

# Ingeniería de software Parcial 2

01/03/2022

← ↻

librerías C#

librerías del Sistema -

librerías colaborativas NuGet -

librería del Sistema

System.Numerics ...

System. ...

librerías colaborativas

NuGet -

Parser

"x<sup>12</sup> - sin(x)" → Parser → f(x) = x<sup>2</sup> - sin(x)

↑

2 + 3 \* 5 / 3.2 -

f(x) = valor.

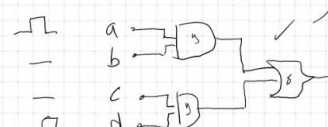
info, lundin, math, dll - → NuGet

← ↻

función lógica:

f(a,b,c,d) = a.b + c.d

f<sub>1</sub>(a,b,c,d) = (a**o**b) **y** (c**y**d)



bool a,b,c,d,r

Operadores y funciones compatibles  
+, -, \*, /, ^, %, %n

^ se eleva a (potencia) por ejemplo 3^2  
%n es el operador del módulo

sqrt, sin, cos, tan, atan, acos, asin, acotan, exp, ln, log, sinh, cosh, tanh, abs, ceil, floor, fac, sfac, round, fpart

Estas funciones se asignan principalmente a las funciones System.Math excepto fac, sfac que es las funciones factoriales y semifactoriales y fpart que devuelve la parte decimal de un valor.

!, ==, !=, ||, &&, >, <, >=, <= Operadores lógicos,  
1.0 significa verdadero, 0.0 significa falso. Si una expresión se evalúa a algo que no sea 1.0, se considera falsa

f<sub>1</sub>(a,b,c,d) = (a**y**b) **y** (c**y**d) **y** (false) ✓

float = logico

Calculadora lógica

f <sub>1</sub>	2
f	2

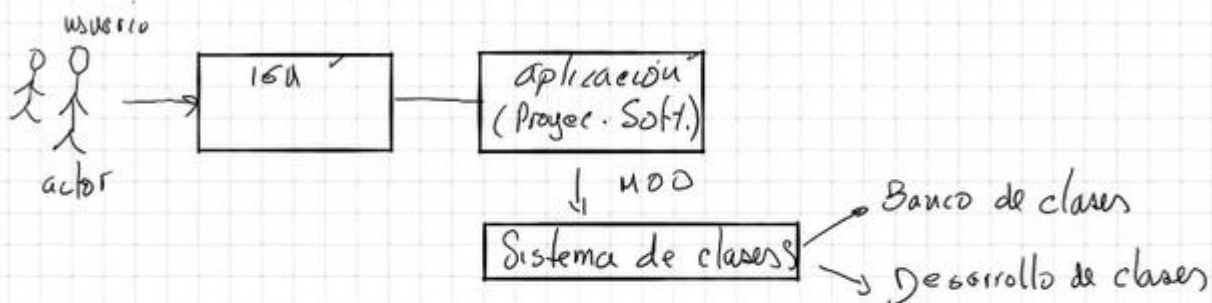
f<sub>1</sub>

f

03/03/2022 DIA DEL SISMO

FALTAN COSAS

# Metodología Orientada a Objetos (MOO)



$$\{c_i \mid c_i \in S\}$$

$$S = \{c_1, c_2, \dots, c_n\}$$

$$c_i \in S \Rightarrow$$

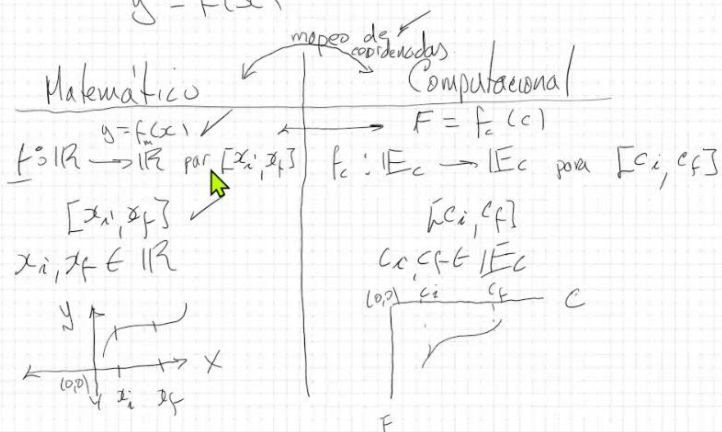
$$\{\delta_j \mid \delta_j \in c_i\}$$

$$c_i = \{\delta_1, \delta_2, \dots, \delta_n\}$$

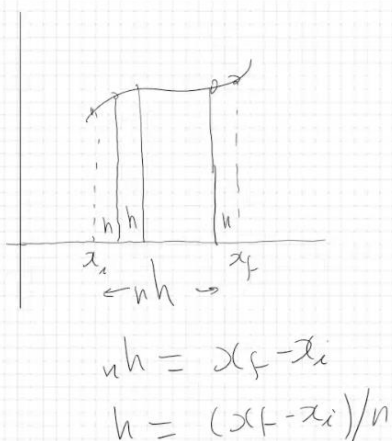
Enunciación:

Crear una aplicación para un graficador de funciones

$$y = f(x)$$



Análisis:



```

12 {
13     3 referencias
14     public partial class Form1 : Form
15     {
16         double xi, xf, x, yi;
17         int n;
18         1 referencia
19         public Form1()
20         {
21             InitializeComponent();
22         }
23         1 referencia
24         private void btgraficar_Click(object sender, EventArgs e)
25         {
26             double h;
27             xi = double.Parse(tBxi.Text);
28             xf = double.Parse(tBxf.Text);
29             n = chart1.Width;
30             h = (xf - xi / n);
31             for(int k=0; k<n; k++)
32             {
33                 x = xi + k * h;
34                 y = Math.Cos(x);
35             }
36         }
37     }
38 }

```

07/03/2022

Continúa Practica graficador:

Practica 1

Hacer una aplicación para graficar una función  $y=f(x)$  en  $[x_i, x_f]$

$y = \cos(x)$

---

Practica 2 del graficador

A la practica 1, agregar el parser info. lunen. math. dll para editar funciones

---

Practica 3

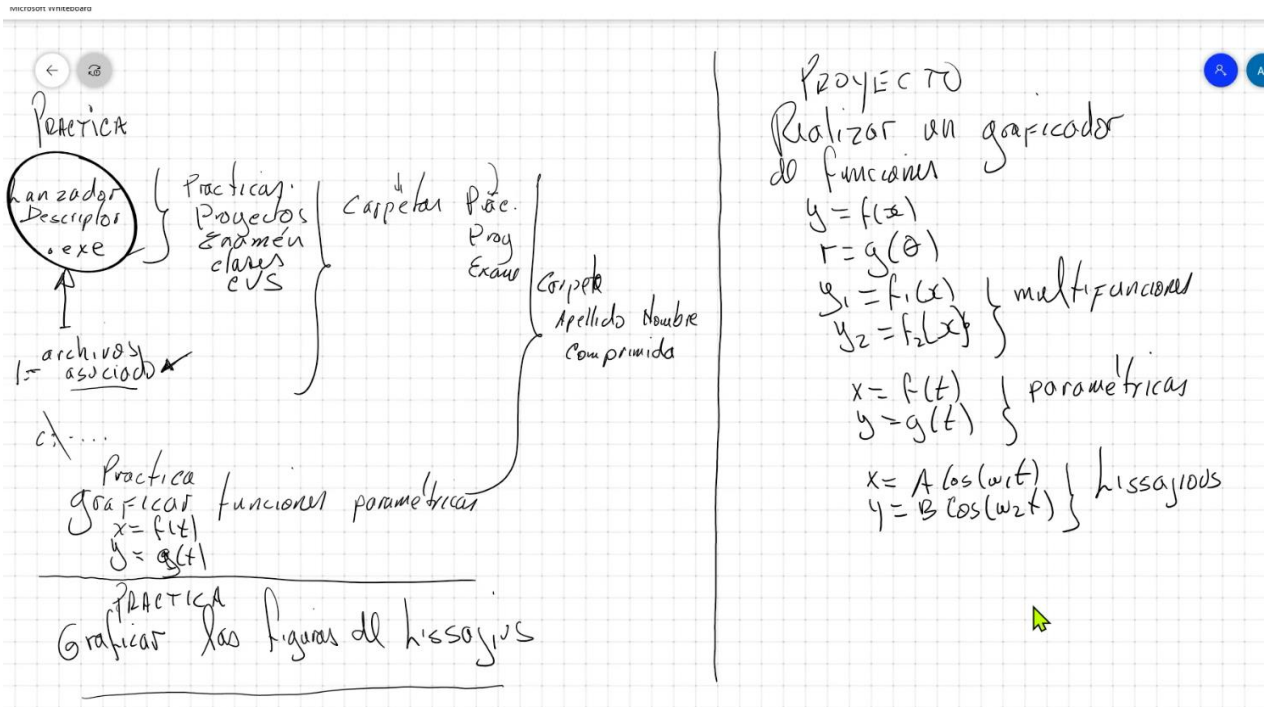
Tabular la función a la practica 2

Practica 1

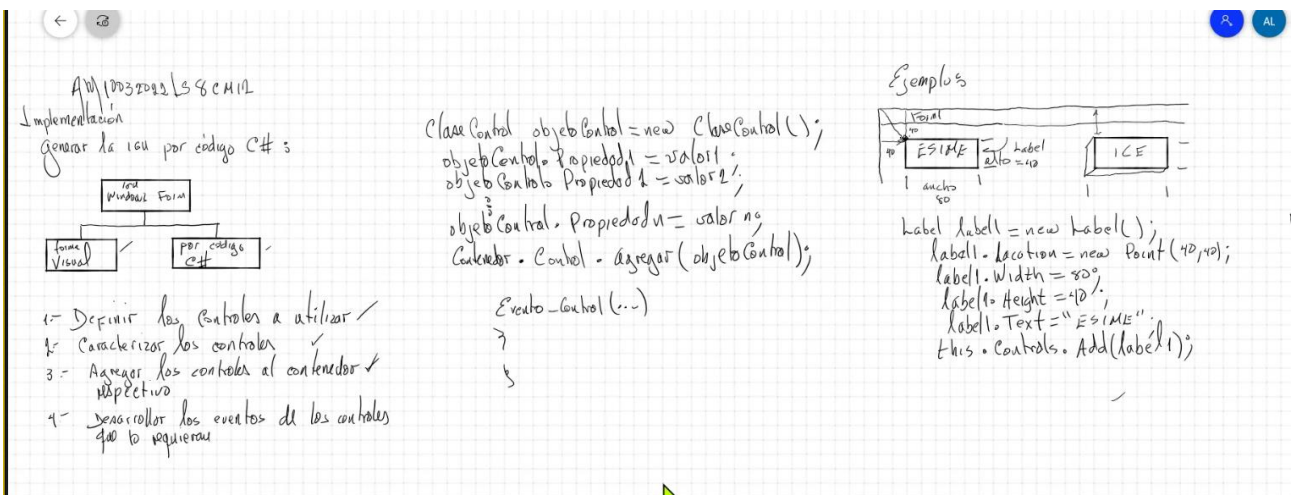
Hacer una aplicación para graficar funciones polares  $r=g(\theta)$

dados  $r, \theta \rightarrow x, y \rightarrow$

08/03/2022



10/03/2022





## PRÁCTICA 1

```

Button boton1 = new Button();
boton1.Location = new Point(200, 40);
boton1.Size = new Size(100, 40);
boton1.Text = "Click";
boton1.Click += boton1_Click;
this.Controls.Add(boton1);

private void boton1_Click(object sender, EventArgs arg)
{
    MessageBox.Show("IPN-ESIME");
}

```

15/03/2022

28

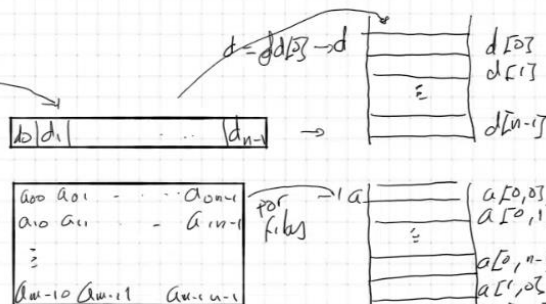
Amf 15/03/2022 JS ESIME

Implementación  
Elementos de C#  
Arreglos

Sean los conjuntos de datos:

$d = \{d_1, d_2, \dots, d_n\}$

$a = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$



- 1.- En un arreglo los elementos son adyacentes
- 2.- los elementos son de un solo tipo
- 3.- El nombre del arreglo representa al arreglo y es una referencia del inicio del arreglo
- 4.- En C# los arreglos son dinámicos
- 5.- El nombre del arreglo es una referencia

Definición de un arreglo:

tipo [ ] d ;  
 tipo [ , ] a ;  
 tipo [ , ] b ;  
 tipo [ ] [ ] c ; (arreglo irregular)

Instanciar un arreglo

d = new tipo [ n ]  
 a = new tipo [ m, n ]  
 b = new tipo [ m, n, k ]  
 → c = new tipo [ m ] [ n ] ;

Para Manipular un arreglo, se utiliza iteraciones:

$\forall i = 0, m-1$   
 $d[i]$

$\forall i = 0, m-1$   
 $\forall j = 0, n-1$   
 $a[i, j]$

Ejemplo:

```

    static void Main(...)
    {
        int [] d = new [] { 1, 2, 3, 4 };
        double [] e = new [] { 1.1, 2.1, 3.1 };
        float [] r = new [] { 1.1F, 2.1F };
        double [,] a = new [,] {
            { 1.1, 2.1 },
            { 3.1, -2.5 },
            { 1.1, 4.3 }
        };
        int [][ ] x = new [ ] {
            new [ ] { 1, 2 },
            new [ ] { 1 },
            new [ ] { 1, 2, 3 },
        };
    }
  
```

PRACTICA  
 Del ejemplo anterior  
 poner a la salida  
 los datos de los conjuntos  
 con Console.WriteLine()

$\begin{bmatrix} 1 & 2 \\ 1 & 2 & 3 \end{bmatrix}$

## Asignación de Multitipos en C#

se pueden definir variable con una asignación de tipo dado por la constante asignada.

var Variable = cte ;

- El tipo de la cte es el que define el tipo de la Variable

- Ej

```
var a = 3.5;
a = 1000;
a = 1.5F;
```

```
var b = 21;
b = 10000;
b = 3.5 X
b = 2.5FX
```

a (double)

b (int)

- Si el tipo definido inicialmente es de tamaño menor que tipos definidos mayores no se acepta.

