer 5 Programas 5 Mayo
5.- Grafos

IV Tecnicas de ordenamiento y Busqueda

Por Intercambio
Por Seleccion
Por Insercion
Por Burbuja
Shell → Beta
Quick Sort
Bin Sort → Alma
Radix Sort
Busqueda Secuencial
Busqueda Binaria

Con Programa → 2 Mayo
Programa con Menu
Lunes 23 examen
16 Mayo

Desarrollo de Tema

1 Teoria
Algoritmo { Declarar, Insertar, Eliminar

Programa

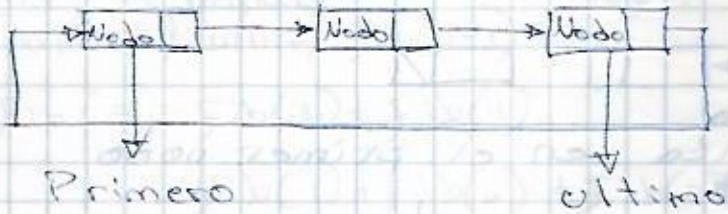
Estructuras
FIFO
First in
First out

Colas

C. Lineal:

Doble Cola (De los dos lados inserta y elimina

Cola Circular



F = 1er elem cola
A = n. elem cola
overflow = llena
underflow = vacia.

C. L. Algoritmo

Insertar

Si A = Maximo entonces
mensaje (overflow)
en caso contrario
A ← A + 1
cola[A] valor

Se Agrega al ultimo

Extraer

Si A = F entonces
mensaje (underflow)
en contrario
F ← F + 1
F ← cola(A)

Se quita el Primero

En Circulares

Si (F + 1) = (A = max) entonces
mensaje (overflow)
en caso contrario
inicio
si A = max entonces
A ← 1

cola [A] ← Valor
en caso contrario
A ← A+1

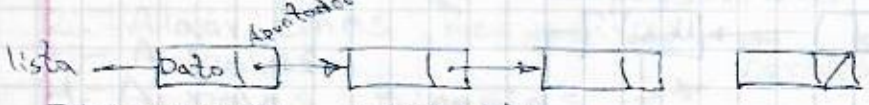
cola [A] ← Valor
si F=0 entonces
F ← 1
fin

El que insertamos se vuelve primero

En extracción
El primero q' entra primero q' sale

Listas

Conjunto de nodos que cambia rapidamente



Diferencia con cola
siempre se inserta en el primer nodo
Lita: Estatica
Lista ligada: es por puntero

Operaciones
Insertar
Eliminar
Recorrer
Comprobar vacia

2-Mayo-05

Listas

Conjunto de nodos, tales nodos se componen de 2 partes.
Campo información
Campo Enlace

11-Mayo-05

Recursividad vs Reiterativas

Ejm: for while

Se utiliza para funciones q'e se llaman así mismas.

Ejm: Factorial

$n! = n(n-1)(n-2)\ldots 1$

$8! = 8 \cdot 7!$

$8 \cdot 7 \cdot 6!$

$8 \cdot 7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$

Def: $1! = 1$ $\quad 0! = 1$

Ejm: Fibonacci

$fib(n) = fib(n-2) + fib(n-1)$

$fib(5) = fib(3) + fib(4)$

$fib(1) + fib(2) + fib(2) + fib(3)$

$fib(0) = 0$

$fib(1) = 1$

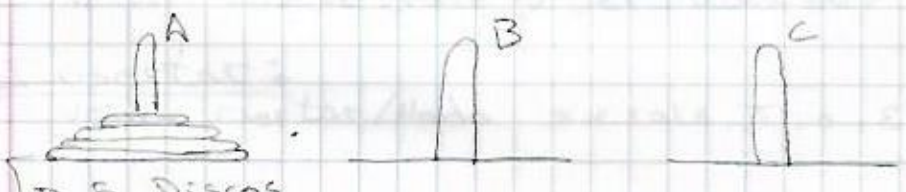
$fib(2) = fib(0) + fib(1) = 0 + 1 = 1$

$fib(3) = fib(1) + fib(2) = 1 + 1 = 2$

$fib(4) = fib(2) + fib(3) = 1 + 2 = 3$

$fib(5) = fib(3) + fib(4) = 2 + 3 = 5$

Ejm: Bcahama



↳ 5 Discos

$n = \#$ Discos

Movimientos $2^n - 1$

Se trata de:

Pasar los discos de A a C

Recursividad, Arboles, Grafos

```
int fact (int n) {
 int x, y;
 if (n<0) {
 printf ("%s", "Parametro negativo");
  exit(1);
 }
 if (n==0)
    return (1);
 x=n-1;
 y=fact(x);
 return (n*y);
 }
```

```
inicio
si n=0 entonces
devolver (1)
si_no devolver
(n*factorial(n-1))
Fin_si
Fin_función
```

Arbol.

Busqueda:

```
Nodo *buscar (Nodo *raiz, Tipo elemento buscado)
{
 if (raiz)
 return;
 else if (buscado == raiz->Dato)
  return raiz;
  else if (buscado < raiz -> dato)
  return buscar (raiz -> Izq, buscado);
  else
  return buscar (raiz -> Der, buscado)
```

Insertar:

```
void insertar (Nodo **raiz, Tipo Elemento
```