Bioinformática y análisis de datos ómicos

Tema 1: Introducción al Uso de Herramientas Bioinformáticas

Ignacio Varela Egocheaga
Departamento de Biología Molecular

Tema 1: Introducción al uso de herramientas bioinformáticas

1.1 Principios básicos del funcionamiento de un ordenador.

1.2 Codificación de distintos tipos de archivos.

1.3 Sistemas operativos. Estructuras de archivos.

1.4 Ejecución de comandos y programas desde la terminal de comandos.

Tema 1: Introducción al uso de herramientas bioinformáticas

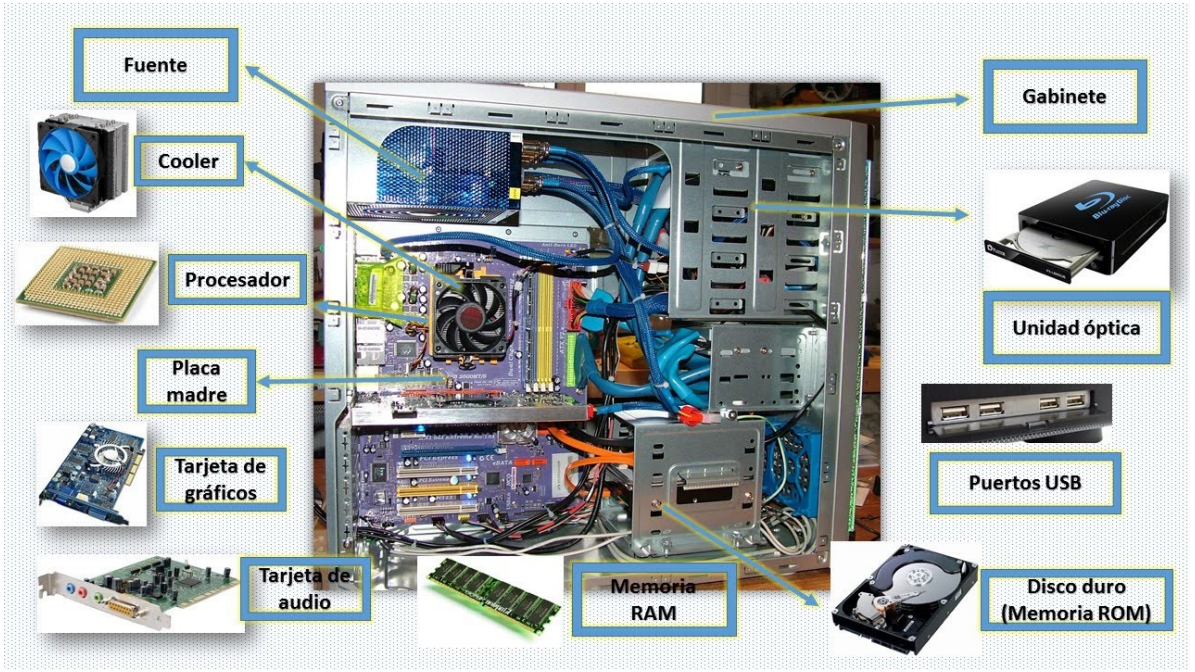
1.1 Principios básicos del funcionamiento de un ordenador.

1.2 Codificación de distintos tipos de archivos.

1.3 Sistemas operativos. Estructuras de archivos.

1.4 Ejecución de comandos y programas desde la terminal de comandos.

Componentes y funcionamiento de un ordenador



Componentes principales del procesamiento:

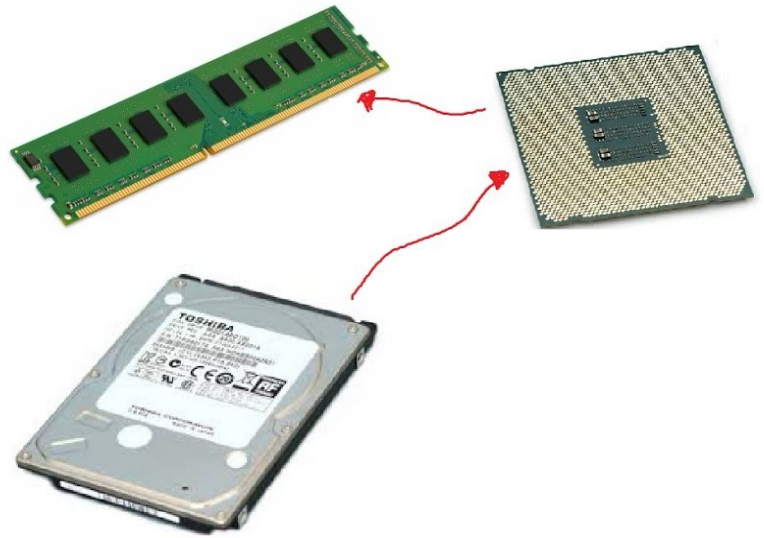
Memoria ROM

Memoria RAM

Procesador (CPUs)

