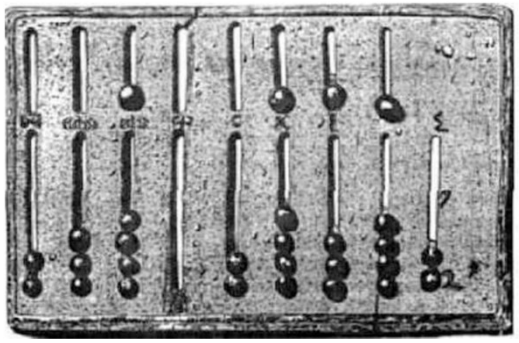
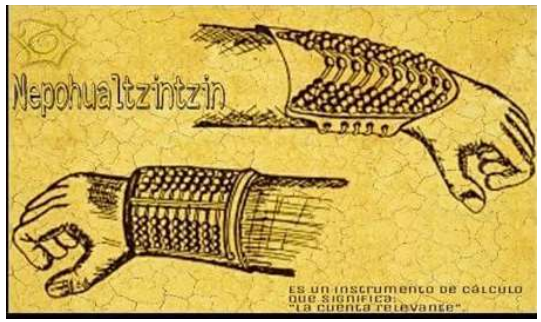
TEMA 1.
Estructura de datos

OBJETIVO: El alumno conocerá los antecedentes de la representación de los datos y las estructuras de datos lineales fundamentales.

1.1 Antecedentes y Generalidades

- La historia de la computación tiene sus orígenes en el inicio mismo de la civilización.
- La búsqueda de nuevos métodos para realizar cálculos ha evolucionado desde la antigüedad.





1.1 Antecedentes y Generalidades

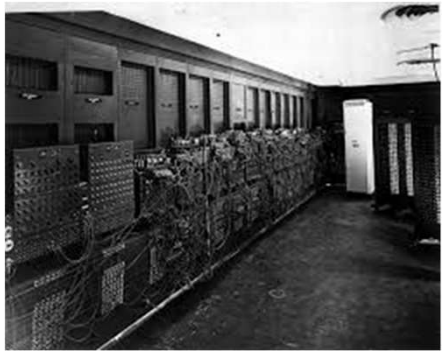
- Una computadora digital es un dispositivo electrónico, utilizado para procesar información y obtener resultados, capaz de ejecutar cálculos a velocidades considerablemente más rápidas de lo que pueden hacerlo los seres humanos.
- Componentes físicos:
  - CPU
  - DISPOSITIVOS E/S
  - MEMORIA

1.1 Antecedentes y Generalidades

- La computación como disciplina nace a principios de 1940 con base en la teoría de algoritmos, la lógica matemática y la aparición del concepto de **programa**.
- Un programa es un conjunto de instrucciones que una vez ejecutadas realizarán una o varias tareas en una computadora.

