Tema 1

Introducción

Fundamentos de Informática

Departamento de Informática
Universidad de Oviedo

Índice

1.1 Visión general de la informática

1.2 Estructura y funcionamiento de un ordenador

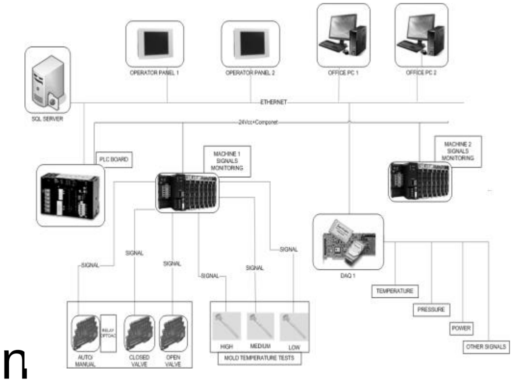
1.3 Representación de la Información en un ordenador

- 1.3.1 Sistemas de representación: decimal, binario, hexadecimal
  - Representación de números enteros
  - Representación de números reales
  - Representación de caracteres
  - Representación de booleanos

1.4 La Lógica de Proposiciones

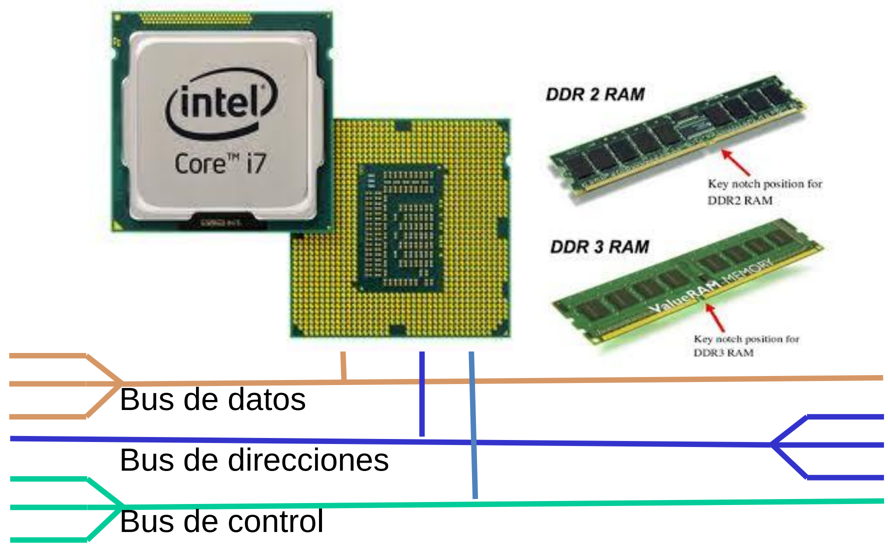
1.1 Visión general de la informática

- **Informática**: **Infor~~mación~~** + ~~auto~~**mática**
  - Según la RAE: Conjunto de conocimientos científicos y técnicas que hacen posible el tratamiento automático de la información por medio de ordenadores.
    - Ciencia que trata de la adquisición, almacenamiento, representación, tratamiento y transmisión de la información, utilizando máquinas denominadas ordenadores o computadoras.
  - La información en las empresas se gestiona con diferentes herramientas informáticas integradas, incluyendo
    - Diferentes sistemas operativos
    - Comunicaciones entre máquinas
    - Bases de datos
    - Aplicaciones de usuario
    - Aplicaciones de control y gestión de producción



1.2 Estructura y funcionamiento de un ordenador

- Pero, ¿qué es un ordenador?
  - Es una máquina que recoge unos datos, opera con ellos y retorna una salida.
  - Procesa a una velocidad infinitamente superior a la humana.
  - Es programable.
- ¿Cómo realiza estas funciones un ordenador?
  - Arquitectura y Diagrama de bloques
    - Espacio de direcciones único
  - Programa
    - Algoritmo