Memoria RAM

Almacenamiento temporal rápidamente accesible



CPUs

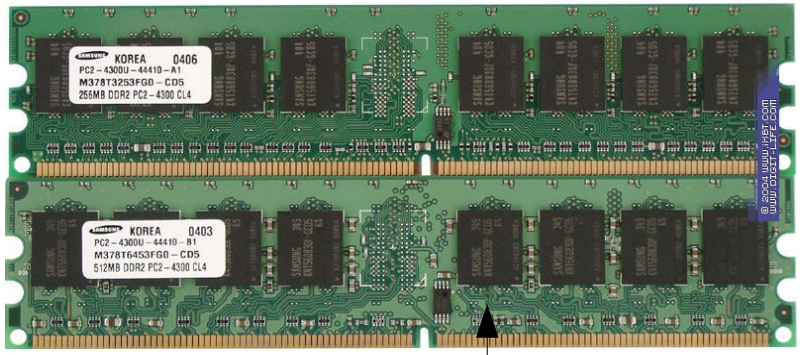
Operaciones que modifican la información.

Memoria ROM

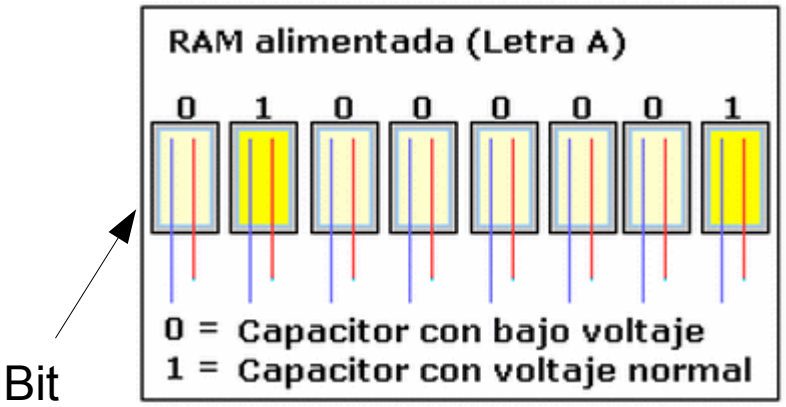
Almacenamiento más permanente

Almacenamiento de la información

Chips modernos



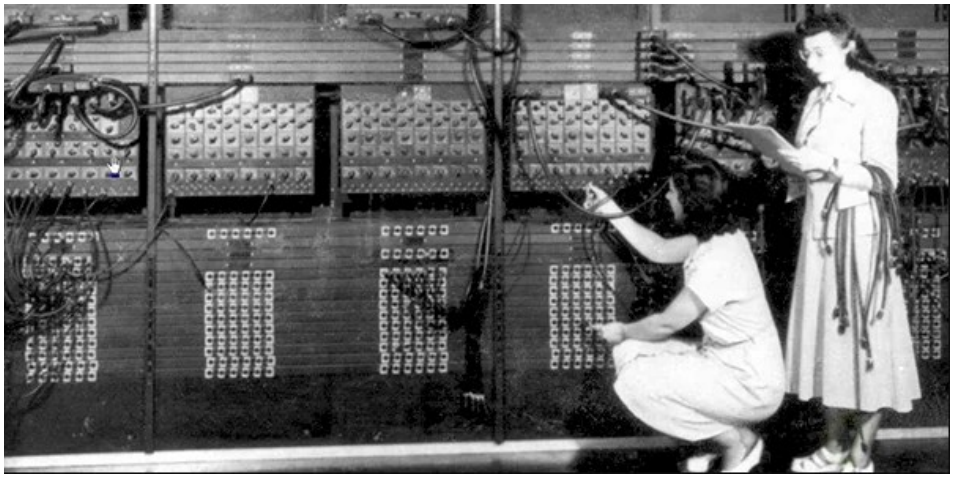
Chip de memoria



Bit

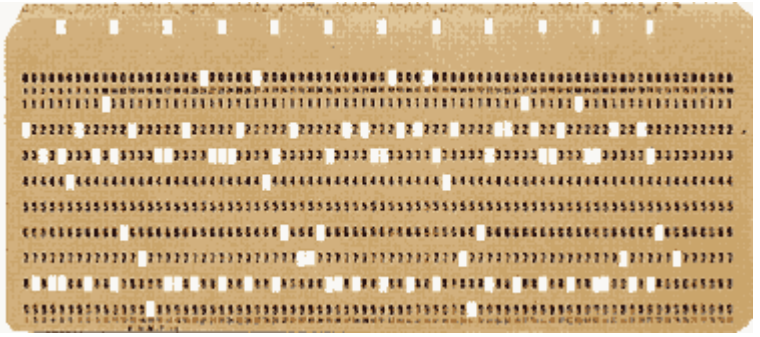
1 byte = 8 bits

Primeros computadores

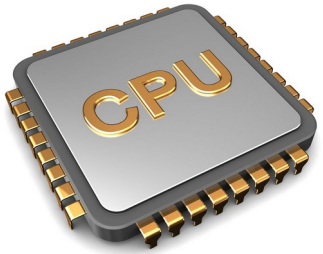


1940s

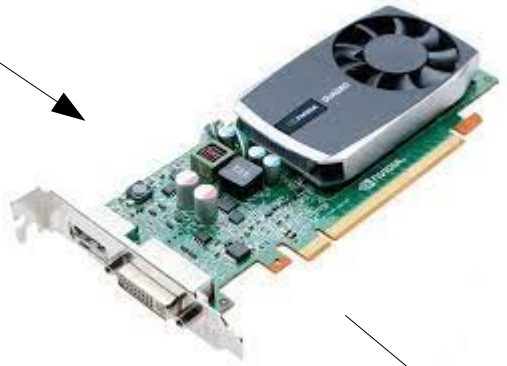
1960s



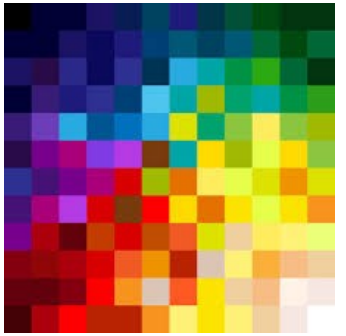
Componentes periféricos