1.1 Antecedentes y Generalidades

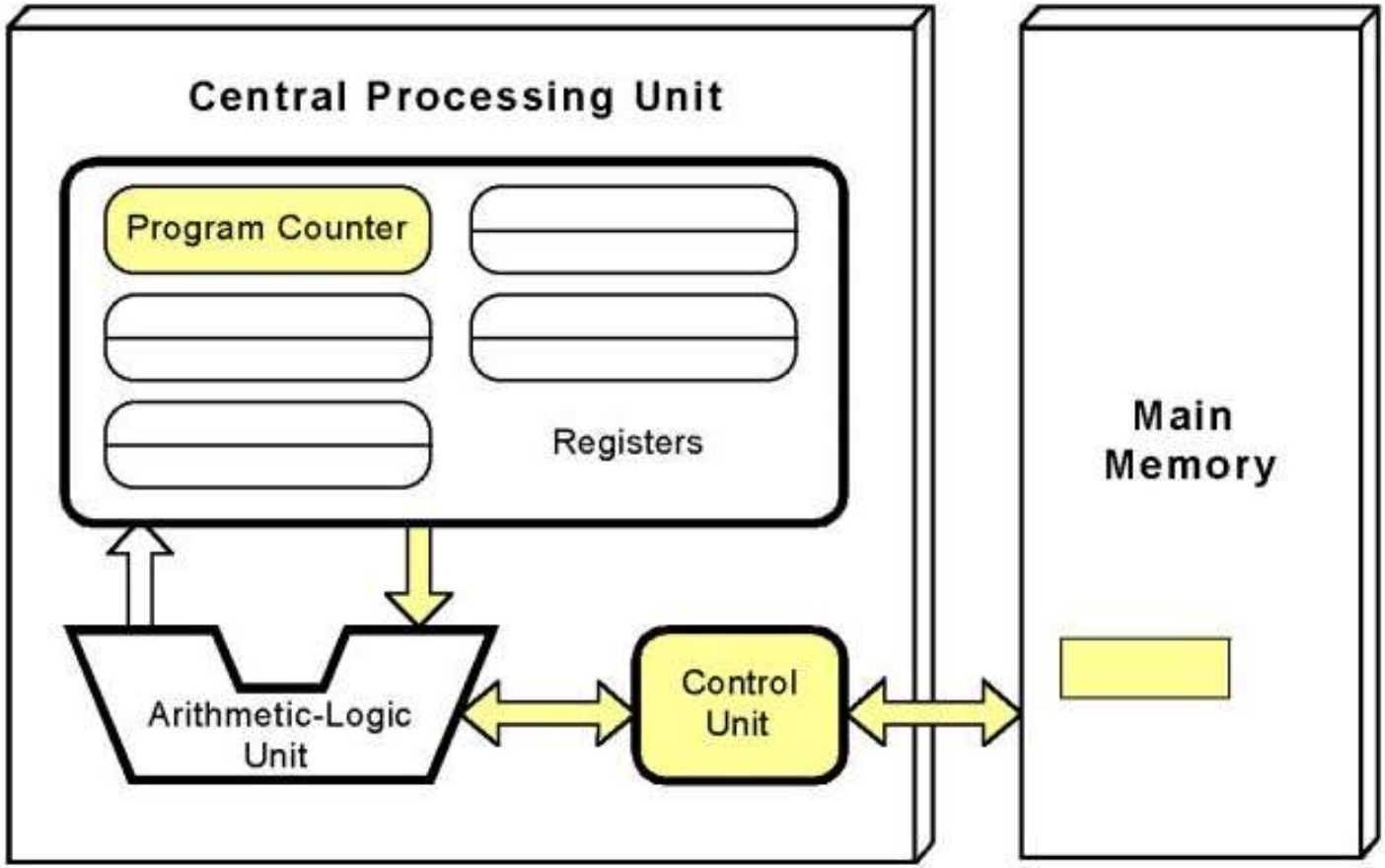
- John von Neumann propuso un esquema con una computadora con una memoria para almacenar datos y programas.
- Este modelo se conoce como arquitectura von Neumann y es el modelo que se sigue utilizando en las computadoras actuales.



https://histinf.blogs.upv.es/2011/12/05/proyecto-eniac/

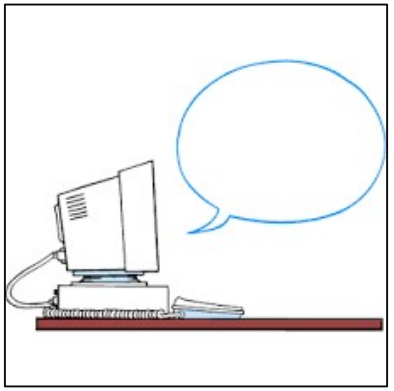
https://es.wikipedia.org/wiki/John_von_Neumann

1.1 Antecedentes y Generalidades



1.1 Antecedentes y Generalidades

- Un **lenguaje de programación** es un lenguaje artificial diseñado para expresar instrucciones que pueden ser llevadas a cabo por máquinas como las computadoras.
  - Lenguajes de alto nivel
  - Lenguajes de bajo nivel



1.1 Antecedentes y Generalidades

Clasificaciones de Lenguajes de Programación

- Manera de Ejecutarse:
  - ✓**Compilados**. Código fuente > código objeto > Programa ejecutable.
  - ✓**Interpretados**. Un programa ejecuta las instrucciones de manera directa.