## tipo object

este multitipo asigna diferentes tipos en compilación

object Variable ;

Ejemplo

```
object a;
a = 30;
a = 2.5;
a = 3.1F;
a = 'A';
a = "ESIME";
```

## PRÁCTICA

Para los ejemplos, agregar la salida writeLine()

17/03/2022

17/03/2022 8:01:12  
... Añados

tipos multitipo

var  
object  
dynamic

El multitipo dynamic en tiempo de ejecución puede crear los multitipos (c# no administra el código de dynamic)

dynamic variable

```
dynamic a;
a = 320;
a = 3.5;
a = 'A';
a = "ESIME"
```

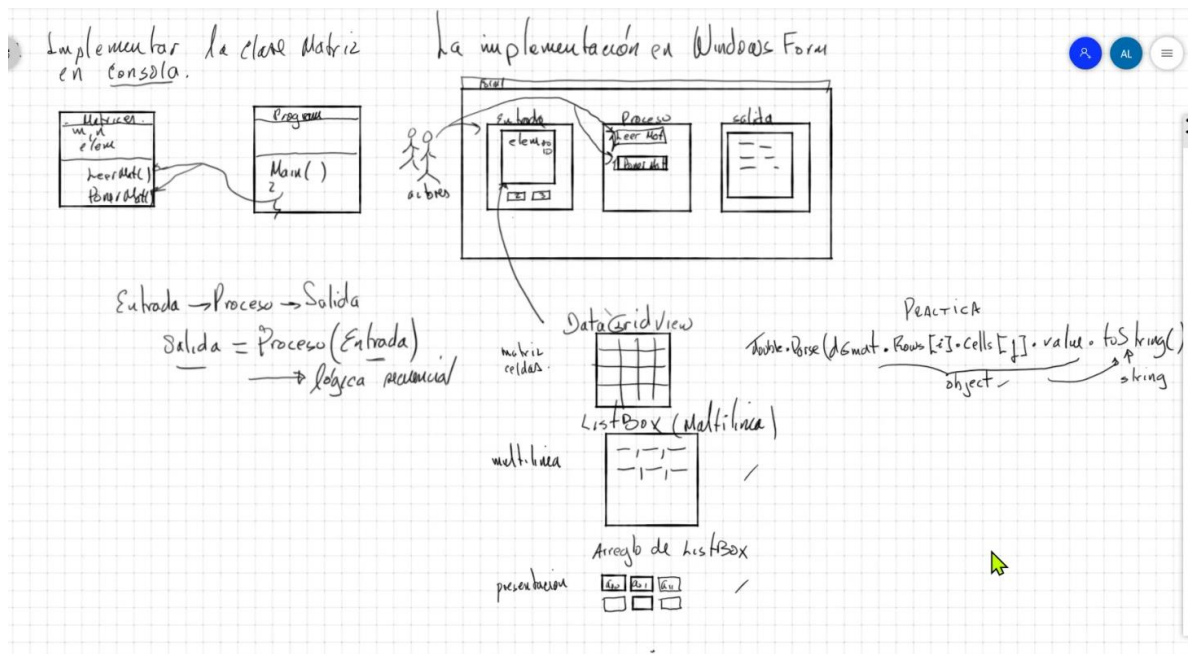
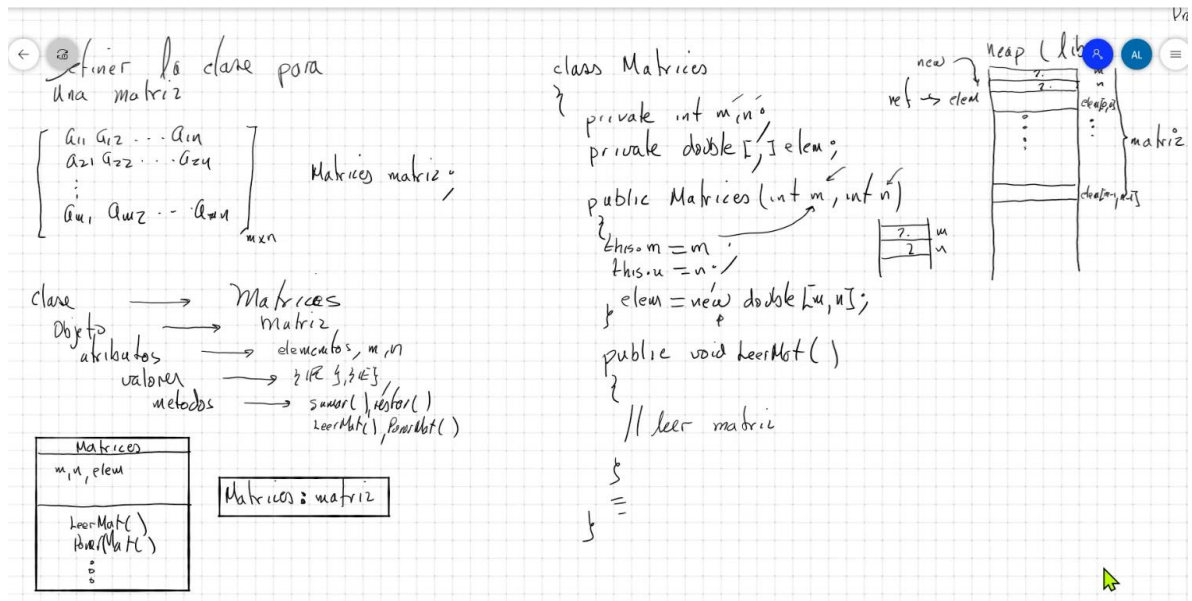
PRÁCTICA a consola.  
Ejemplos: con multitipo

```
var I[] x = new I[] { 1.2, 3.1, 1.1 };
x = new I[] { 3, 4, 8 };
x = new I[] { 'A', 'B' };
```

```
object I, J y = new object I, J = { 3.1, 2.1;
                                     2.1, 3.5; }
```