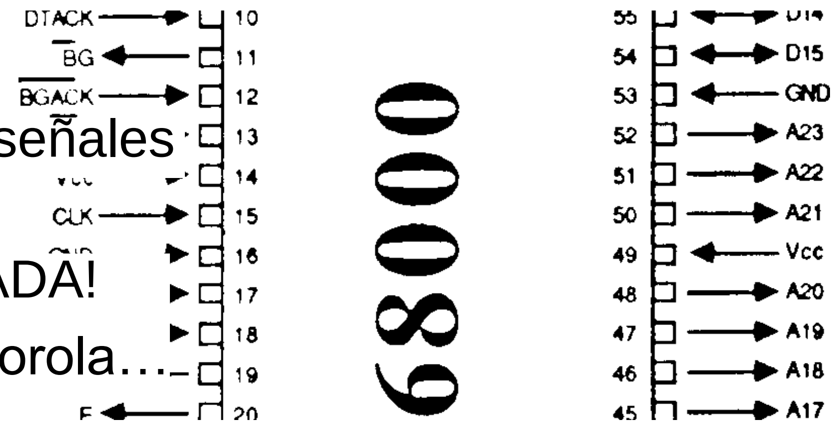
Dado un polinomio de segundo orden $p(x)=a \cdot x^2+b \cdot x+c$, evaluar p(x) para todo x distinto de cero introducido por el usuario.

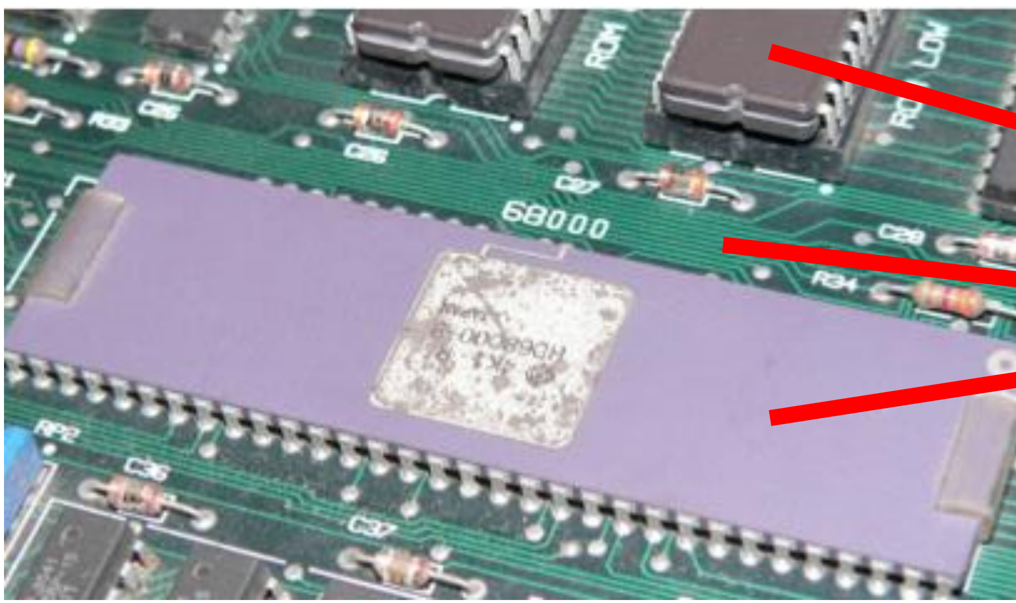
Algoritmo:
- Pedir coeficientes
- Pedir x
- Mientras x!=0
  - Calcular p(x)
  - Mostrar p(x)
  - Pedir x

```
print "Evaluacion de polinomios de 2do orden:"
print "p(x)=a*x*x+b*x+c"
a = float(raw_input("Dame a: "))
b = float(raw_input("Dame b: "))
c = float(raw_input("Dame c: "))
x = float(raw_input("Dame x: "))
while not x==0:
    s = a * x * x + b * x + c
    print "p(", x, ")=",s
    x = float(raw_input("Dame x: "))
print "Fin del programa"
```

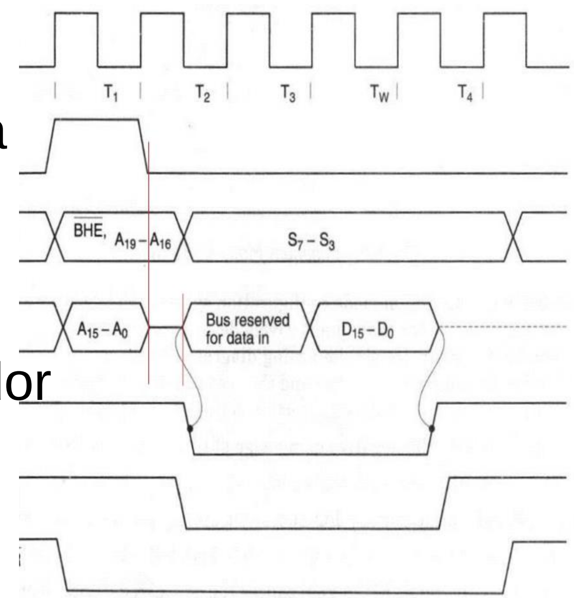
1.2 Estructura y funcionamiento de un ordenador