1.1 Antecedentes y Generalidades

Clasificaciones de Lenguajes de Programación

- Manera de Abordar la tarea:
  - ✓**Imperativos**. Indican cómo hay que hacer la tarea.
  - ✓**Declarativos**. Indican qué tarea hay que hacer, sin embargo no indica cómo realizarla

1.1 Antecedentes y Generalidades

Clasificaciones de Lenguajes de Programación

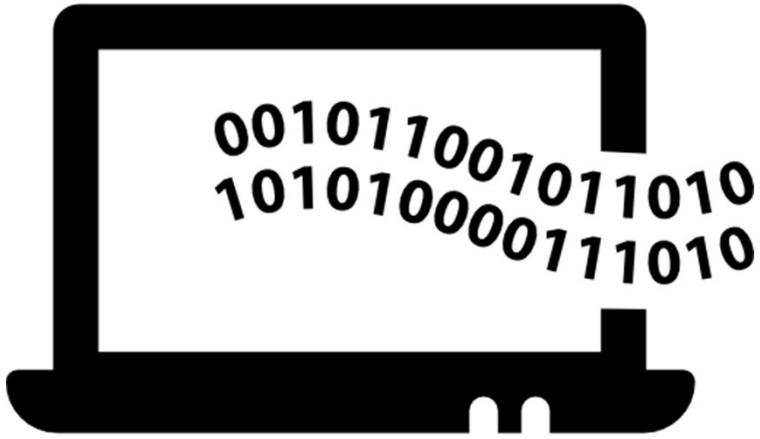
- De acuerdo al paradigma de Programación:
  - ✓ **Procedural.** Divide el problema en partes más pequeñas.
  - ✓ **Orientado a Objetos**. Sistema de clases y objetos siguiendo el mundo real.
  - ✓ **Funcional**. Evaluación de funciones.
  - ✓ **Lógica.** Tareas expresadas empleando lógica formal matemática.



1.2 Representación de datos

1.2 Representación de datos

- La información que se procesa en una computadora tiene diferentes tipos de datos.
- Para el manejo adecuado de datos en una computadora digital, se implementó una representación uniforme de los datos.
- Esta representación se conoce como **Patrón de Bits**.



1.2 Representación de datos

- **BIT**: Es la unidad mínima de almacenamiento en las computadoras y representa dos estados.

- **BYTE**: Es un patrón de 8 bits mediante el cual se mide el tamaño de una memoria o de otros dispositivos de almacenamiento.

1.2 Representación de datos

- ¿Cuántos bits se necesitan en un patrón para representar un símbolo?

1.2 Representación de datos

Longitud del patrón de bits	Símbolos representados
1	2
2	4
3	8
4	16
…	…
7	128
8	256
…	…
16	65536

1.2 Representación de datos

- Existen diversas cadenas de bits que se establecen como secuencias de patrones para representar símbolos de texto, numéricos, etc.
- El proceso de representar los símbolos se conoce como **codificación**.

1.2.1 Representación de texto

- Los diferentes estándares para representar texto han evolucionado conforme a la necesidad de representar una mayor cantidad de símbolos.

1.2.1 Representación de texto

Código ASCII

- Código Norteamericano de Estándares para Intercambio de Información (American Standard Code for Information Interchange, 1967).
- Este código utiliza siete bits para cada símbolo.
- **ASCII extendido**: Para hacer que el tamaño de cada patrón sea de 1 byte (8 bits), a los patrones de bits ASCII se les aumenta un cero más a la izquierda.
- Cada patrón cabe fácilmente en un byte de memoria.

