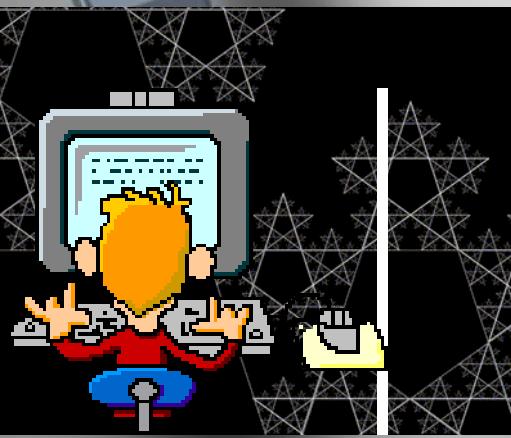




Programación I

Ing. Carlos R. Rodríguez

**Tecnicatura
Universitaria en
Programación**



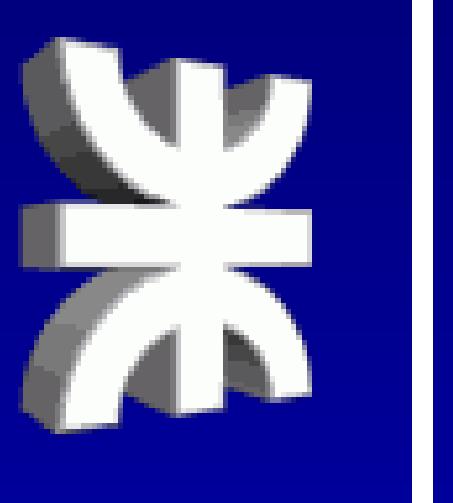
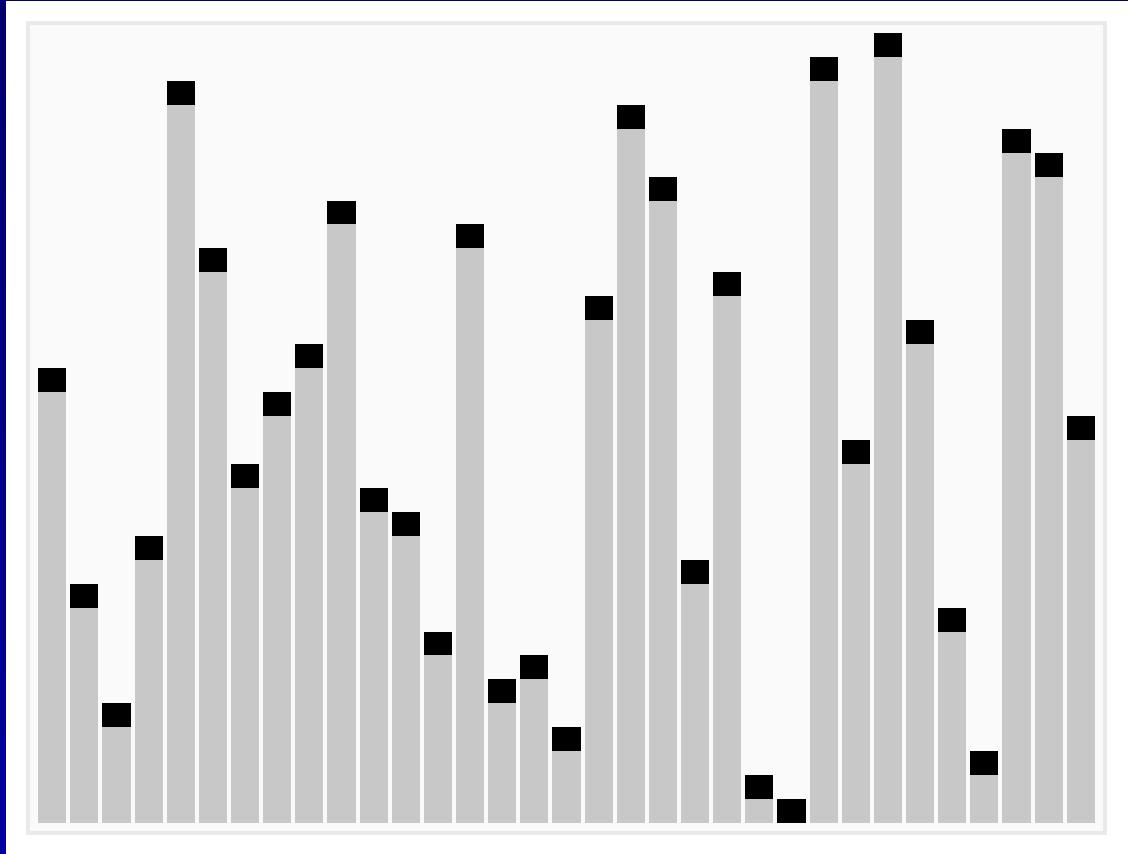
UNIVERSIDAD TECNOLÓGICA NACIONAL
Facultad Regional Mendoza

ARGENTI
el pais del





Un breve análisis sobre el por qué producimos software:





¿Qué es un Problema?



Veamos una posible definición:

“En ciencias de la computación un problema es la relación que existe entre un conjunto de instancias y un conjunto de soluciones”. Los problemas abstractos suelen definirse en dos partes:

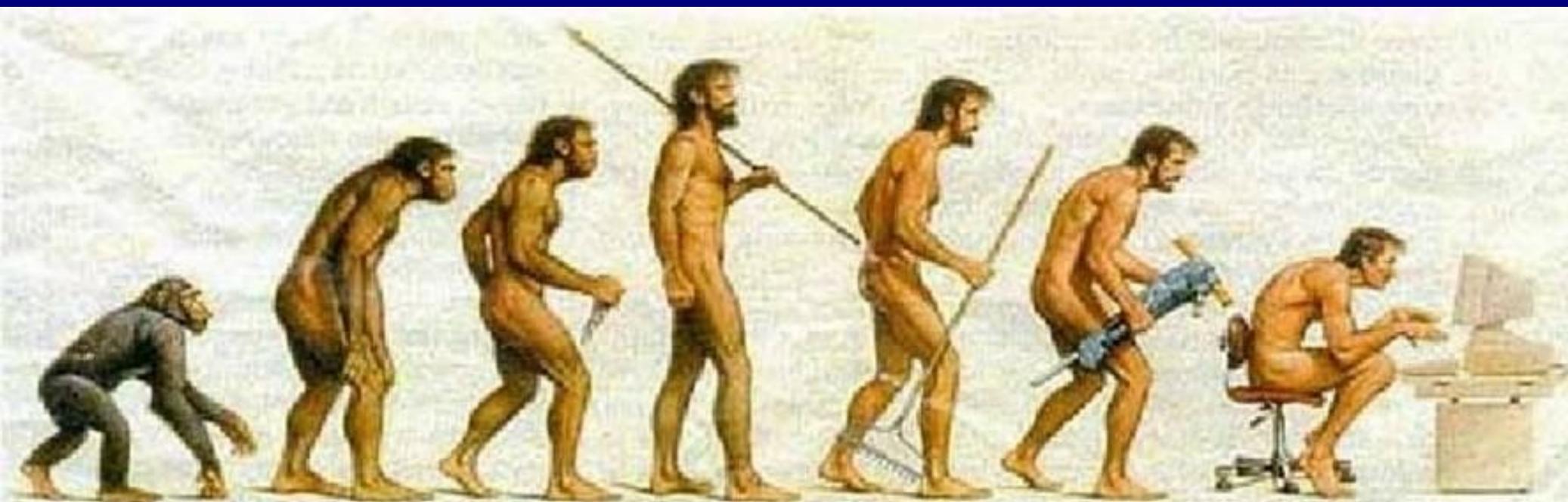
- En la primera se describe al conjunto de instancias.
 - En la segunda se describe la solución esperada para cada instancia.
- Fuente: <http://es.wikipedia.org/wiki/Problema>



¿Qué es un Problema?



Visto de otra forma:





¿Por qué programamos?



Porque alguien tiene un problema...