- La información en un ordenador
  - Mediante los buses se intercambian señales eléctricas.
  - Las señales eléctricas son TODO/NADA!
  - Un ejemplo, el antiguo 68000 de Motorola…





Memoria
Buses
Procesador



1.3 Representación de la Información en un ordenador

- Para procesar la información en un computador hay que representarla
  - Un computador solo admite 1/0's → **¡Información binaria!**
  - Sistema binario
    - Base 2 → dígitos {0, 1}
      - BIT → bit digit
    - Agrupaciones
      - 8 bits → 1 **Byte**
      - 10^3 bytes → **kB**
      - 2^{10} bits → **kb**

- Sistema decimal
  - Base 10, dígitos {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
  - Sabemos extraer las unidades y sus pesos relativos!!!

138,32
2 1 0 -1 -2

$2x10^{-2}$
$3x10^{-1}$
$8x10^{0}$
$3x10^{1}$
$1x10^{2}$

138,32

En general, si tenemos el número **abcd.ef** representado en la **base B**, EXPRESARLO EN BASE DECIMAL se realiza mediante multiplicaciones sucesivas por potencias de la base…

abc.d → $a*B^2+b*B^1+c*B^0+d*B^{-1}$
2 1 0 -1

1.3 Representación de la Información en un ordenador

- Sistema hexadecimal
  - La base es 16
  - Valores posibles: entero en el rango [0, 15]
  - Dígitos posibles: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}
  - Correspondencia directa con los números binarios:
    - $16=2^4$ → con 4 bits puedo representar todos los dígitos binarios.
  - Mismo procedimiento para llevar de decimal a hexadecimal y viceversa
    - El visto para sistema binario

10	0x	2	10	0x	2
0	0	0000	8	8	1000
1	1	0001	9	9	1001
2	2	0010	10	A	1010
3	3	0011	11	B	1011
4	4	0100	12	C	1100
5	5	0101	13	D	1101
6	6	0110	14	E	1110
7	7	0111	15	F	1111

Es fácil pasar de binario a hexadecimal. Y viceversa… ¡solo hay que usar la tabla!

Representación de los Números Enteros Positivos

DE BASE B A BASE 10	DE BASE 10 A BASE B
En general, si tenemos el número **abcd.ef** representado en la **base B**, EXPRESARLO EN BASE DECIMAL se realiza mediante MULTIPLICACIONES sucesivas por potencias de la base B…	En general, si tenemos el número **abcd.ef** representado en la **base 10**, EXPRESARLO EN BASE B se realiza mediante DIVISONES sucesivas entre la base B…

$$\underset{2\,1\,0\;\;-1}{abc.d} \rightarrow a*B^{2}+b*B^{1}+c*B^{0}+d*B^{-1}$$

25/2=12, resto 1
12/2=6, resto 0
6/2=3, resto 0
3/2=1, resto 1
½=0, resto 1 $\rightarrow 11001_2$

- Vamos a utilizar básicamente 3 tipos de representación:

	Decimal	Binario	Hexadecimal
¿Por qué?	Lo usa el humano	Lo usa el computador	Representación intermedia
Base	Base B=10	Base B=2	Base B=16
Dígitos	0 1 2 3 4 5 6 7 8 9	0 1	0 1 2 3 4 5 6 7 8 9 A B C D E F

Representación de los Números Enteros Positivos

- De decimal a binario
  - 18
  - 29
- De binario a decimal
  - 11010
  - 0110111
- De binario a hexadecimal (8 bits)
  - 11010
  - 0110111
- De hexadecimal a decimal
  - 1A2F
- De hexadecimal a binario
  - 1A2F

- 18/2=9 r=0; 9/2=4 r=1; 4/2=2 r=0; 2/2=1 r=0; 1/2=0 r=1 →10010
- 29/2=14 r=1; 14/2=7 r=0; 7/2=3 r=1; 3/2=1 r=1; 1/2=0 r=1 → 11101
- $1*2^4+1*2^3+0*2^2+1*2^1+0*2^0=26$
- $0*2^6+1*2^5+1*2^4+0*2^3+1*2^2+1*2^1+1*2^0=55$
- 0001 1010 → 0x1A
- 0011 0111 → 0x37
- $1*16^3+10*16^2+2*16^1+15*16^0=6703$
- 0001 1010 0010 1111