Código ASCII

USASCII code chart

b_7 b_6 b_5 →	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
Bits b_4 b_3 b_2 b_1 ↓ Row / Column →	0	1	2	3	4	5	6	7
0 0 0 0	0	NUL	DLE	SP	0	@	P	`	p
0 0 0 1	1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	2	STX	DC2	"	2	B	R	b	r
0 0 1 1	3	ETX	DC3	#	3	C	S	c	s
0 1 0 0	4	EOT	DC4	$	4	D	T	d	t
0 1 0 1	5	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	6	ACK	SYN	&	6	F	V	f	v
0 1 1 1	7	BEL	ETB	'	7	G	W	g	w
1 0 0 0	8	BS	CAN	(	8	H	X	h	x
1 0 0 1	9	HT	EM	)	9	I	Y	i	y
1 0 1 0	10	LF	SUB	*	:	J	Z	j	z
1 0 1 1	11	VT	ESC	+	;	K	[	k	{
1 1 0 0	12	FF	FS	,	<	L	\	l	|
1 1 0 1	13	CR	GS	-	=	M	]	m	}
1 1 1 0	14	SO	RS	.	>	N	^	n	~
1 1 1 1	15	SI	US	/	?	O	_	o	DEL

Código ASCII Extendido

128	Ç	144	É	160	á	176	▒	193	┴	209	╤	225	ß	241	±
129	ü	145	æ	161	í	177	▒	194	┬	210	╥	226	Γ	242	≥
130	é	146	Æ	162	ó	178	▓	195	├	211	╙	227	π	243	≤
131	â	147	ô	163	ú	179	│	196	─	212	╘	228	Σ	244	⌠
132	ä	148	ö	164	ñ	180	┤	197	┼	213	╒	229	σ	245	⌡
133	à	149	ò	165	Ñ	181	╡	198	╞	214	╓	230	µ	246	÷
134	å	150	û	166	ª	182	╢	199	╟	215	╫	231	τ	247	≈
135	ç	151	ù	167	º	183	╖	200	╚	216	╪	232	Φ	248	°
136	ê	152	_	168	¿	184	╕	201	╔	217	┘	233	Θ	249	·
137	ë	153	Ö	169	_	185	╣	202	╩	218	┌	234	Ω	250	·
138	è	154	Ü	170	¬	186	║	203	╦	219	▉	235	δ	251	√
139	ï	156	£	171	½	187	╗	204	╠	220	▄	236	∞	252	_
140	î	157	¥	172	¼	188	╝	205	═	221	▌	237	ϕ	253	²
141	ì	158	_	173	¡	189	╜	206	╬	222	▐	238	ε	254	■
142	Ä	159	ƒ	174	«	190	╛	207	╧	223	▀	239	∩	255	
143	Å	192	└	175	»	191	┐	208	╨	224	α	240	≡		

Codificación EBCDIC

- Extended Binary Coded Decimal Interchange Code (EBCDIC).
- Representa caracteres alfanuméricos, controles y signos de puntuación.
- Cada carácter está compuesto por 8 bits, define un total de 256 caracteres.
- Es un código estándar usado por computadoras mainframe IBM.

Codificación EBCDIC

Bits 3210	7654 0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	NUL	DLE	DS	IRS	SP	&	-	ø	Ø	°	µ	¢	ä	å	E	0
0001	SOH	DC1	SOS	ITB	RSP	'	/	\	a	j	ü	£	A	J	÷	1
0010	STX	DC2	FS	SYN	á	é	A	E	b	k	s	¥	B	K	S	2
0011	ETX	TM	WUS	IR	{	è	#	E	c	l	t	·	C	L	T	3
0100	PF	RES	BYP	PN	à	è	A	E	d	m	u	©	D	M	U	4
0101	HT	NL	LF	RS	á	ì	A	I	e	n	v	[	E	N	V	5
0110	LC	BS	ETB	UC	ã	ì	A	I	f	o	w	¶	F	O	W	6
0111	DEL	IL	ESC	EOT	}	I	$	I	g	p	x	¼	G	P	X	7
1000	GE	CAN	SA	SBS	ç	ì	Ç	I	h	q	y	½	H	Q	Y	8
1001	SPS	EM	SFE	IT	ñ	ß	N	è	i	r	z	¾	I	R	Z	9
1010	SMM	CC	SM	RFF	§	¤	ö	:	«	ª	¡	-	SHY	¹	²	³
1011	VT	CU1	CU2	CU3	.	Á	,	A	»	º	¿	]	ò	û	O	U
1100	FF	IFS	MFA	DC4	<	*	%	O	ð	æ	Ð	—	¦	~	@	U
1101	CR	IGS	ENQ	NAK	(	)	_	'	ý	¸	Ý	]	ò	ù	O	U
1110	SO	IRS	ACK		+	;	>	=	Þ	Æ	þ	`	ó	ú	O	U
1111	SI	IUS	BEL	SUB	!	^	?	"	±	]	®	×	õ	ÿ	O	EO