- Y no sabe como resolverlo, pero tiene los **recursos** como para *contratar* a una **persona o empresa** que lo resuelva.
- Por lo tanto se convierte en un **cliente**.





¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Pretende que su **solución** se materialice como un programa ejecutando en alguna computadora...
- **Supone que sabemos resolver su problema.**



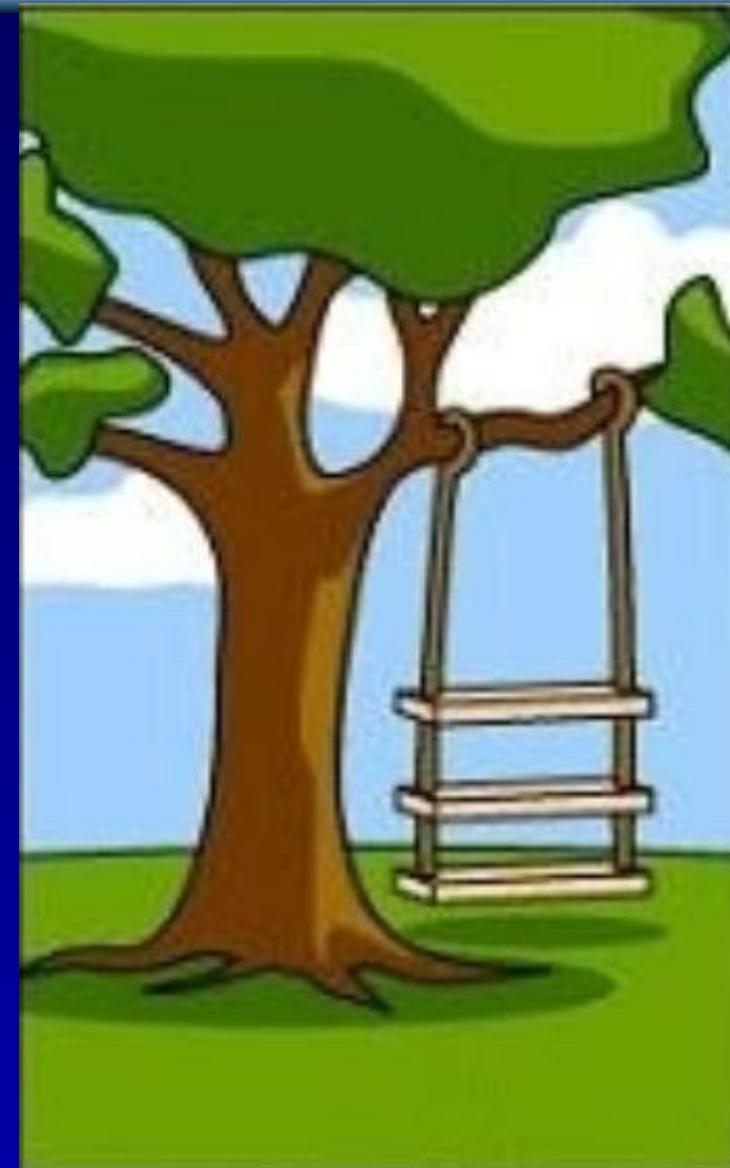


¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Pretende que su **solución** se *materialice como un programa ejecutando en alguna computadora...*
- *Supone que sabemos resolver su problema.*
 - **¿Qué explicó el cliente?**





¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Pretende que su **solución** se *materialice como un programa ejecutando en alguna computadora...*
- *Supone que sabemos resolver su problema.*
 - **¿Qué entendió el Líder del Proyecto?**



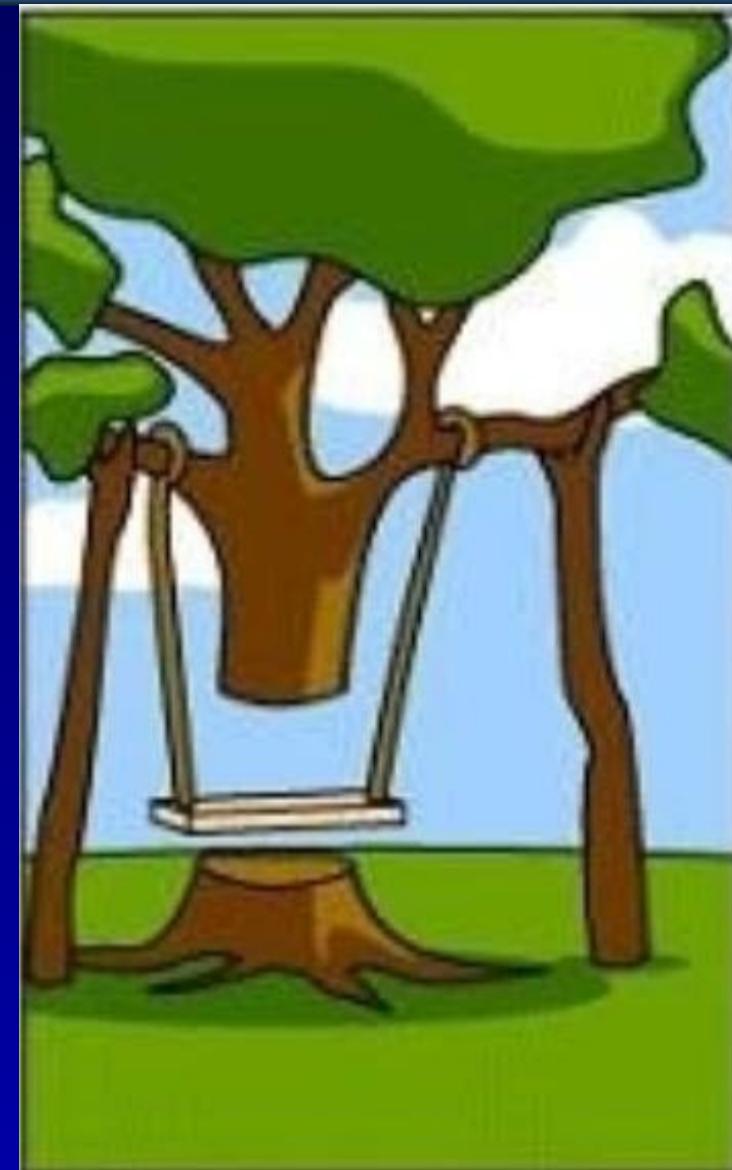


¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Pretende que su **solución se materialice como un programa ejecutando en alguna computadora...**
- **Supone que sabemos resolver su problema.**
 - **¿Qué diseñó el Analista de Sistemas?**