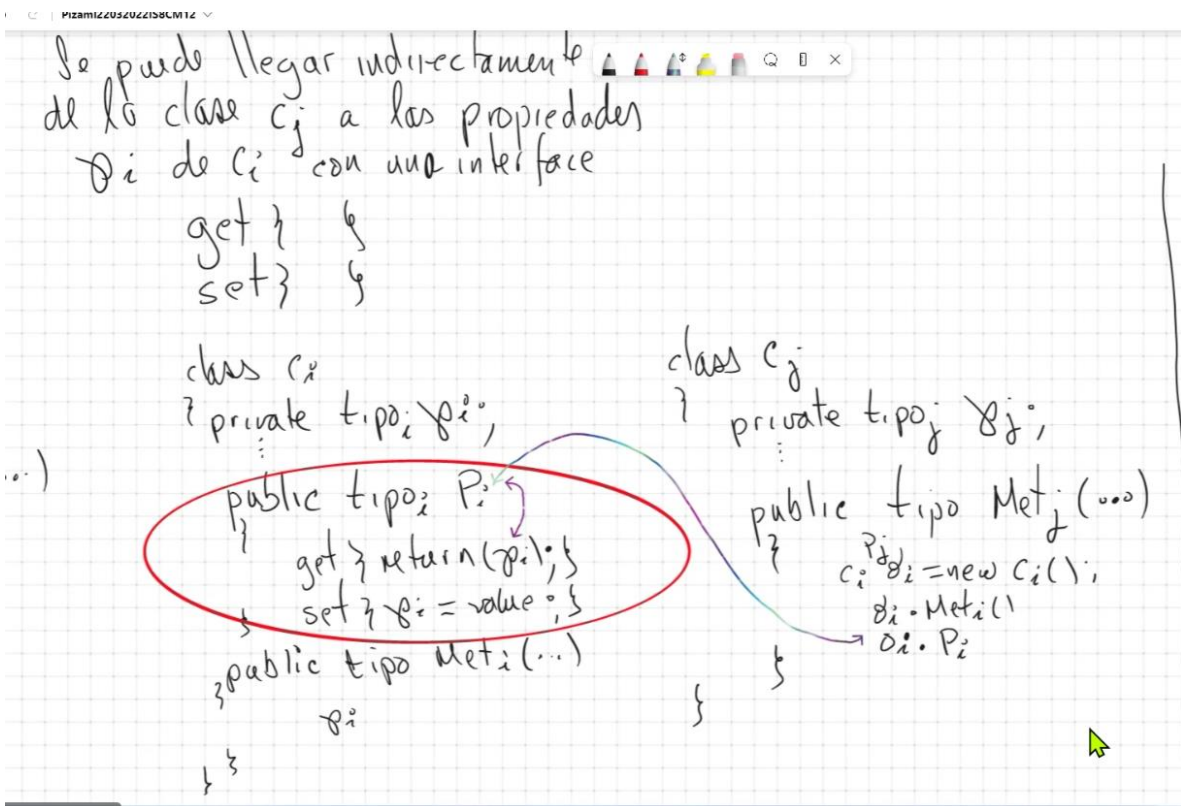
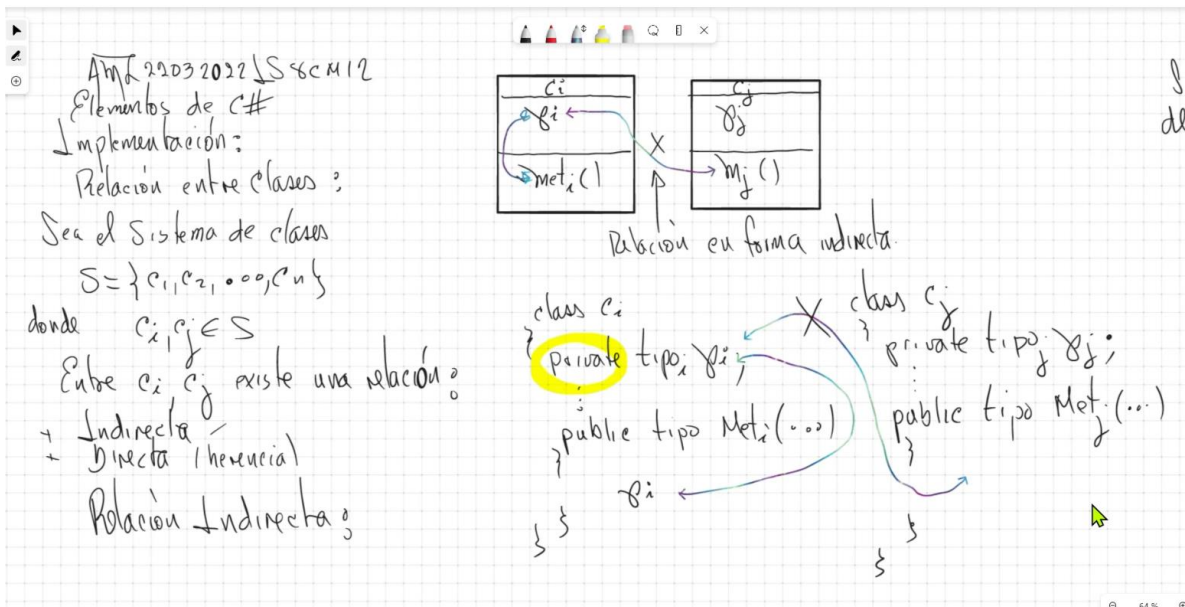
```
y = new object I, J = { "ESIME", "ICE";
                        "IPN", "COMPU"; }
```

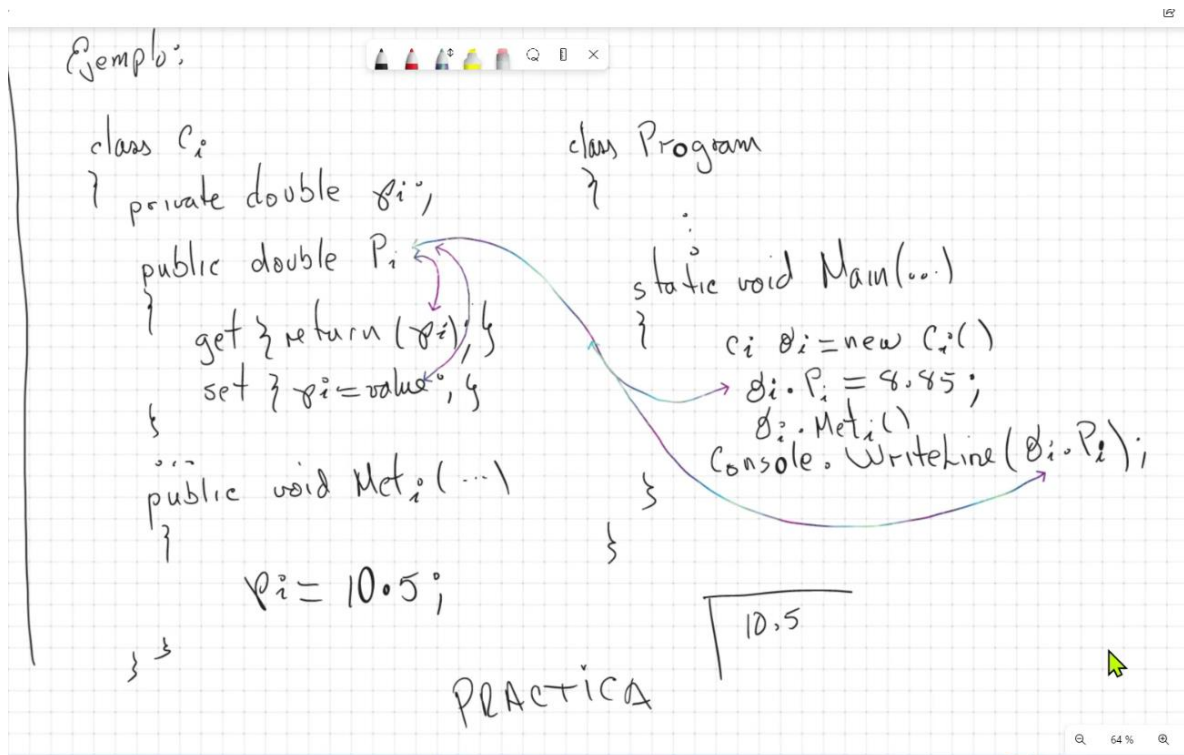
```
dynamic I[] w = new dynamic I[] { "ESIME", "ICE";
w = new dynamic I[] { 3.1, 1.6E-19, 9.1E-31; }
```