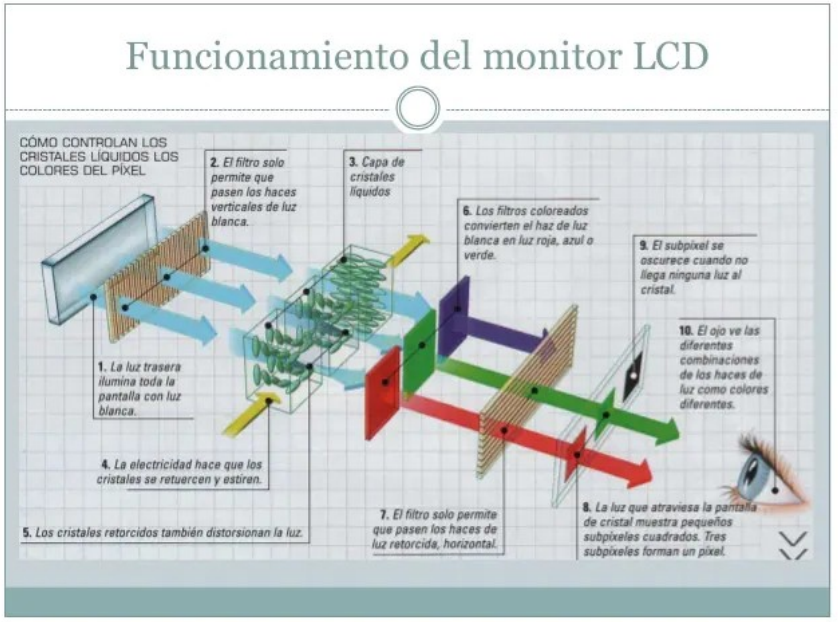


Procesador



Tarjeta de video





Monitor

Tema 1: Introducción al uso de herramientas bioinformáticas

1.1 Principios básicos del funcionamiento de un ordenador.

1.2 Codificación de distintos tipos de archivos.

1.3 Sistemas operativos. Estructuras de archivos.

1.4 Ejecución de comandos y programas desde la terminal de comandos.

Codificación de información digital

Corriente eléctrica → CODIFICACIÓN → ***INFORMACIÓN***

Codificación numérica

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

Codificación texto ASCII

1 byte → 1 caracter (256 caracteres)