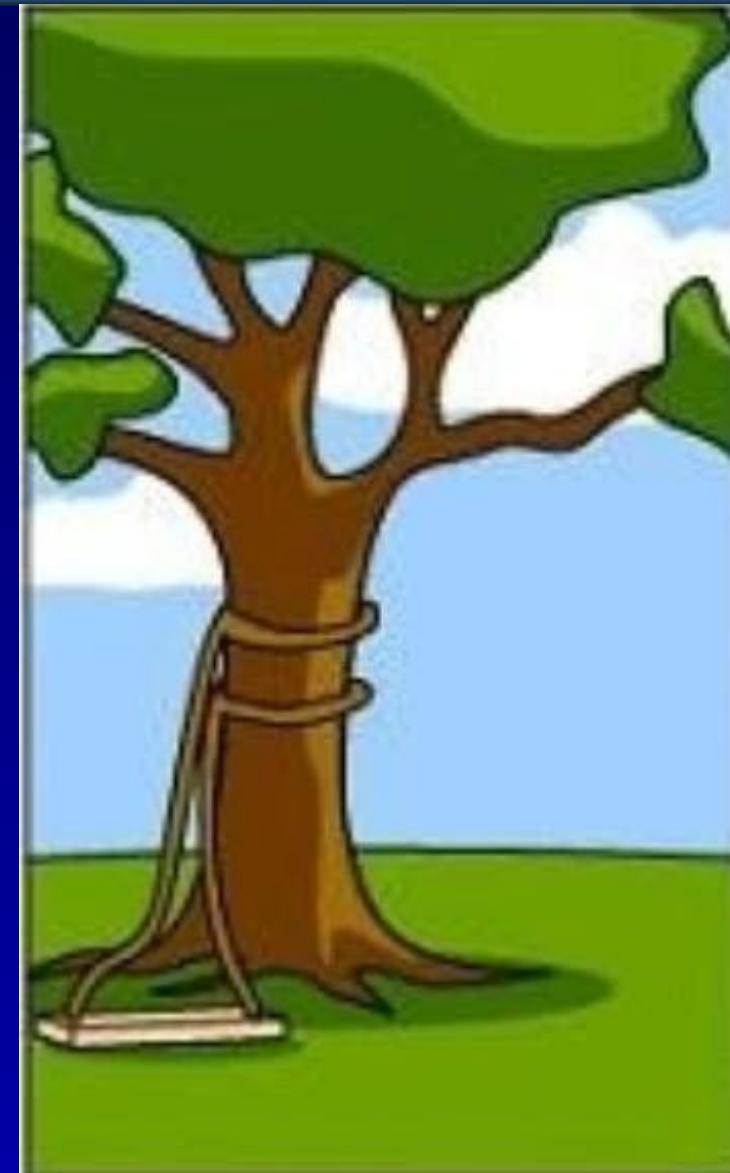


¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Pretende que su **solución** se *materialice como un programa ejecutando en alguna computadora...*
- **Supone que sabemos resolver su problema.**
 - **¿Cómo lo resolvió el programador?**





¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Pretende que su **solución** se *materialice como un programa ejecutando en alguna computadora...*
- **Supone que sabemos resolver su problema.**
 - **¿Qué recomendó el consultor?**



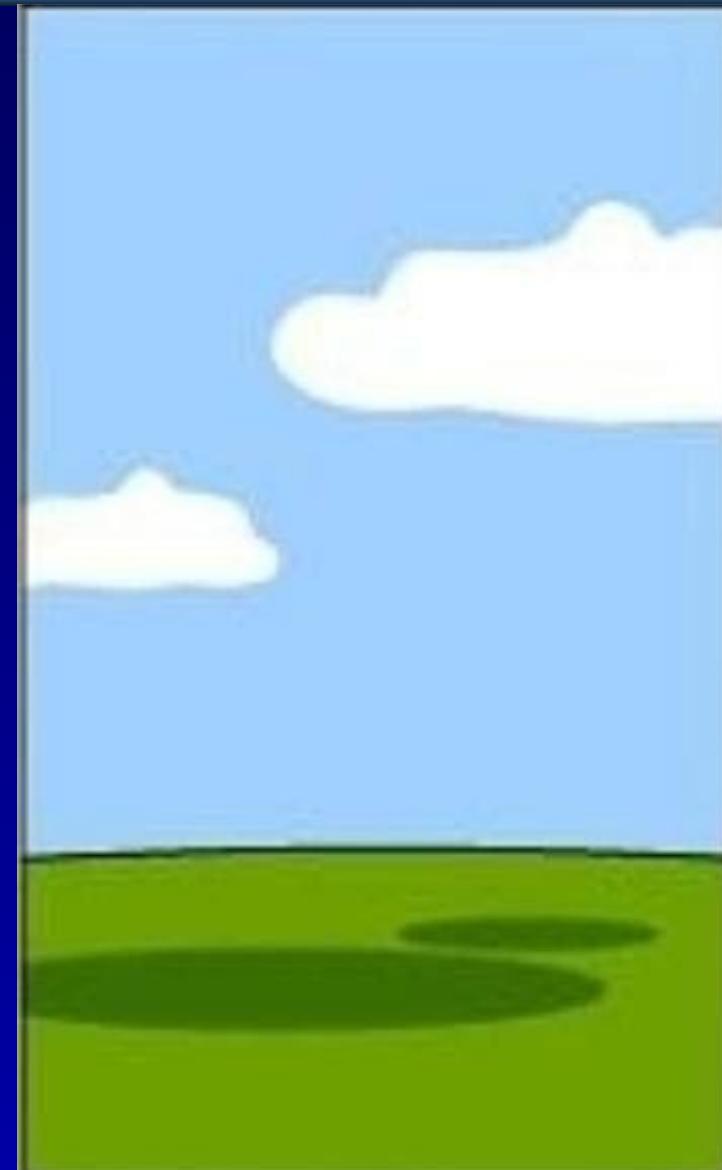


¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Pretende que su **solución se materialice como un programa ejecutando en alguna computadora...**
- **Supone que sabemos resolver su problema.**
 - **¿Cómo se documentó el proyecto?**



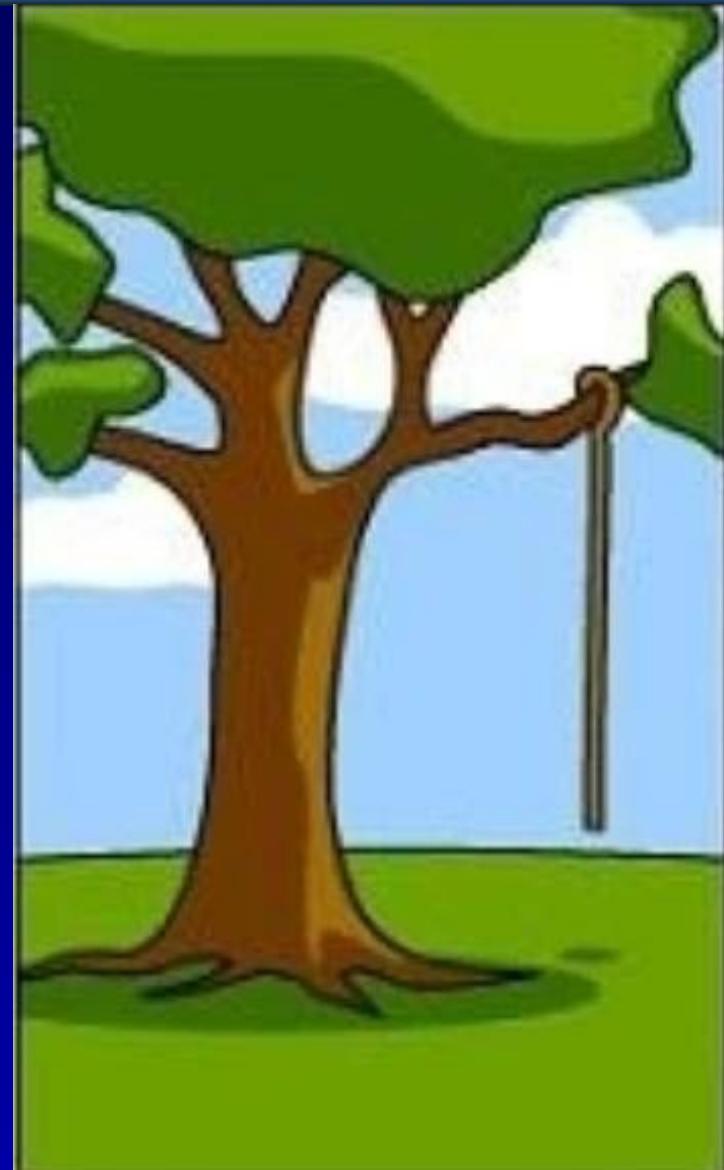


¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Pretende que su **solución** se *materialice como un programa ejecutando en alguna computadora...*
- *Supone que sabemos resolver su problema.*
 - **¿Qué se implementó hasta ahora?**





¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Pretende que su **solución** se *materialice como un programa* ejecutando en alguna computadora...
- *Supone que sabemos resolver su problema.*
 - **¿Cuánto le facturamos?**