22/03/2022







```
5 using System.Threading.Tasks;
6
7 namespace WPFmatrices01
8 {
9     internal class Matrices
10    {
11        private int m,n;
12        private double[,] elem;
13
14        public Matrices(int m,int n)
15        {
16            this.m = m;
17            this.n = n;
18            elem = new double[m,n];
19        }
20        public int M
21        {
22            get { return m; }
23            set { m = value; }
24        }
25        public int N
26        {
27            get { return n; }
28            set { n = value; }
29        }
30        public double[,] Elem
31        {
32            get { return elem; }
33            set { elem = value; }
34        }
35    }
36 }
```

No se encontraron problemas.

de errores Salida

```

2 using System.Windows.Media.Imaging;
3 using System.Windows.Navigation;
4 using System.Windows.Shapes;
5
6 namespace WPFmatrices01
7 {
8     /// <summary>
9     /// Lógica de interacción para MainWindow.xaml
10    /// </summary>
11    public partial class MainWindow : Window
12    {
13        int m, n;
14        Matrices matA;
15
16        public MainWindow()
17        {
18            InitializeComponent();
19        }
20
21        private void btleermat_Click(object sender, RoutedEventArgs e)
22        {
23            string[] split;
24            for(int i=0; i<m;i++)
25            {
26                split=tBmatA.GetLineText(i).Split(',');
27                for(int j=0;j<n;j++)
28                {
29                    matA.Elem[i,j]=double.Parse(split[j]);
30                }
31            }
32        }
33        private void btponmat_Click(object sender, RoutedEventArgs e)
34        {
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64

```

```

42 }
43 private void btponmat_Click(object sender, RoutedEventArgs e)
44 {
45     string aux;
46     for (int i = 0; i < m; i++)
47     {
48         aux = " ";
49         for (int j = 0; j < n; j++)
50         {
51             aux+= matA.Elem[i, j]+" ";
52         }
53         lBmatA.Items.Add(aux);
54     }
55 }
56 private void btamano_Click(object sender, RoutedEventArgs e)
57 {
58     m = int.Parse(tBm.Text);
59     n = int.Parse(tBn.Text);
60     matA=new Matrices(m,n);
61 }
62 }
63 }
64

```

24/03/22022

Implementación:  
 Interface Gráfica WPF con XAML  
 Como plantilla se tiene:

```
<Window Enabezado de inicialización
  propiedades = valor >
  <Grid>
  </Grid>
</Window>
```

XAML es un lenguaje hipertexto.

Formato general XAML por bloque

```
<ClaseControl propiedad1=valor1 propiedad2=valor2 ... >
</ClaseControl>
```

por línea

```
<ClaseControl propiedad1=valor1 propiedad2=valor2 ... />
```

Contenedores secundarios

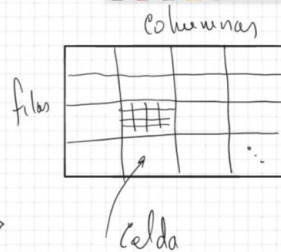
- Grid
- Stack Panel
- Canvas

Contenedor Grid

```
<Grid propiedades = valor >
```

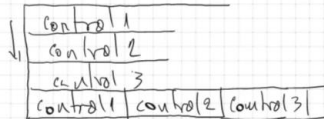
```
</Grid>
```

PRÁCTICA 1 XAML



Stack Panel

Este contenedor apila los controles agrupados en forma vertical (por default) y horizontal



```
<StackPanel propiedad=valor >
  }
</StackPanel>
```

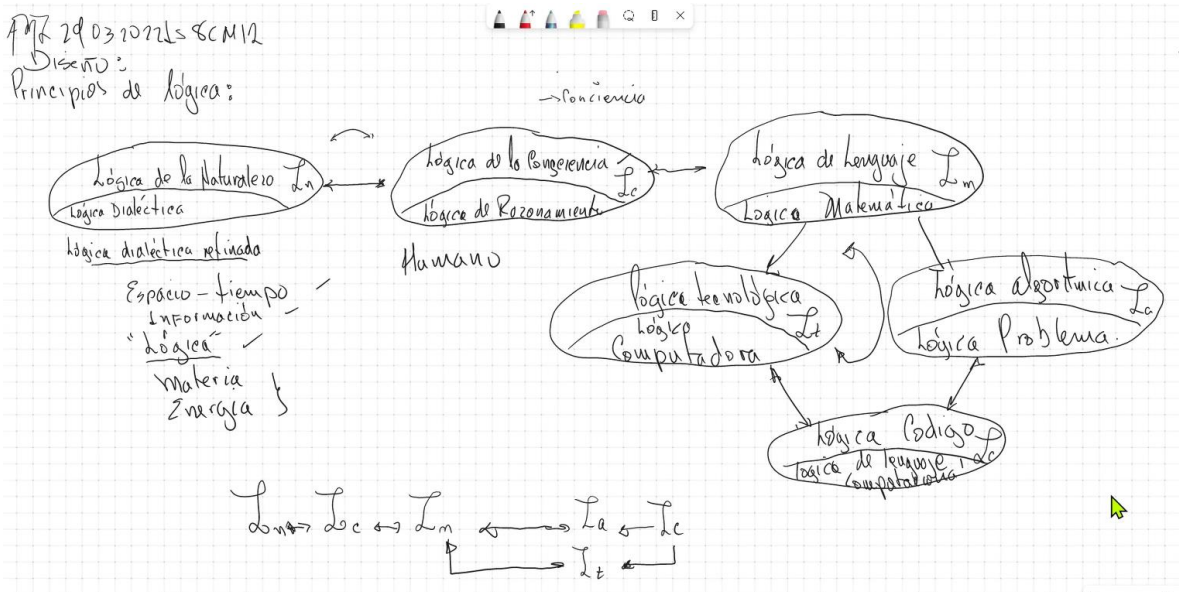
falta





28/03/2022

29/03/2022



Sea  $p, q, r$  variables lógicas

## Axiomas:

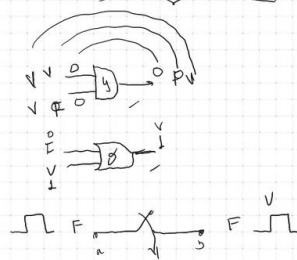
$\neg p$	$q$	$p \wedge q$	$p \vee q$	$\neg p$	$\neg q$	$p \rightarrow q$	$\neg(p \rightarrow q)$
F	F	F	F	F	F	F	F
F	F	F	F	F	F	F	F
F	V	F	V	F	V	V	F
F	V	F	V	F	V	V	F
V	F	F	V	V	F	F	V
V	V	V	V	V	V	V	F

implicação de Verdade

⇒ implicación de Verdad

↑  
fundamental  
para la algoritmia

si  $p$  es verdadero,  
implica verdadero  
de otro modo  
implicaría falso  
 $\neg p$

$$2^2 = 4$$


Tanaka's

20 ejemplos narrativos  
la implicación

$$p \rightarrow q$$

$\begin{matrix} \nearrow & \text{lover} \\ 8; & \\ \searrow & \end{matrix}$ 
 $\begin{matrix} \nearrow & \text{P} \rightarrow q \\ q; & \downarrow \\ \searrow & \end{matrix}$

parages

~~no moja~~

keze  

1.  $\neg A \rightarrow B \rightarrow F$  ✓

8	11	10	1	F
F	F		F	
V	F		F	

2.  $\varphi \wedge \psi \leftrightarrow \varphi$  ✓

$P'$	$P_1 v$
F	F
V	V

3.  $\phi V_E \rightarrow \phi$

$\varphi$	$\varphi \vee \neg \varphi$
F	F
V	V

4.  $\neg \neg \neg \neg \neg \neg \neg$

~~2/1/2~~
$$5: \varphi \wedge \psi \leftrightarrow \psi$$
$$b: \mathcal{P} \vee \mathcal{Q} \mapsto \mathcal{P}$$