Caracteres ASCII imprimibles

DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo
32	20h	espacio	64	40h	@	96	60h	`
33	21h	!	65	41h	A	97	61h	a
34	22h	"	66	42h	B	98	62h	b
35	23h	#	67	43h	C	99	63h	c
36	24h	$	68	44h	D	100	64h	d
37	25h	%	69	45h	E	101	65h	e
38	26h	&	70	46h	F	102	66h	f
39	27h	'	71	47h	G	103	67h	g
40	28h	(	72	48h	H	104	68h	h
41	29h	)	73	49h	I	105	69h	i
42	2Ah	*	74	4Ah	J	106	6Ah	j
43	2Bh	+	75	4Bh	K	107	6Bh	k
44	2Ch	,	76	4Ch	L	108	6Ch	l
45	2Dh	-	77	4Dh	M	109	6Dh	m
46	2Eh	.	78	4Eh	N	110	6Eh	n
47	2Fh	/	79	4Fh	O	111	6Fh	o
48	30h	0	80	50h	P	112	70h	p
49	31h	1	81	51h	Q	113	71h	q
50	32h	2	82	52h	R	114	72h	r
51	33h	3	83	53h	S	115	73h	s
52	34h	4	84	54h	T	116	74h	t
53	35h	5	85	55h	U	117	75h	u
54	36h	6	86	56h	V	118	76h	v
55	37h	7	87	57h	W	119	77h	w
56	38h	8	88	58h	X	120	78h	x
57	39h	9	89	59h	Y	121	79h	y
58	3Ah	:	90	5Ah	Z	122	7Ah	z
59	3Bh	;	91	5Bh	[	123	7Bh	{
60	3Ch	<	92	5Ch	\	124	7Ch	|
61	3Dh	=	93	5Dh	]	125	7Dh	}
62	3Eh	>	94	5Eh	^	126	7Eh	~
63	3Fh	?	95	5Fh	_	elCodigoASCII.com.ar		

UNICODE

UTF8

UTF16

...