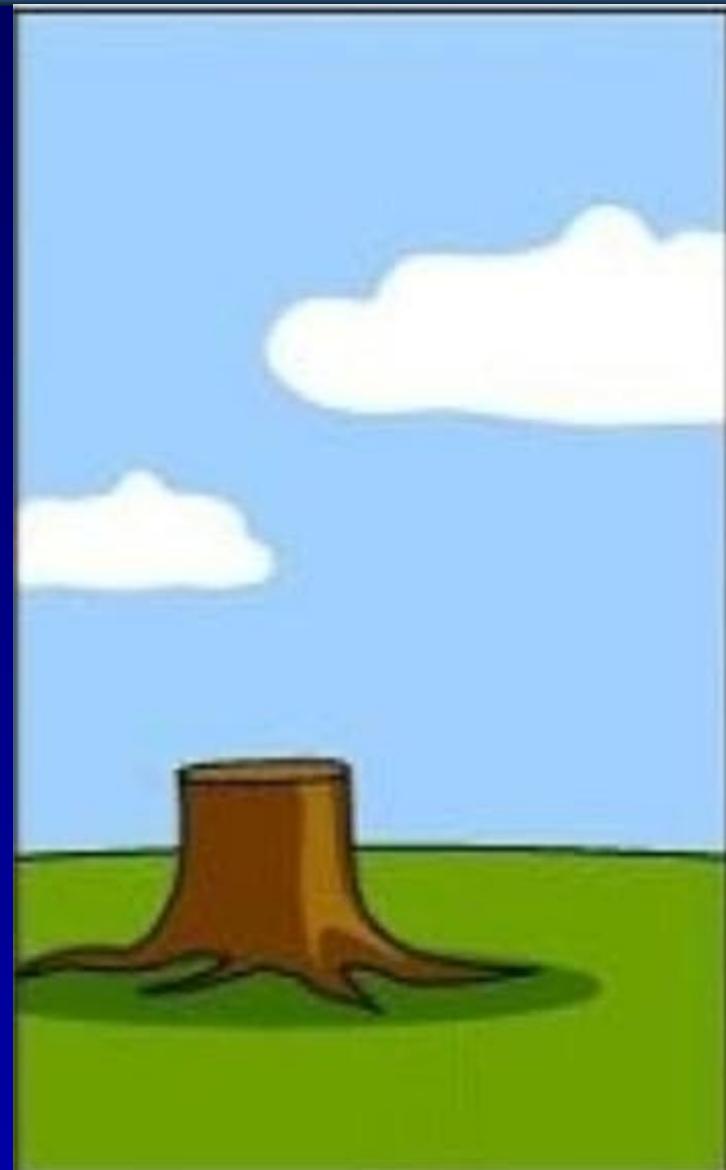


¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Pretende que su **solución** se *materialice como un programa ejecutando en alguna computadora...*
- *Supone que sabemos resolver su problema.*
 - **¿Qué nivel de soporte le brindamos?**





¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Pretende que su **solución** se *materialice como un programa ejecutando en alguna computadora...*
- **Supone que sabemos resolver su problema.**
 - **¿Qué necesitaba?**





¿Por qué programamos?

Entonces nuestro **cliente** tiene un problema...

- Para **resolver su problema**, lo que necesitamos es...
 - **¡Comprenderlo!**



Cliente
Problema
Producto
Tecnología



¿Por qué programamos?

Entonces nuestro **cliente** tiene un problema...

- Para **resolver su problema**, lo que necesitamos es **comprenderlo: dialogando con él...**





¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Para *resolver su problema*, lo que necesitamos es **comprenderlo: dialogando con él...**





¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Para *resolver su problema*, lo que necesitamos es **evitar el léxico reservado a especialistas...**





¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Para *resolver su problema*, lo que necesitamos es **dialogar con él en su mismo idioma.**



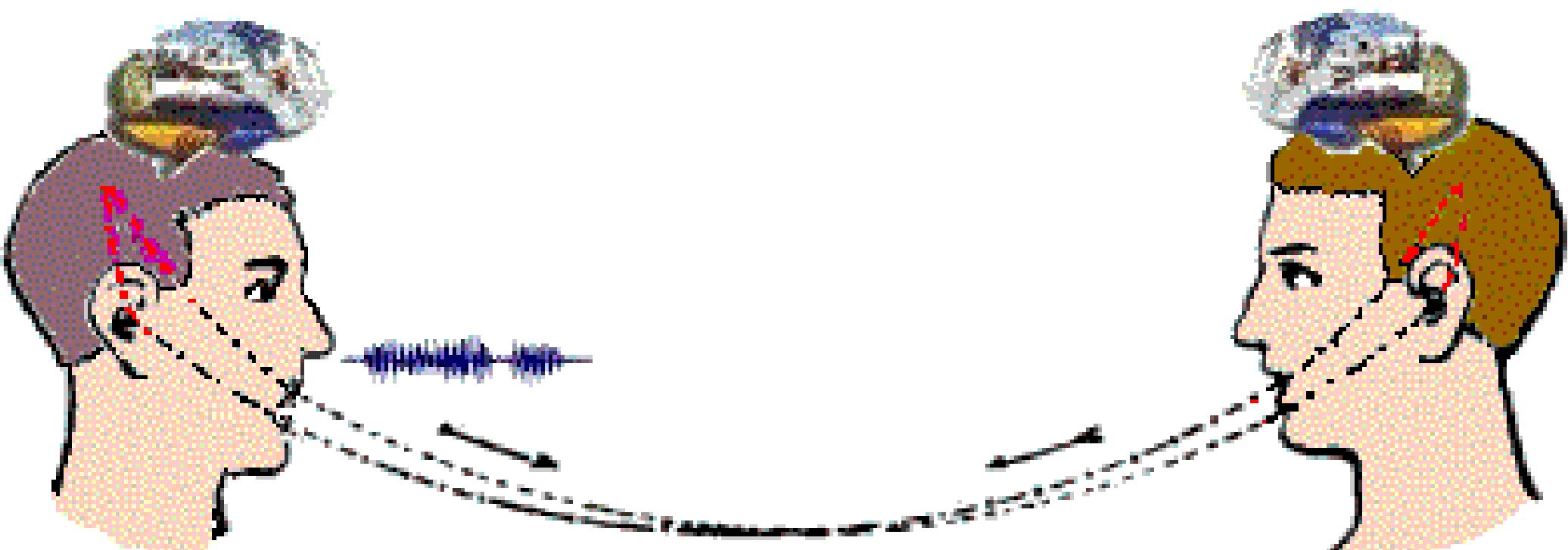


¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Para *resolver su problema*, lo que necesitamos es ***dialogar con él en su mismo idioma.***