7:-  $p \wedge (q \vee r) \leftrightarrow$

$$g = \varphi \vee (q \wedge r) \leftarrow$$

9.  $\neg \neg p \leftrightarrow p$

$$10 - 7(1019) \rightarrow$$

11.  $\neg(p \vee q) \leftrightarrow$

12:  $\forall x (A \rightarrow B) \leftarrow$

13  $p \vee \neg p \leftarrow$

Tarea demostrar  
las leyes

$P$	$\neg P$	$\neg \neg$
F	V	F
V	F	V

31/03/2022

### Implementación:

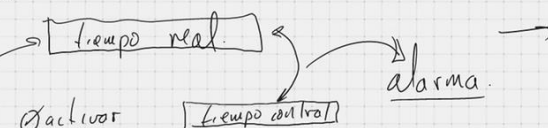
Control timer :



- Evento de finar : tiempo de Reloj del sistema

poner el tiempo de iteración

Ejemplo Alarma



↳ alarma.

### Practica

Diagram components: **tiempo real** (Label), **RadioButton** (activo), **Formo de Control** (TextBox), **Message**. Logic:  $pe\ tr == te.$

### Practica

Realizar una aplicación para lanzar en tiempo controlado aplicaciones

**tiempo real**

<input checked="" type="checkbox"/> Juego1	<input type="text" value="Fecha-hora 1"/>	}
<input checked="" type="checkbox"/> VS	<input type="text" value="h2"/>	
<input type="checkbox"/> Tarea	<input type="text" value="h3"/>	
<input checked="" type="checkbox"/> Practico	<input type="text" value="h4"/>	}
<input checked="" type="checkbox"/> ID aplicaciones		

### PROYECTO

Realizar la aplicación para una mascota virtual

Vida = 100

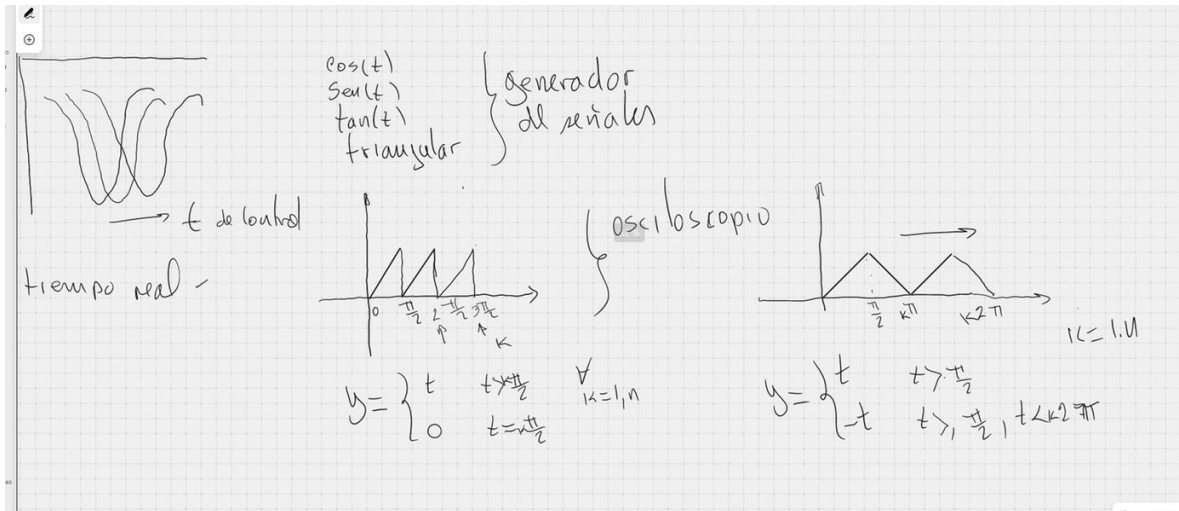
**tiempo real**

**Vida**

<input type="radio"/> desayuno	<input type="text" value="h1"/>	Vida = Vida - 10
<input checked="" type="radio"/> comida	<input type="text" value="h2"/>	
<input type="radio"/> cena	<input type="text"/>	Vida = Vida - 10
<input type="radio"/> jugar	<input type="text"/>	
⋮		

10 actividades.

Si Vida = 0 ⇒ se fue mascota.

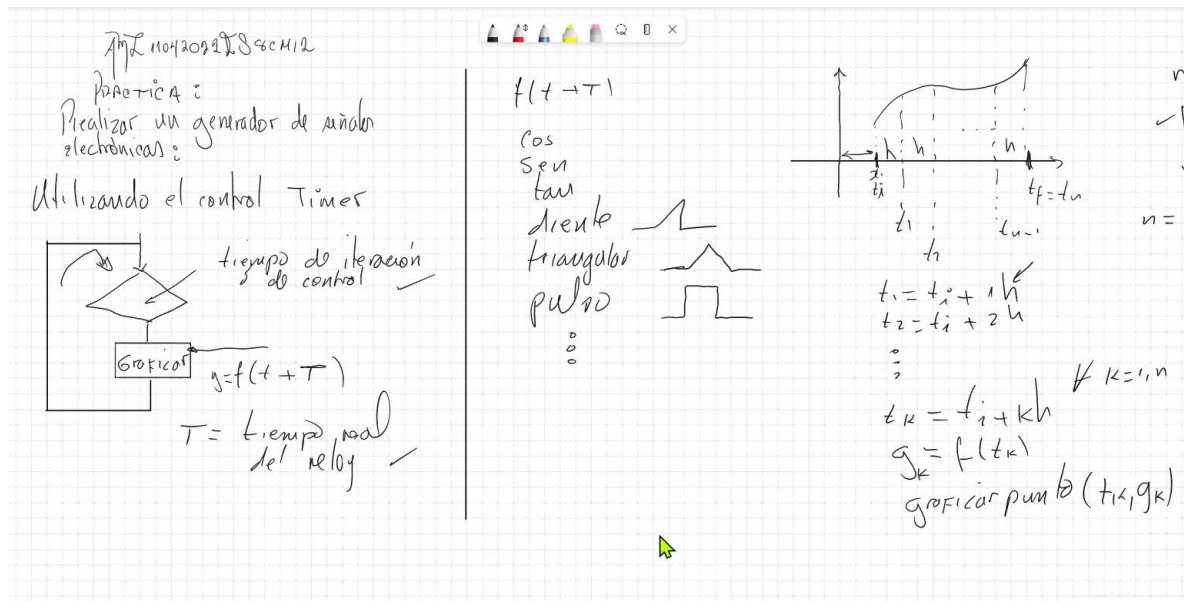


```

1  using System.Linq;
2  using System.Text;
3  using System.Threading.Tasks;
4  using System.Windows.Forms;
5
6  namespace WPFalarma1WF01
7  {
8      public partial class Form1 : Form
9      {
10         public Form1()
11         {
12             InitializeComponent();
13         }
14
15         private void timer1_Tick(object sender, EventArgs e)
16         {
17             label1.Text = DateTime.Now.ToString("hh:mm:ss:f-dd/MM/yyyy");
18             if (rBavtivar.Checked)
19             {
20                 if (label1.Text == tBalarma.Text)
21                 {
22                     System.Diagnostics.Process.Start("m2.mp3");
23                     MessageBox.Show("Alarma!!!!!!");
24                 }
25             }
26         }
27     }
28 }