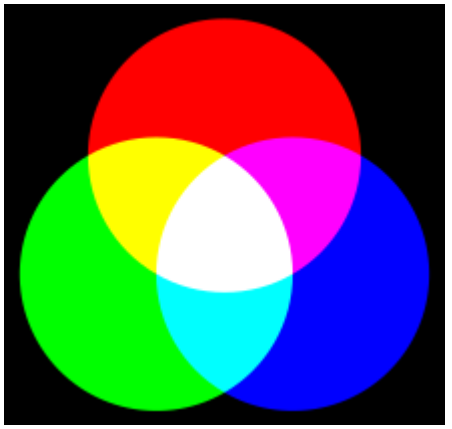
Codificación de información digital

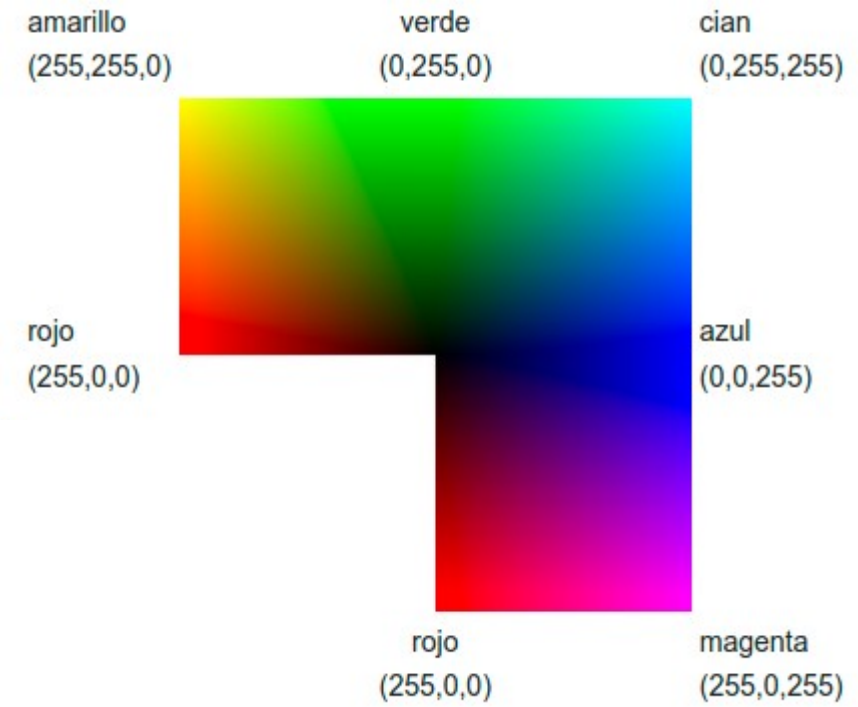
Codificación de imágenes

$256^3 \sim 16 \times 10^6$ posibles combinaciones/colores

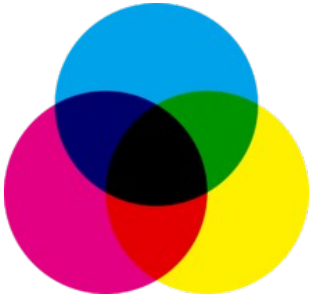
1 Byte 1 Byte 1 Byte

Codificación RGB: Red + Green + Blue



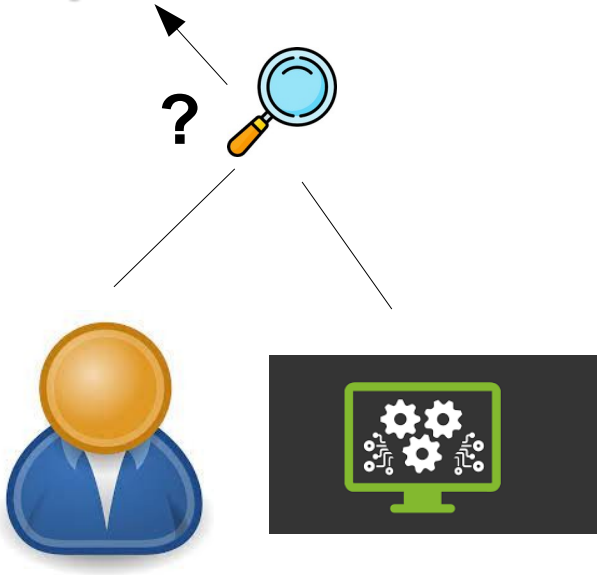


Codificacion CMYK: Cian + Magenta + Yellow + Key (black)



Importancia de la extensión del archivo

Distinta codificación dependiendo del tipo de archivo



La extensión no modifica el archivo

El código **MD5** es un código binario calculado a partir de los bits de un archivo y que identifica de manera única el contenido de un archivo

Tema 1: Introducción al uso de herramientas bioinformáticas

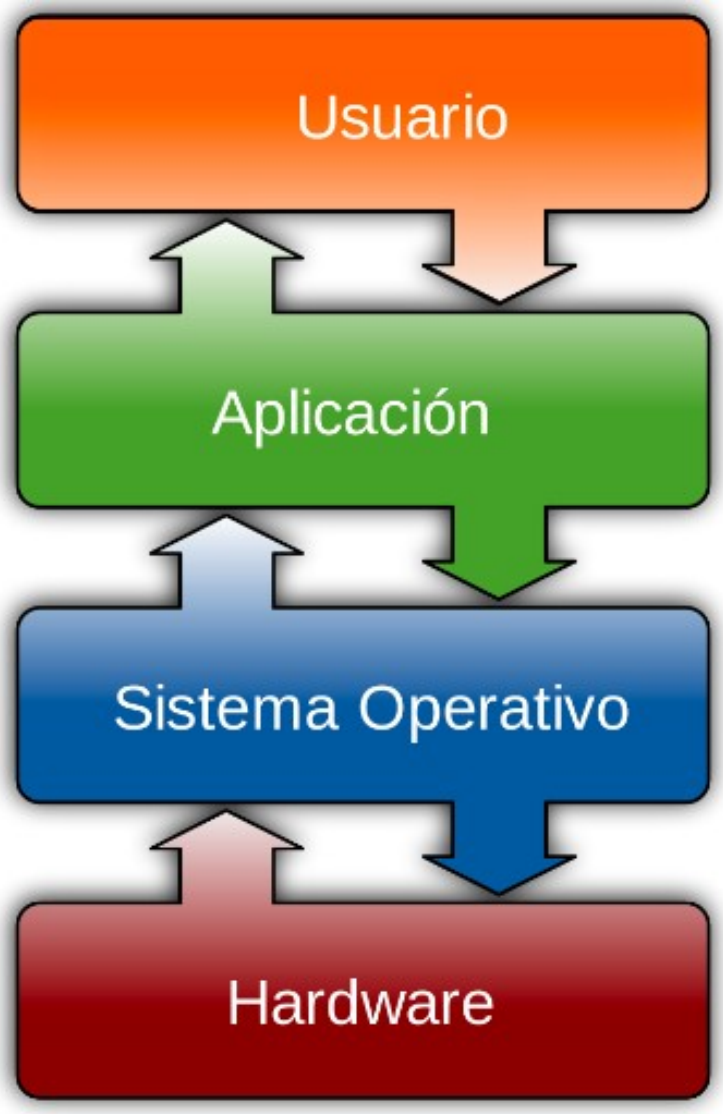
1.1 Principios básicos del funcionamiento de un ordenador.