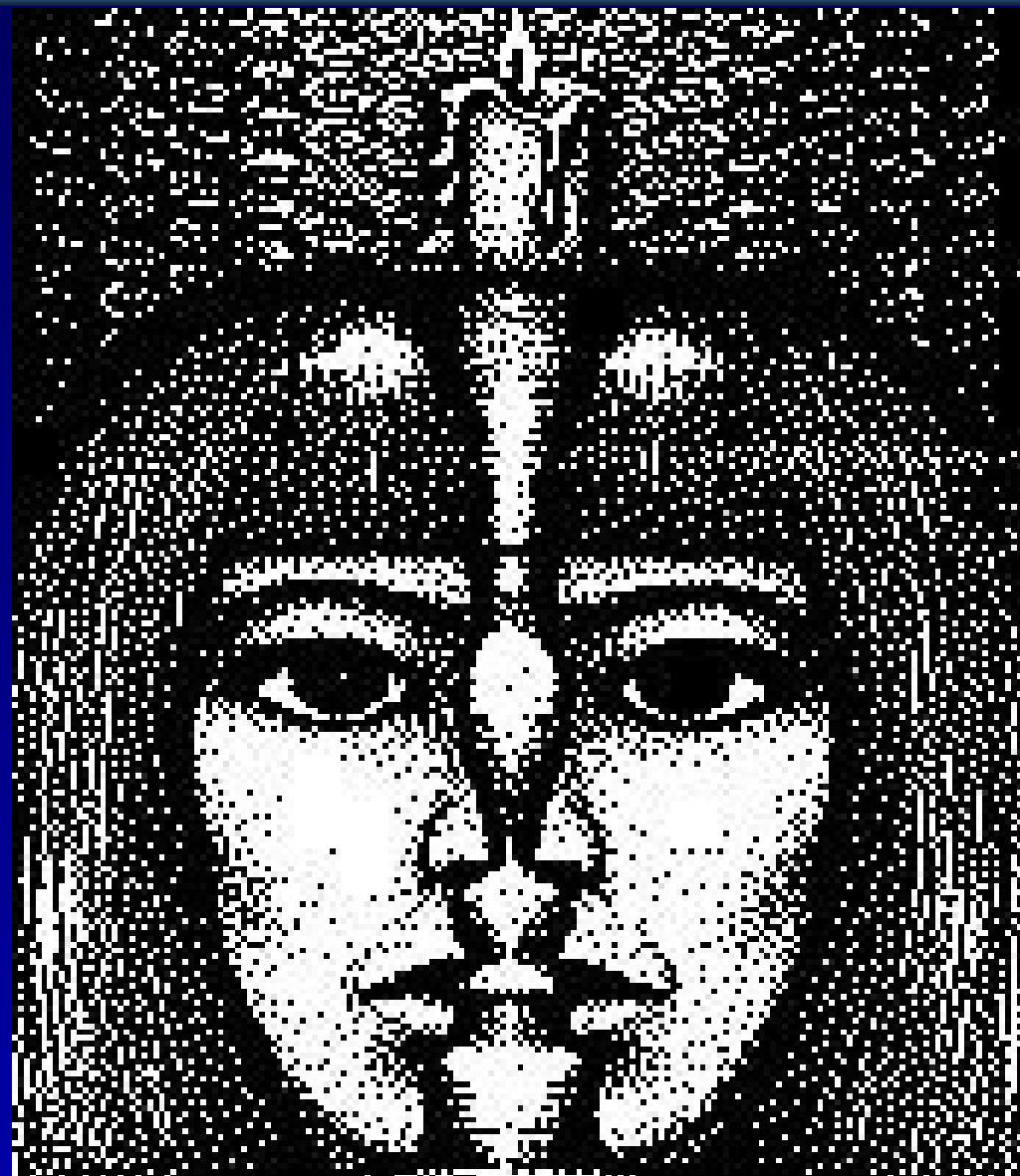




¿Por qué programamos?

Entonces nuestro **cliente** tiene un problema...

- Para *resolver su problema*, lo que necesitamos es **evitar ambigüedades:**





¿Por qué programamos?



Entonces nuestro **cliente** tiene un problema...

- Para ***resolver su problema***, lo que necesitamos es **evitar contradicciones**:





¿Cómo aprendemos?



Cliente:

- El profesor asume ese rol.

Problema:

- Se formula como un **enunciado** (*una descripción precisa de lo que se quiere lograr*)

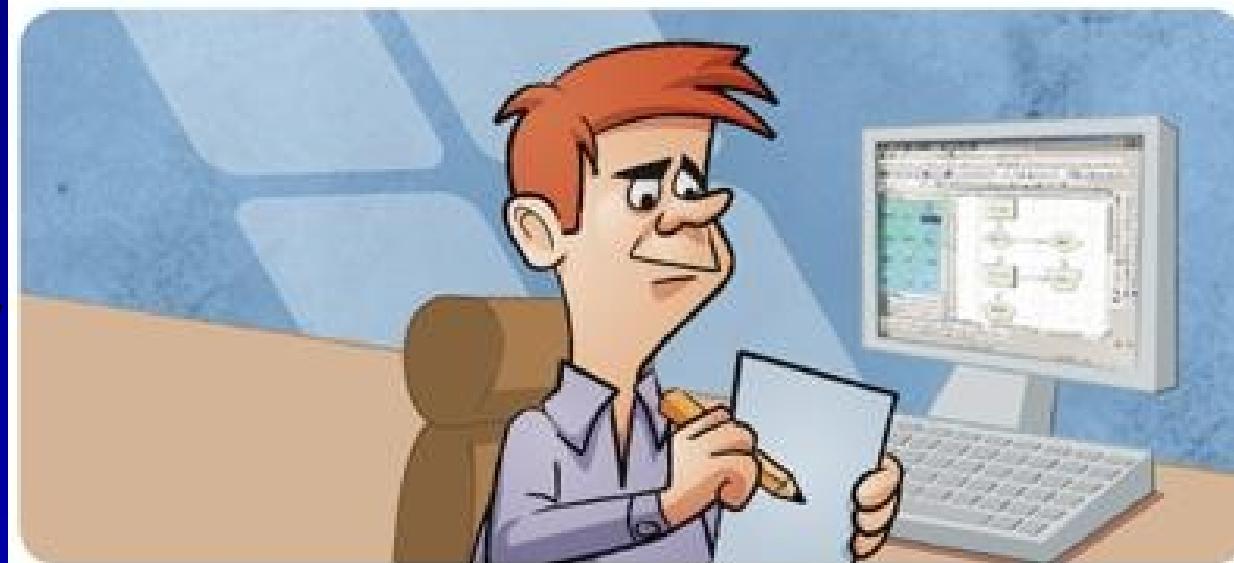




¿Cómo aprendemos?

Programador:

- El estudiante cumple ese rol.
- El **resultado** debe resolver el **problema**.
- De existir **ambigüedad** en el enunciado, *es a favor del alumno*.





¿Cómo aprendemos?



Cliente:

- ¿Qué **límites** tiene?
 - La imaginación...





Cliente:

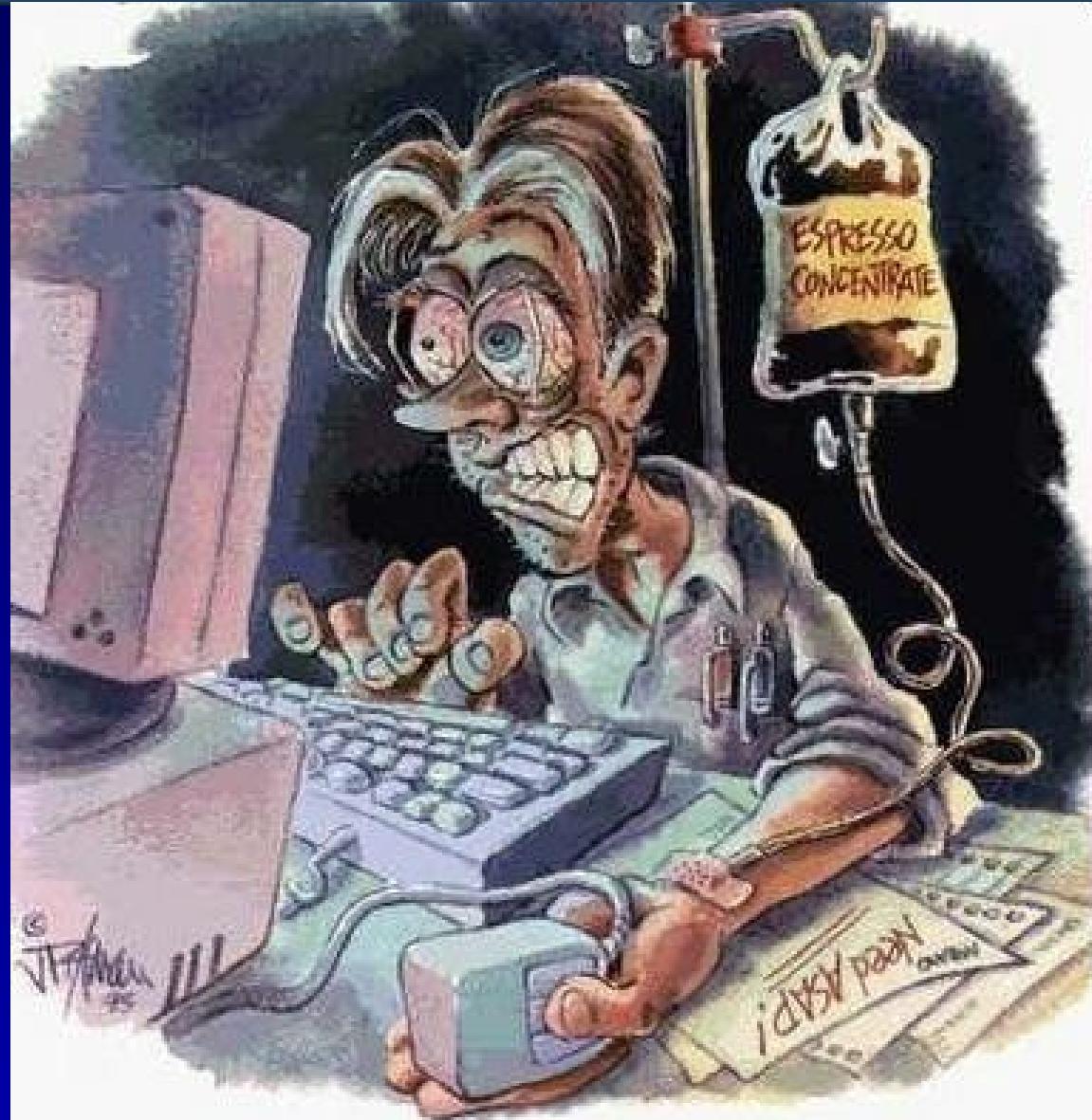
- ¿Qué **límites** tiene?
 - El dinero...