```





04/04/2022

## Definición

### Axiomas

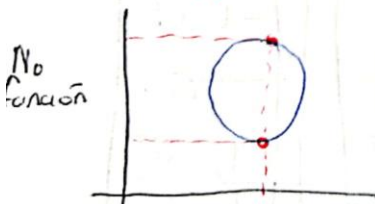
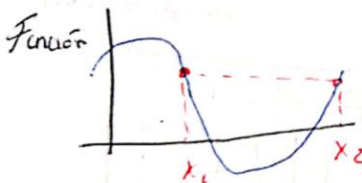
Si  $P, Q, R$  son variables lógicas de verdad Falso (F) o verdadero (V)

$P$	$Q$	$P \rightarrow Q$
F	F	V
F	V	V
V	F	F
V	V	V

En un evento, no pueden existir 2 estados diferentes al mismo tiempo.

$$y = f(x)$$

$f: \mathbb{R} \rightarrow \mathbb{R}$  en  $[x_i, x_f]$



$P_i$		$P = \text{Verdadero}$		$Q = \text{Verdadero}$	$P \rightarrow Q$ si hay pulso V
$P_i$		$P = \text{Falso}$		$Q = \text{Falso}$	$P \rightarrow Q$ F
$P_i$		$P = \text{falso}$		$Q = \text{Verdadero}$	$P \rightarrow Q$ Es verdad que no pasa
$P_i$		$P = \text{verdadero}$		$Q = \text{Falso}$	$P \rightarrow Q$ El pulso es falso que pasa el pulso

Tarea:

Realizar 10 ejemplos de implicación.

04/04/2022

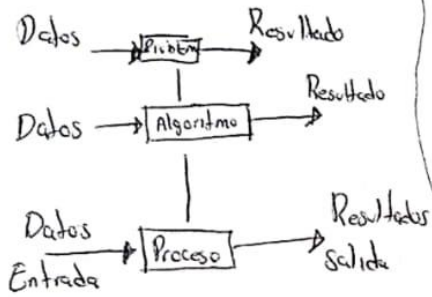
07/04/2022

07/04/2022

## DISEÑO

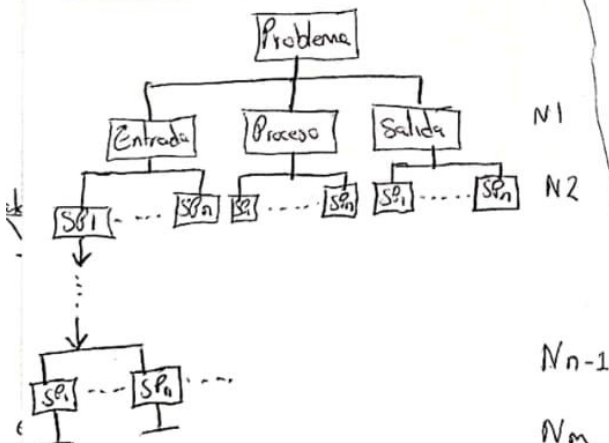
### Árbol de solución

Reducción de la complejidad de un problema



$Salida = Proceso(Entrada)$

Aplicando el axioma a el árbol de solución  
En el Primer Nivel

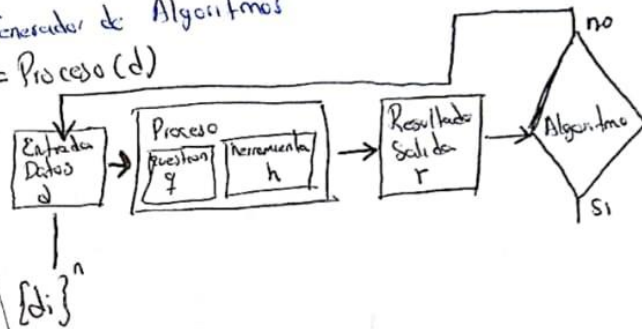


- 1.- Si un  $SP_i$  no tiene solución el Problema no tiene solución por esta estrategia
- 2.- Si en el algoritmo nivel  $N_n$  todos son hojas se obtiene solución.

- 3.- Las hojas son datos o solución intrínseca
- 4.- Si el árbol es muy frondoso el método no es adecuado.
- 5.- Para resolver el problema se empieza de Entrada  $\rightarrow$  Salida y del nivel  $N_n \rightarrow N_1$ .

Generador de Algoritmos

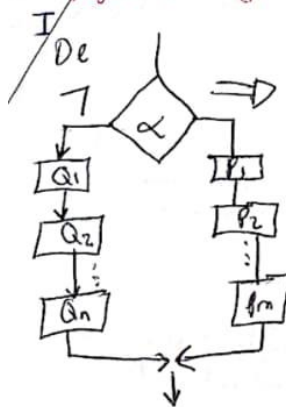
$r = Proceso(d)$



NOTAS:

Prácticas P1 en consola  
P2 en wpf

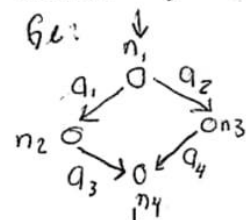
# Diagrama de Flujo de Control



## Mapa lógico de Me

	no	si
$Q_1$	$P_1$	$P_1$
$Q_2$	$P_2$	$P_2$
$\vdots$	$\vdots$	$\vdots$
$Q_n$	$P_m$	$P_m$

## Grados lógicos



$$q = \{q_1, q_2, q_3, q_4\}$$

$$n = \{n_1, n_2, n_3, n_4\}$$

$$nn = \{(n_1, n_2), (n_1, n_3), (n_2, n_4), (n_3, n_4)\}$$

## Matriz del Grafo (incidencia)

	$n_1$	$n_2$	$n_3$	$n_4$
$n_1$	0	1	1	0
$n_2$	0	0	0	1
$n_3$	0	0	0	1
$n_4$	0	0	0	0

$$T_1 q = \{q_1, q_3\}$$

$$T_1 q = \{q_2, q_4\}$$

Programa lógico  $P_L$

Trazectorias  
linealmente  
independientes

if ( $\alpha$ )

{  $P_1$ ;

$P_2$ ;

$\vdots$

$P_m$ ;

else

{  $Q_1$ ;

$Q_2$ ;

$\vdots$

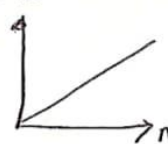
$Q_n$ ;

}

Forma compacta

if ( $\alpha$ )  $P$ ; else  $Q$ ;

Función de iteración



$$t = cf(n) = f(n)$$

$$\therefore \Theta(n)$$

Es polinomial de grado uno  
e: constante.