1.2 Codificación de distintos tipos de archivos.

1.3 Sistemas operativos. Estructuras de archivos.

1.4 Ejecución de comandos y programas desde la terminal de comandos.

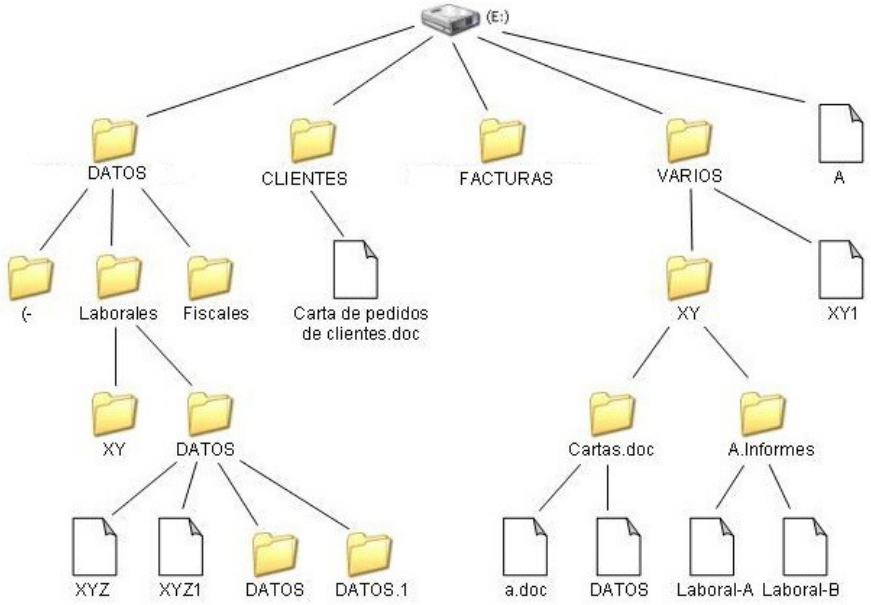
Sistema operativo



El sistema operativo: Director de operaciones, procesos y recursos del ordenador

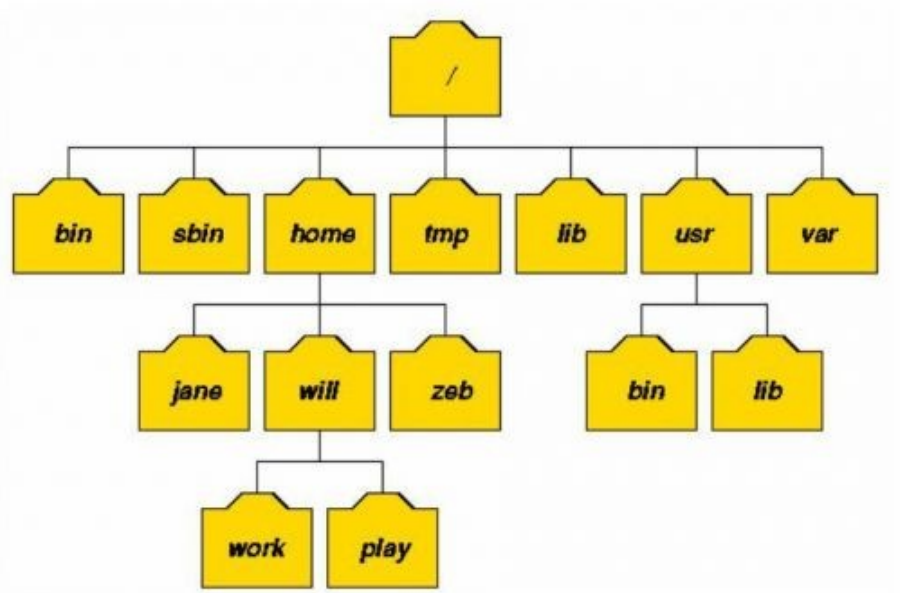
Sistemas de archivos

WINDOWS



C:\DATOS\Laborales\DATOS\XYZ

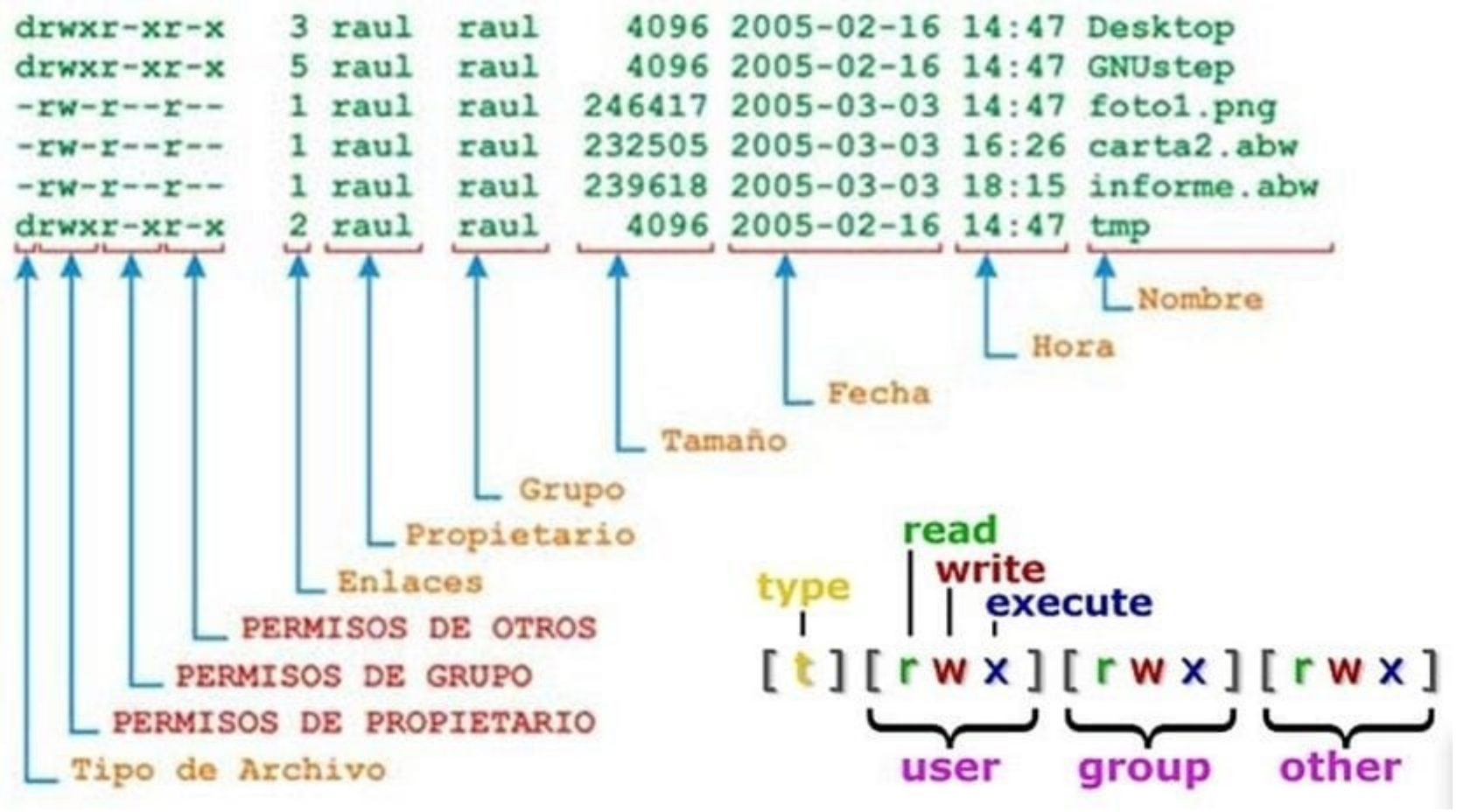
UNIX



User: ~$ /home/will/work

pwd → Directorio actual.
mkdir → Crear directorio
cd → Cambiar directorio
cp → copiar archivos/directorios
mv → mover/renombrar archivos/directorios
rm → borrar archivos/directorios

Permisos de archivo (UNIX)



chown → Cambio de propietario/grupo
chmod → Cambio de permisos en archivos/directorios
ln → crear enlaces a archivos.
sudo → ejecutar órdenes como administrador

Tema 1: Introducción al uso de herramientas bioinformáticas

1.1 Principios básicos del funcionamiento de un ordenador.

1.2 Codificación de distintos tipos de archivos.

1.3 Sistemas operativos. Estructuras de archivos.

1.4 Ejecución de comandos y programas desde la terminal de comandos.

Manera de ejecutar programas

Si no se especifica la localización del archivo, el sistema operativo busca primero el archivo del programa en el directorio actual y, si no lo encuentra, en las rutas especificadas en la variable de entorno $PATH. Por defecto estas rutas en linux son /usr/bin y /usr/local/bin. Ésta y otras variables de entorno se pueden modificar (ver anexo: variables de entorno).

Programas compilados en lenguaje nativo

Una gran cantidad de herramientas informáticas, como las escritas en lenguaje C y derivados (C+, C++), están compilados directamente en nuestro sistema operativo generado un archivo ejecutable directamente por el sistema operativo. Estos programas se ejecutan directamente:

```
User: ~$ cp
```

```
User: ~$ fdisk
```

```
User: ~$ bwa
```

Programas pre-compilados

Muchas herramientas bioinformáticas usan lenguajes que generan archivos pre-compilados que necesitan un intérprete para ejecutarlos. Estos programas se ejecutan llamando al intérprete y pasando el archivo como parámetro:

```
User: ~$ java -jar file.jar
```

```
User: ~$ python file.py
```