Programador:

- ¿Qué **límites** tiene?
 - El plazo...





Programador:

- ¿Qué límites tiene?
 - El precio...

Consulta de Precios de Venta y Stock

Productos con la Descripción : 'MEM'

F5: Ir a Búsqueda de Productos

Total de Stock: 0,00

[RePag] < Anteriores 20 | [AvPag] Siguientes 20 > | Salir

Código	Descripción	Stock	LISTA ...	LISTA ...	LISTA A...	OFERT...	[Lista d ^]
123	MEMORIA DDR 128 333 SP...	0	U\$S 18,...	U\$S 18,...	U\$S 22,2		\$ 15,56
44	MEMORIA DDR 128 PC 266...	-2	U\$S 13,...	U\$S 12,...	U\$S 17,...		\$ 17,91
7	MEMORIA DDR 256 333 NA...	0	U\$S 33,5...	U\$S 32,...	U\$S 35		\$ 33,03
158	MEMORIA DDR 256 333 SP...	0	U\$S 36,8	U\$S 36	U\$S 39,		\$ 32,22
8	MEMORIA DDR 256 400 AE...	-7	U\$S 25,7	U\$S 25,...	U\$S 26,1	U\$S 24,...	U\$S 22,
290	MEMORIA DDR 256 400 SP...	1	U\$S 26,...	U\$S 25,...	U\$S 34	U\$S 24,...	U\$S 19,
9	MEMORIA DDR 512 400 NA...	0	U\$S 51,5	U\$S 50,...	U\$S 55	U\$S 50	U\$S 20,
162	MEMORIA DDR 512 400 SA...	0	U\$S 52,...	U\$S 51,...	U\$S 55	U\$S 42	U\$S 39,
255	MEMORIA DIMM 128MB PC...	0	U\$S 27,...	U\$S 26,7	U\$S 32		
256	MEMORIA DIMM 128MB PC...	0	U\$S 19,...	U\$S 19,6	U\$S 22		
208	MEMORIA DIMM 256 PC 133...	0	U\$S 41,...	U\$S 40	U\$S 45		\$ 36,04
67	MEMORIA SD 128MB	0	U\$S 20,...	U\$S 20,...	U\$S 25		\$ 15,03
96	MEMORIA SD 256	0	U\$S 30,...	U\$S 32,...	U\$S 34,...		
352	MEMORIA SD 256MB COR.1	0	U\$S 48,...	U\$S 47,...	U\$S 55		\$ 39,82
163	MEMORY STICK SONY	0	U\$S 48,...	U\$S 47,...	U\$S 55		



¿Cómo aprendemos?



Programador y cliente:

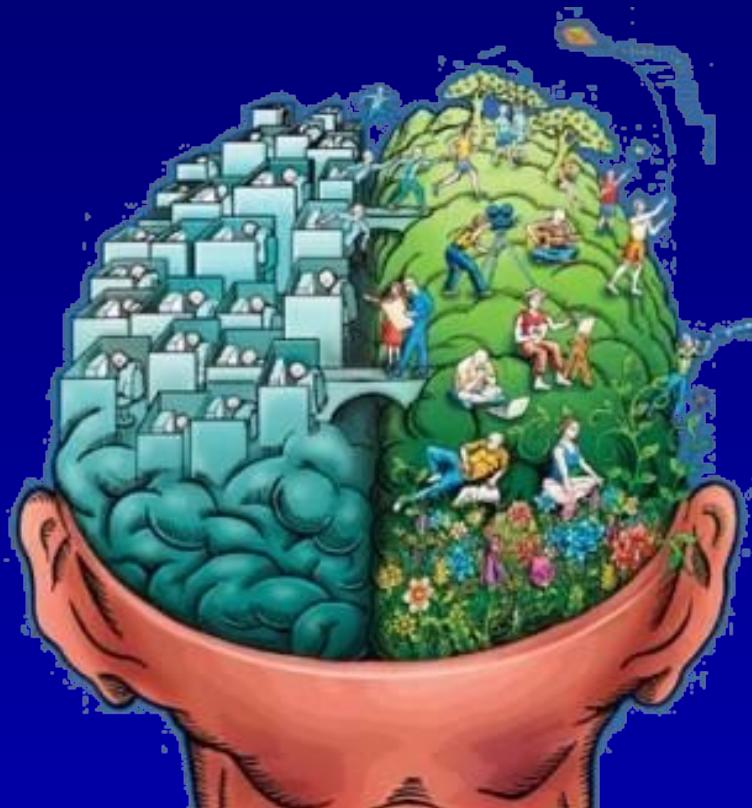
- Si son límites compatibles se firma un contrato.





Los problemas se dividen en dos grandes categorías:

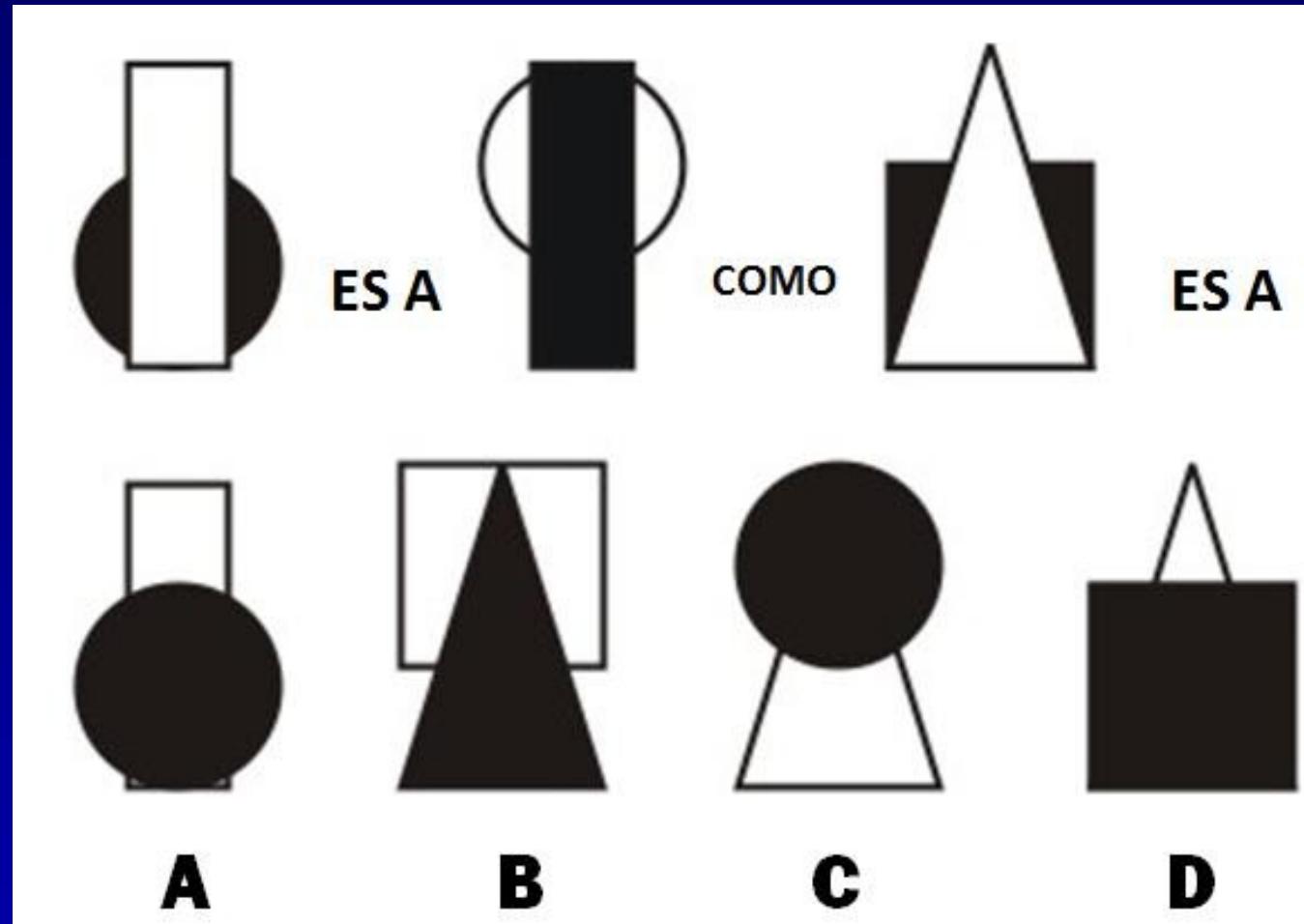
- Los que no tienen solución
- Los que sí tienen solución siempre (algorítmicos)
- Los que tienen solución a veces (heurísticos).