Tabla de implicaciones:

implicaciones	procesos
#implementación	#Estados Activos
2 = Combinaciones	

una tabla de implicaciones,  
no da la vista de que  
estados están activados  
y de que implicaciones  
depende, por lo que se  
obtiene una función  
de implicaciones.

$$Procesos = f(implicaciones)$$

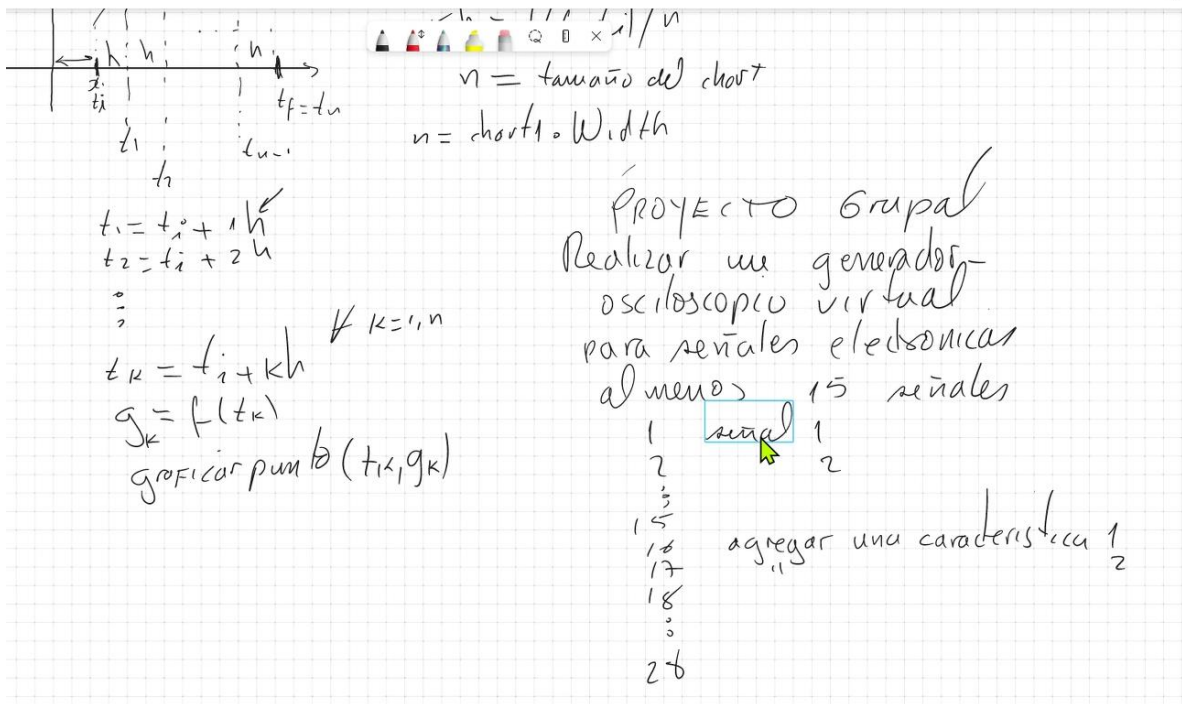
Para la estructura  
condicional

$\alpha$	$P$	$Q$
F	F	V
V	V	F

$$P = \alpha$$

$$Q = \neg \alpha$$





12/04/2022

no:

12/09/2022

de la arquitectura lógica de la cual

$A = \{e_i, e_j \mid e_i, e_j \text{ están interconectados}\}$

Tipos de estructuras lógicas

- Estructura lineal  $e_i$
- " Condicional  $e_c$
- " Iterativa Condicional  $e_{ic}$
- " " Interada  $e_{ii}$

Representación de una  $A$   
por función lógica  $F_L$

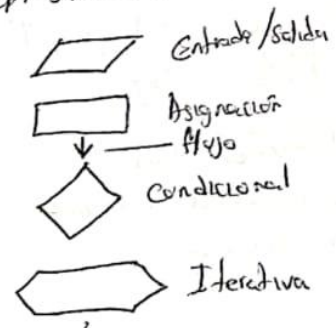
$F_L = \{e_i \mid e_i \in A\}$

representa la lógica de la  $A$

Representación gráfica

- Diagrama de flujo de control  $DC$
- mapa lógico  $Mc$
- Grafos lógicos  $GL$

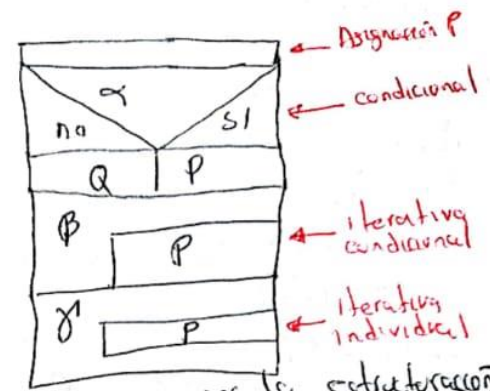
Diagrama de flujo  $DC$   
representado por los símbolos



ANSII

Representa el flujo de procesamiento.

Representación en mapa lógico.  $Mc$



Representa por regiones la estructuración

Grafos lógicos  $GL$

$q = \{q_i\}_{i=1}^n$

$n = \{n_j\}_{j=1}^m$

$n \times n = \{(n_{inicio}, n_{fin})_k\}_{k=1}^n$

Representa la topología del algoritmo  
El Grafo  $GL$  es orientado y lógico

Representación por un programa  $P_L(C\#)$

Proceso  $\lambda()$

```
{
  P1;
  P2;
  ...
  Pm;
}
```

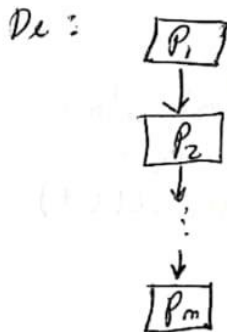
Es el programa ejecutable.

### Estructura lineal:

Se función lógica es:  
 $Fe: P_1, P_2, \dots, P_m$   
 donde  $P_i$  son procesos ejecutables  
 Sea  $P_i, P_j \in Fe$   
 Si hay una <sup>secuencia</sup> ~~frontera~~ lógica  
 $P_j = f(P_i)$   
 $\therefore P_i = P_i$   
 si  $P_j \neq f(P_i) \quad P_i \rightarrow P_j$   
 $P_j \rightarrow P_i$

- Siempre hay una estructura lógica en una estructura lineal

Se diagrama de flujo de



Mapa lógico Mx



Gratos lógico Gx

Gx: Es un grato orientado lógico

$Q = \{Q_i\}$

$n = \{n_1, n_2\}$

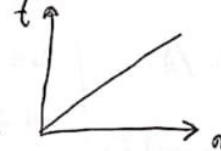
$n+n = \{n_1, n_2\}$

$Q_i, Q_j$

Programa lógico: (C#)

$\{$   
 $P_1;$   
 $P_2;$   
 $\vdots$   
 $P_n;$   
 $\}$

Función iterativa



$t = f(n)$   
 si  $f(n)$  esta acotada  
 se dice que tiene un  
 orden de iteración  
 $O(f(n)) = O(n)$

Estructura condicional  $\alpha$

Sea la implicación

$\alpha \rightarrow \beta$

Del axioma de implementación de verdad

$\alpha \Rightarrow \beta$

Se puede deducir la función lógica  $Fe$

$Fe: \{S, \alpha \Rightarrow \{P_1, P_2, \dots, P_m\}$

$\vee \{ \neg \alpha \Rightarrow \{Q_1, Q_2, \dots, Q_n\}$

Si el operador condicional  $\alpha$  la implementación una expresión lógica.

$P_i$  procesos P

$Q_j$  procesos Q

$\vee \emptyset$

$\neg$  no

$\Rightarrow$  implementación  
 Verdad