... Números Enteros con signo y Números Reales

- **Números Enteros con signo**
  - Con signo: ídem pero usando el bit más significativo como signo +(0) o –(1).
  - En complemento a 2

Para obtener la representación de -50:

1º- Se representa 50 en binario
    50 → 00110010
2º.- Se obtiene el c2 de 50 (-50)
    Buscar el primer 1 por la derecha
    Cambiar el resto de bits
    00110010 → **110011 10**

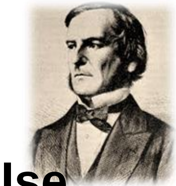
- **Números reales**
  - Estándar IEEE 754
  - Uso de notación científica más algunas reglas de simplificación
  - En 32 o en 64 bits

Signo	Exponente	Mantisa
1	8 ó 11 bits	23 ó 52 bits

Representación de los Caracteres y Datos Booleanos

- **Caracteres**
  - Alfabéticos, Dígitos, Signos Puntuación…
  - Convenio: tabla de representación
    - ASCII & ASCII extendido

Dec	Hx	Oct	Char		Dec	Hx	Oct	Html	Chr
0	0	000	NUL	(null)	32	20	040		Space
1	1	001	SOH	(start of heading)	33	21	041	!	!
2	2	002	STX	(start of text)	34	22	042	"	"
3	3	003	ETX	(end of text)	35	23	043	#	#
4	4	004	EOT	(end of transmission)	36	24	044	$	$
5	5	005	ENQ	(enquiry)	37	25	045	%	%
6	6	006	ACK	(acknowledge)	38	26	046	&	&
7	7	007	BEL	(bell)	39	27	047	'	'
8	8	010	BS	(backspace)	40	28	050	(	(
9	9	011	TAB	(horizontal tab)	41	29	051	)	)
10	A	012	LF	(NL line feed, new line)	42	2A	052	*	*
11	B	013	VT	(vertical tab)	43	2B	053	+	+
12	C	014	FF	(NP form feed, new page)	44	2C	054	,	,
13	D	015	CR	(carriage return)	45	2D	055	-	-
14	E	016	SO	(shift out)	46	2E	056	.	.



- **Booleanos:**
  - Valores posibles **True** o **False** (Verdadero o Falso; 1 ó 0)
  - Nombre en honor a George Boole (1815-1864), padre de la de Lógica Proposiciones moderna.
  - Ejemplos de situaciones que producen un dato booleano:
    - 3 > 5 False
    - 14 >= 2 True
    - 3 es entero True
  - Un dato de tipo booleano conlleva la menor cantidad de información posible: 1 bit

Un sistema de Unidades de Información

- En el **Sistema Internacional de Unidades** los múltiplos de **bit** son:
- Kilobit (kb) 10^3 bits
- Megabit (Mb), 10^6 bits= 1000 kb
- Gigabit (Gb), 10^9 bits= 1000 Mb
- Terabit (Tb), 10^{12} bits= 1000 Gb
- Petabit (Pb), 10^{15} bits= 1000 Gb

- En el **Sistema Internacional de Unidades** los múltiplos de **byte** son:
- Kilobyte (kB) 10^3 bytes
- Megabyte (MB) 10^6 bytes=1000 kB
- Gigabyte (GB) 10^9 bytes= 1000 MB
- Terabyte (TB) 10^{12} bytes= 1000 GB
- Petabyte (PB) 10^{15} bytes= 1000 TB

1.4 La Lógica de Proposiciones

- La Lógica de Proposiciones se remonta en sus inicios a la época griega clásica
- Renace su estudio con esplendor en el siglo IX con Boole, tomando el carácter de *ciencia*
- Es un modelo de cierto tipo de razonamientos humanos.
- Conlleva un lenguaje como forma de representar y manejar simbólicamente los razonamientos.
- El lenguaje de esta lógica es lo que nos interesa en esta asignatura.
- El lenguaje de la Lógica de Proposiciones es un *lenguaje formal*
- Un **lenguaje formal** es un lenguaje bien definido que consta de:
  - Un **alfabeto**: los símbolos del lenguaje
  - Una **sintaxis**: las reglas de construcción de las palabras del lenguaje
  - Una **semántica**: el significado de las palabras del lenguaje

El Alfabeto

El <u>Alfabeto</u> de la L.P. consta de los:

- **Símbolos de *Verdadero* y *Falso***

  {V, F}

- **Símbolos de proposiciones elementales**
  - Referencia:

  {A, B, C, A_1, A_2, A_3, ... ,
  B_1, B_2,B_3, ... C_1, C_2, C_3, …}
  - Alternativos:

  {llueve, hace_ frio, es_blanco,…},
  {x<=3, 8/2 = 4, x>y, x >0, …}

- **Símbolos de las conectivas u operadores lógicos**
  - De referencia:

    {$\vee$,$\wedge$,$\neg$}
  - Alternativos:
    - {or, and, not},
    - {|, &, $\neg$},
    - {+, $\cdot$, $\overline{\ }$},
    - etc.
- **Símbolos auxiliares**

  {, },(,), [,]

Cuando nos tengamos que referir a proposiciones cualesquiera utilizaremos los metasímbolos P, Q, R, P_1, P_2, …

La Sintaxis

Las **proposiciones de la L.P.** se construyen de acuerdo a las siguientes reglas sintácticas:

1. **Los símbolos V, de *Verdadero,* y F, de *Falso,* son proposiciones.**
2. **Los símbolos de proposiciones elementales son proposiciones.**
3. **Si P_1 y P_2 son proposiciones también los son:**
   - $P_1 \vee P_2$
   - $P_1 \wedge P_2$
   - $\neg P_1$

Ejemplos de proposiciones con diferentes alfabetos:

- **V** (por la 1º regla)
- A (por la 2º regla)
- **V** $\wedge$ A (por la 3º regla)
- $\neg$B (por la 3º regla)
- (A $\wedge$ $\neg$B) (por la 3º regla, 2 veces)
- (A $\wedge$ $\neg$B) $\vee$ C
- $\neg$B $\vee$ B

- llueve $\wedge$ hace_frío

- {(x>0 and x<=3) or y<6}

La Semántica (1)

- **La semántica de una proposición se reduce a** V **o** F**.**
- **Una interpretación I de una proposición consiste en una asignación de valores de verdad a cada proposición elemental de ella**.

Ejemplo.

I= {A=V, B=V}

Es una interpretación de $(A \wedge \neg B)$, pero no lo es de $(A \wedge \neg B) \vee C$, ni de (llueve $\wedge$ hace_frio).

- **Los símbolos V y F tienen valores de verdad asociados** V **y** F **respectivamente, bajo cualquier interpretación.**

Obsérvese como se distinguen en negrita los símbolos de proposición **V** y **F**, de sus valores de verdad V y F.

- **El valor de verdad de una proposición elemental viene dado siempre por la interpretación que se considere en ese momento**.

A continuación se proporcionarán las reglas que permitirán evaluar una proposición no elemental bajo cualquier interpretación

La Semántica (2)

- Reglas semánticas de las conectivas u operadores lógicos.

Sean P1 y P2 proposiciones cualesquiera. Entonces la semántica de las proposiciones construidas a partir de ellas mediante las conectivas lógicas, viene resumida en las *tablas de verdad* siguientes:

- $\wedge$

P1	P2	P1 $\wedge$ P2
V	V	V
V	F	F
F	V	F
F	F	F

- $\vee$

P1	P2	P1 $\vee$ P2
V	V	V
V	F	V
F	V	V
F	F	F

- $\neg$

P1	$\neg$ P1
V	F
F	V

Evaluación de proposiciones (1)

- Evaluación de $(A \wedge \neg B) \vee C$ bajo I= {A=V, B=F, C= F}

	(A	∧	¬	B)	∨	C
1º	V			F		F
2º	V		V			F
3º		V				F
4º					V	
5º						

- Evaluación de $(A \wedge \neg B) \vee C$ bajo todas sus interpretaciones (una por fila) o **Tabla de Verdad**

A	B	C	$\neg B$	$A \wedge \neg B$	$(A \wedge \neg B) \vee C$
V	V	V	F	F	V
V	V	F	F	F	F
V	F	V	V	V	V
V	F	F	V	V	V
F	V	V	F	F	V
F	V	F	F	F	F
F	F	V	V	F	V
F	F	F	V	F	F

Evaluación de proposiciones (2)

- Evaluación de **(x>3) ∧ (y<=0)** bajo I= {x= 4, y= 5}

	(x>3)	∧	¬	(y<=0)
1º	(4>3)	∧	¬	(5<=0)
2º	V			F
3º	V		V	
4º		V		

- Evaluación de **((x>3) ∧ ¬(y<=0)) ∨ A** bajo I={x= 4, y= 5, A=V}

	(x>3)	∧	¬	(y<=0)	∨	A
1º	((4>3)	∧	¬	(5<=0))	∨	A
2º	V			F		V
3º	V		V			V
4º		V				V
5º					V	

Las Leyes Lógicas

Las **leyes lógicas** son equivalencias muy importantes entre proposiciones; es decir, proposiciones con idéntica tabla de verdad.