Documentación de apoyo

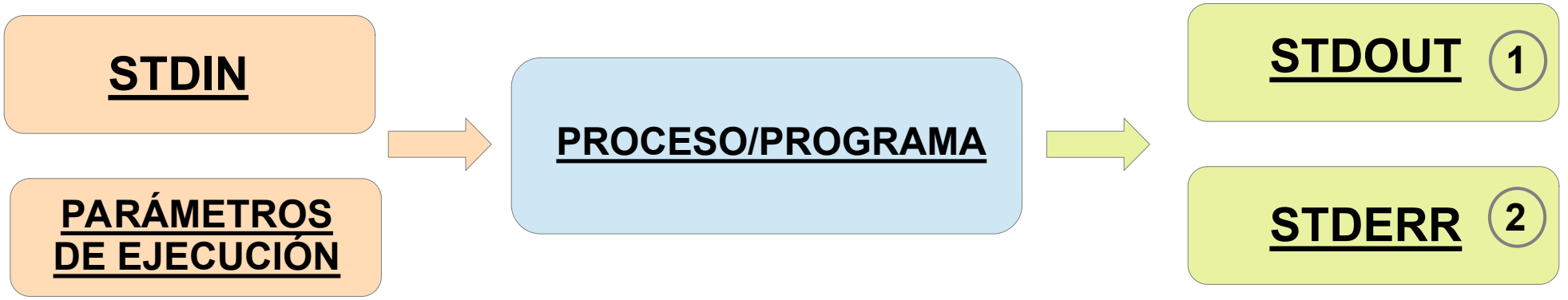
User: ~$ bwa -h

User: ~$ man bwa

User: ~$ bwa

```
Usage: bwa mem [options] <idxbase> <in1.fq> [in2.fq]
               Parámetros      Argumentos
Algorithm options:

       -t INT        number of threads [1]
       -k INT        minimum seed length [19]
       -w INT        band width for banded alignment [100]
       -d INT        off-diagonal X-dropoff [100]
       -r FLOAT      look for internal seeds inside a seed longer than {-k} * FLOAT [1.5]
       -y INT        seed occurrence for the 3rd round seeding [20]
       -c INT        skip seeds with more than INT occurrences [500]
       -D FLOAT      drop chains shorter than FLOAT fraction of the longest overlapping chain [0.50]
       -W INT        discard a chain if seeded bases shorter than INT [0]
       -m INT        perform at most INT rounds of mate rescues for each read [50]
       -S            skip mate rescue
       -P            skip pairing; mate rescue performed unless -S also in use

Scoring options:

       -A INT        score for a sequence match, which scales options -TdBOELU unless overridden [1]
       -B INT        penalty for a mismatch [4]
       -O INT[,INT]  gap open penalties for deletions and insertions [6,6]
       -E INT[,INT]  gap extension penalty; a gap of size k cost '{-O} + {-E}*k' [1,1]
       -L INT[,INT]  penalty for 5'- and 3'-end clipping [5,5]
       -U INT        penalty for an unpaired read pair [17]

       -x STR        read type. Setting -x changes multiple parameters unless overridden [null]
                     pacbio: -k17 -W40 -r10 -A1 -B1 -O1 -E1 -L0  (PacBio reads to ref)
                     ont2d: -k14 -W20 -r10 -A1 -B1 -O1 -E1 -L0  (Oxford Nanopore 2D-reads to ref)
                     intractg: -B9 -O16 -L5  (intra-species contigs to ref)
```

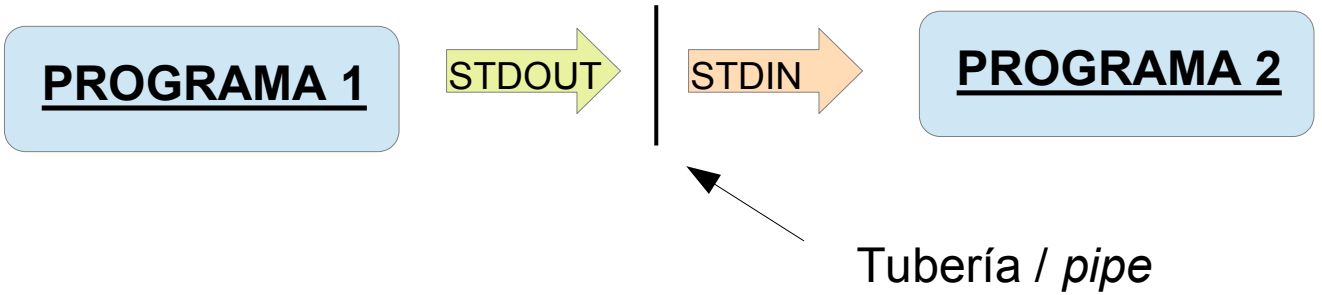
Flujos de información en los programas



$ → Enviar al proceso al fondo/*background*

> → Redirección de salidas a archivo

| → Encadenar STDOUT → STDIN



Elementos de un proceso

Identifica los elementos de este proceso:

```
User: ~$ bwa mem -t 4 Reference.fa Sample.fastq 1> Process.out 2> Process.err $
```