Nos interesan los algorítmicos pero hay estrategias comunes para entender el problema:

- Por analogía:
¿existe un problema similar que sepamos resolver?





Nos interesan los algorítmicos pero hay estrategias comunes para entender el problema:

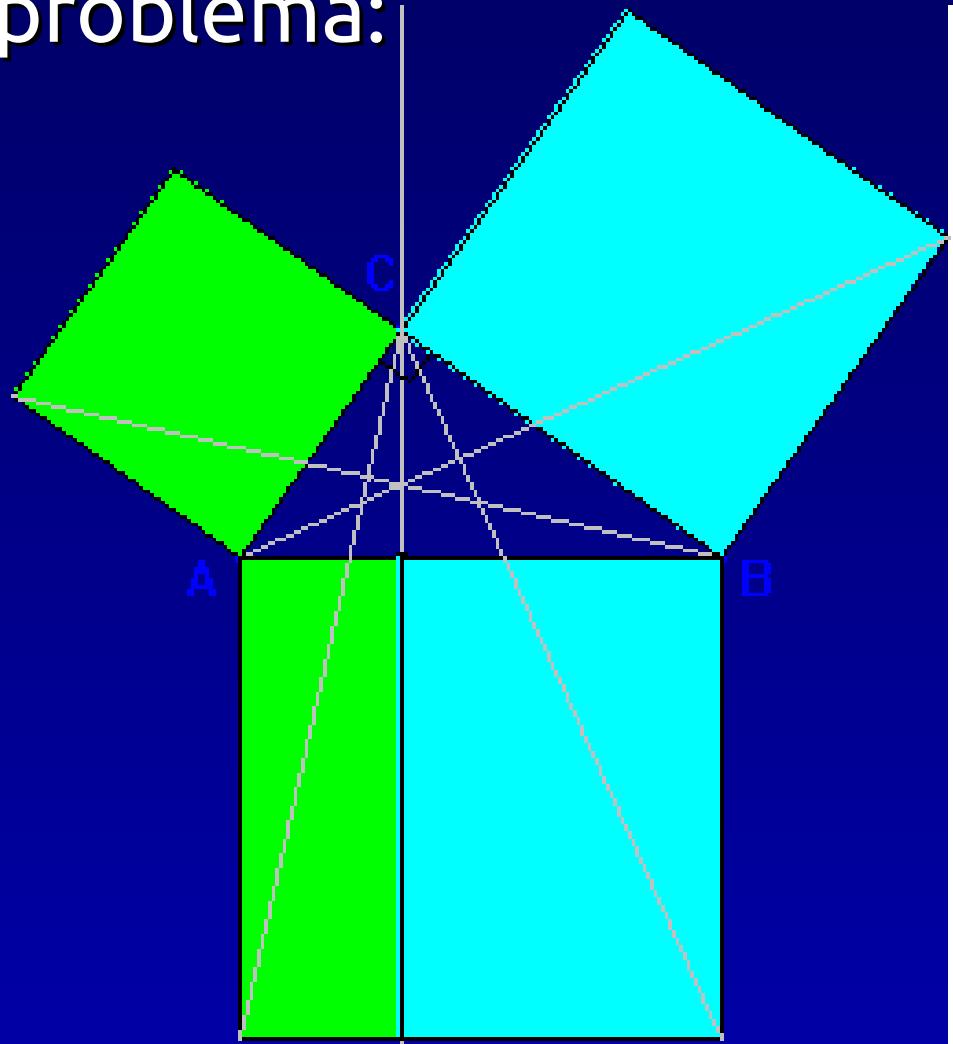
- Por generalización:
¿existen elementos comunes?





Nos interesan los algorítmicos pero hay estrategias comunes para entender el problema:

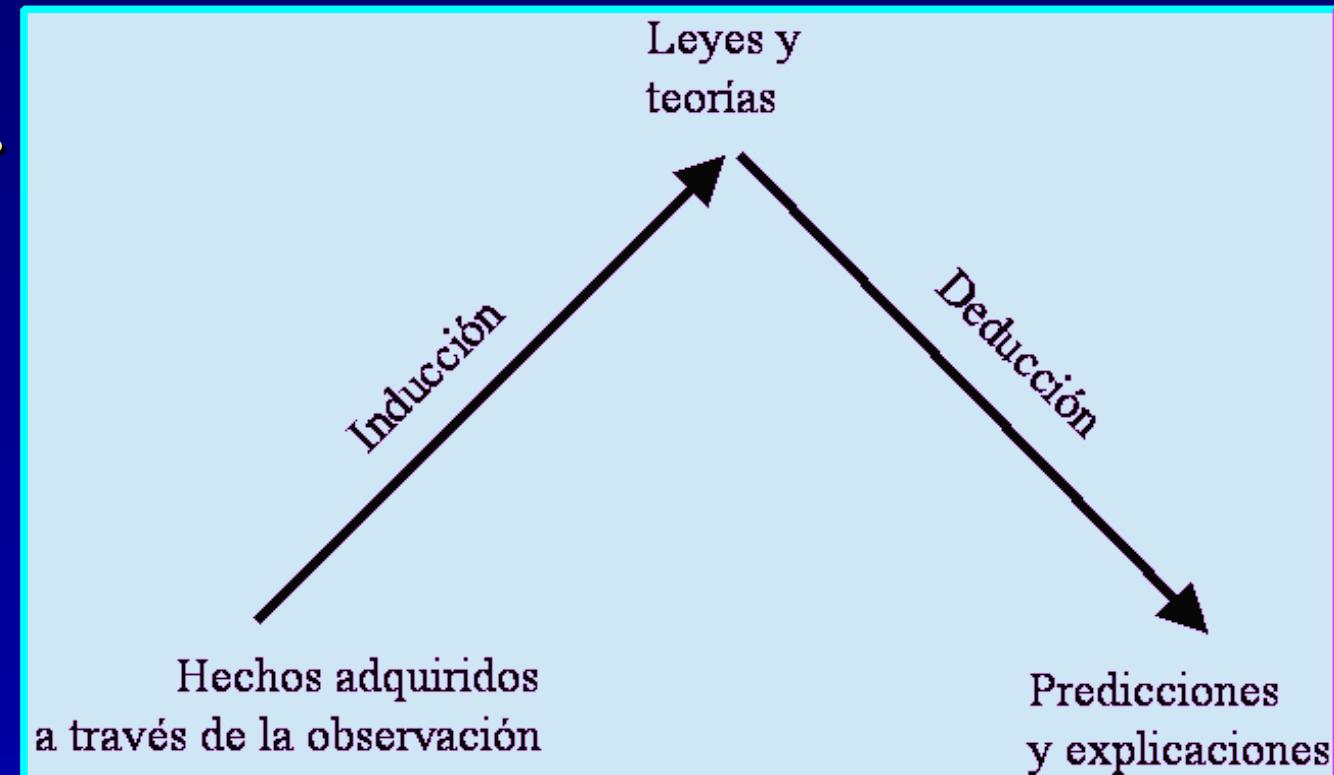
- Por inducción:
De lo particular a lo general.