UNICODE

- Este estándar es mantenido por el Unicode Technical Committee (UTC) mantiene estrecha relación con ISO/IEC, alcanzando el acuerdo de sincronizar sus estándares que contienen los mismos caracteres y puntos de código.
- En la actualidad soporta tres formatos para representar millones de caracteres.
  - UTF-8
  - UTF-16
  - UTF-32

  https://unicode-table.com/es/

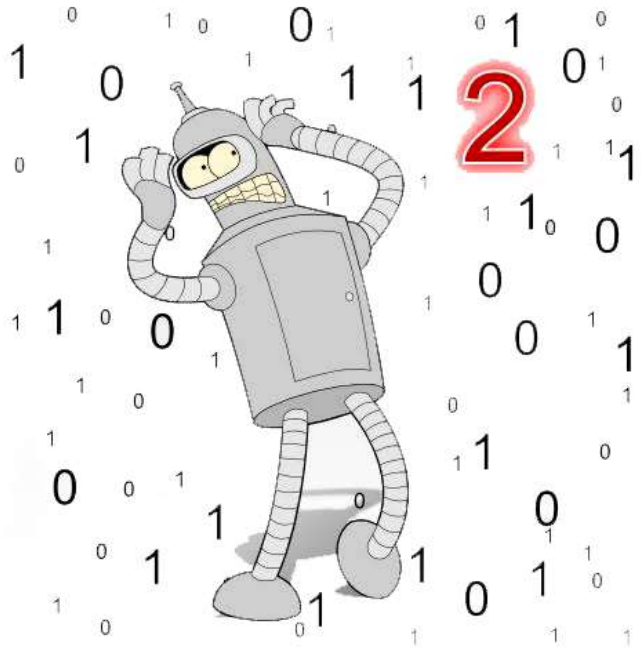
1.2.2 Representación de números enteros

- Los enteros de la computadora se representan mediante agrupaciones de dígitos binarios, se tienen las siguientes formas de representación:
  - SIN SIGNO
  - CON SIGNO
    - ✓SIGNO Y MAGNITUD
    - ✓COMPLEMENTO A 1
    - ✓COMPLEMENTO A 2

1.2.2 Representación de números enteros

Enteros sin signo

- También conocido como **binario convencional o binario puro**, permite representar enteros positivos y el cero.
- El intervalo de números que puede representar depende del número de bits disponibles.



1.2.2 Representación de números enteros

Enteros con signo y magnitud

- Se utiliza de la misma manera que la representación convencional, la diferencia radica en que en este tipo de representación se destina el primer bit para indicar el signo del número:
  - ✓ 0, número positivo;
  - ✓ 1, número negativo.
- Debido a que se utiliza un bit para identificar el signo, se reduce la cantidad de valores que se pueden representar.

1.2.3 Números reales

- Debido a que las computadoras tienen un número finito de bits. No pueden almacenar los números reales en forma exacta, de forma similar a lo que ocurre con los números irracionales (pi, e, etc.) por lo tanto se utilizan aproximaciones.
  - ✓Punto fijo
  - ✓Punto flotante

1.2.3 Números reales

Representación de Punto fijo

- Consiste en destinar una cantidad de dígitos para la parte entera y el resto para la parte fraccionaria.
- La cantidad de dígitos destinados a la parte fraccionaria indica la posición del punto dentro del número.
- La limitante es que dependiendo de los dígitos que se asignen a la parte fraccionaria disminuye el número máximo para la parte entera.