Sean P, Q y R proposiciones cualesquiera:

- **Asociativas**
  - $(P \vee Q) \vee R = P \vee (Q \vee R)$
  - $(P \wedge Q) \wedge R = P \wedge (Q \wedge R)$
- **Conmutativas**:
  - $P \vee Q = Q \vee P$
  - $P \wedge Q = Q \wedge P$
- **Idempotente**
  - $\neg \neg P = P$
- **Distributivas**
  - $(P \vee Q) \wedge R = (P \wedge R) \vee (Q \wedge R)$
  - $(P \wedge Q) \vee R = (P \vee R) \wedge (Q \vee R)$
- **De los complementarios**
  - $P \vee \neg P = \mathbf{V}$,
  - $P \wedge \neg P = \mathbf{F}$
- De la **V** y de la **F**
  - $\mathbf{V} \vee P = \mathbf{V}$, $\mathbf{V} \wedge P = P$
  - $\mathbf{F} \vee P = P$, $\mathbf{F} \wedge P = \mathbf{F}$
- **De *De Morgan***
  - $\neg(P \vee Q) = \neg P \wedge \neg Q$
  - $\neg(P \wedge Q) = \neg P \vee \neg Q$

Simplificaciones sintácticas

- Simplificar las proposiciones siguientes:
- $\neg((\neg A \wedge B) \vee \neg(B \vee \neg A)) =$

  $\neg (\neg A \wedge B) \wedge \neg \neg(B \vee \neg A) =$

  $\neg(\neg A \wedge B) \wedge (B \vee \neg A) =$

  $(\neg\neg A \vee \neg B) \wedge (B \vee \neg A) =$

  $(A \vee \neg B) \wedge (B \vee \neg A)$
- $\neg((x<=3 \vee x>=5) \vee (3>y \vee y>=9)) =$

  $\neg(x<=3 \vee x>=5) \wedge \neg(3>y \vee y>=9) =$

  $(\neg(x<=3) \wedge \neg(x>=5)) \wedge (\neg(3>y) \wedge \neg(y>=9)) =$

  (utilizando las relaciones complementarias)

  $(x>3 \wedge x<5) \wedge (y>=3 \wedge y<9) =$

  (utilizando notación de intervalos)

  $x \in (3,5)$ e $y \in [3,9)$

Traducciones

Realícense las siguientes traducciones:

- **Es blanco y no es blanco.**

Sea el alfabeto {A,…} donde identificamos A≡ *Ser blanco*, entonces la traducción será: $A \wedge \neg A$, que, por otra parte, sabemos que $A \wedge \neg A$ = **F**, por lo que ambas proposiciones se pueden tomar como traducciones correctas.

- **No es verdad que se columpie y no use el móvil, pero al menos canta o ríe**.

A ≡ *Columpiarse*, B ≡ *Usar el móvil*, C ≡ *Cantar*, D ≡ *Reir*

$$\neg(A \wedge \neg B) \wedge (C \vee D)$$

- **Sean x e y enteros: *al menos uno de los dos es mayor o igual que cero, y el primero no es mayor que el segundo*.**

•(x>=0)

- $((x>=0) \vee (y>=0)) \wedge \neg\, (x>y) \;=\; ((x>=0) \vee (y>=0)) \wedge (x<=y)$

Ejercicios (tomados de exámenes)

- Considere la expresión “El número *x* es impar, o es par y mayor o igual a 10”. Escriba una expresión en Python usando una variable *x* con el mismo significado que en la expresión.

  (Nota: la expresión *a%b* en Python devuelve el resto de una división entera)

  (x % 2 != 0) or ((x % 2 == 0) and (x >= 10))

- Niegue la expresión anterior y aplique posteriormente las leyes de De Morgan, indicando la expresión resultante sin negación alguna. ¿Cuál sería su resultado para *x=6*?

  not ((x % 2 != 0) or ((x % 2 == 0) and (x >= 10)))

  not (x % 2 != 0) and not ((x % 2 == 0) and (x >= 10))

  not (x % 2 != 0) and (not (x % 2 == 0) or not (x >= 10))

  (x % 2 == 0) and ((x % 2 != 0) or (x < 10))

  El resultado para *x=6* sería *True*

  ¿Se podría simplificar todavía más esta expresión?