Nos interesan los algorítmicos pero hay estrategias comunes para entender el problema:

- Por deducción:
De lo general
a lo particular.





Nos interesan los algorítmicos pero hay estrategias comunes para entender el problema:

- Por **división**:

De lo más grande a lo más chico.





Nos interesan los algorítmicos pero hay estrategias comunes para entender el problema:

- Por simplificación:

- «Si no puedes resolver ese problema, entonces existe un problema más sencillo que éste que sí podrás resolver: encuéntralo»
- «Si no puedes resolver el problema propuesto, intenta resolver primero un problema relacionado. ¿Podrías imaginar un problema relacionado más accesible?»
 - George Pólia, matemático húngaro (1887-1985).





Los algorítmicos:

- Garantizan alcanzar la solución
- Tienen una longitud finita de instrucciones
- Se ejecutan paso a paso
- Pueden ser:
 - Computables
 - No Computables





El proceso de diseñar un programa es, esencialmente, un proceso creativo. Sin embargo, hay una serie de pasos comunes a seguir:

- Análisis del problema
- Diseño del algoritmo
- Codificación
- Compilación y Ejecución
- Verificación
- Depuración
- Documentación





El proceso de diseñar un programa es, esencialmente, un proceso creativo. Sin embargo, hay una serie de pasos comunes a seguir:

- **Análisis del problema**
- Diseño del algoritmo
- Codificación
- Compilación y Ejecución
- Verificación
- Depuración
- Documentación





El proceso de diseñar un programa es, esencialmente, un proceso creativo. Sin embargo, hay una serie de pasos comunes a seguir:

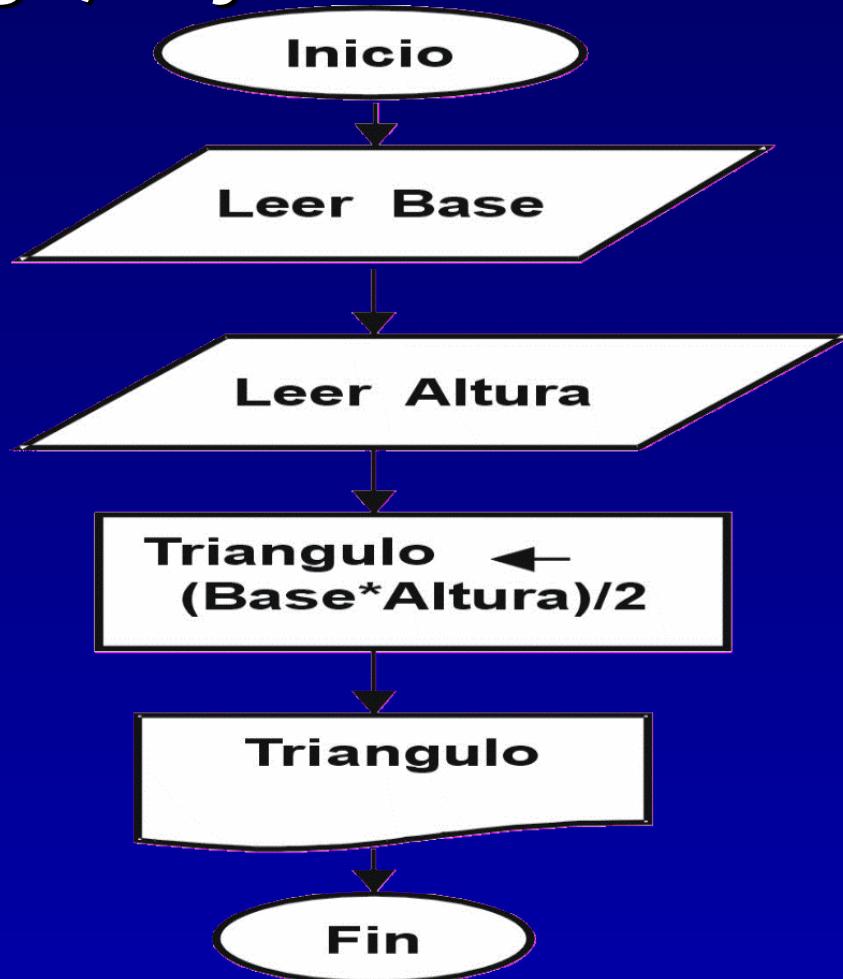
- **Análisis del problema**
 - **Depuración del enunciado**
- Diseño del algoritmo
- Codificación
- Compilación y Ejecución
- Verificación
- Depuración
- Documentación





El proceso de diseñar un programa es, esencialmente, un proceso creativo. Sin embargo, hay una serie de pasos comunes a seguir:

- Análisis del problema
- **Diseño del algoritmo**
- Codificación
- Compilación y Ejecución
- Verificación
- Depuración
- Documentación





El proceso de diseñar un programa es, esencialmente, un proceso creativo. Sin embargo, hay una serie de pasos comunes a seguir:

- Análisis del problema
- Diseño del algoritmo
- **Codificación**
- Compilación y Ejecución
- Verificación
- Depuración
- Documentación

The screenshot shows the Zinjal IDE interface. The main window displays a C++ code file named 'main.cpp'. The code is as follows:

```
1 // Archivo de traducción de seudocódigo a C++
2 #include <program1.h>
3 /*
4  * Enunciado: Dada la altura y base de un triángulo, calcular la superficie del mismo.
5  */
6 principal
7 // Declaraciones:
8 real base, altura;
9 limpiar;
10 //Código:
11 pausa;
12 finPrincipal
13
14
```

The code includes comments explaining the problem statement and the steps of the algorithm. The interface includes a toolbar, menu bar, and various tool windows at the bottom.

Zinjal - Presentacion

Archivo Editar Ver Ejecucion Depuracion Herramientas Ayuda

main.cpp*

```
1 // Archivo de traducción de seudocódigo a C++
2 #include <program1.h>
3 /*
4     Enunciado: Dada la altura y base de un triángulo, calcular la superficie del mismo.
5 */
6 principal                                // Unidad de programa principal
7 // Declaraciones:
8 real base, altura;
9 limpiar;                                    // Limpia la pantalla.
10 //Código:
11 pausa;                                      // Pausa antes de finalizar.
12 finPrincipal                                // Fin de unidad de programa principal.
```

Compilador Ayuda/Busqueda Trazado Inverso Inspecciones





El proceso de diseñar un programa es, esencialmente, un proceso creativo. Sin embargo, hay una serie de pasos comunes a seguir:

- Análisis del problema
- Diseño del algoritmo
- **Codificación**
- **Compilación y Ejecución**
- **Verificación**
- **Depuración**
- Documentación

The screenshot shows the Zinjal IDE interface. The main window displays the following C++ code:

```
principal
// Declaraciones:
real base, altura;
// Datos (a leer)
// Limpia la pantalla.
limpiar;
// Código:
leerM(base, "Base:");
leerM(altura, "Altura:");
mostrar << "La superficie del triangulo es:" << base * altura / 2 << salto;
pausa;
finPrincipal
```

The terminal window shows the message: "Operation not permitted". Below the code editor are two inspection windows: "Trazado Inverso" and "Inspecciones".

Nivel	Función	Archivo	Línea	Argumentos
0	main	/home/carlos/Zir	9	argc=1, argv=0xbfffff74

Nivel	Expresión	Tipo	Valor	Formato	catchPoint
*	base	<<?>>	0	<<?>>	no
*	altura	<<?>>	3.98878799e-34	<<?>>	no

