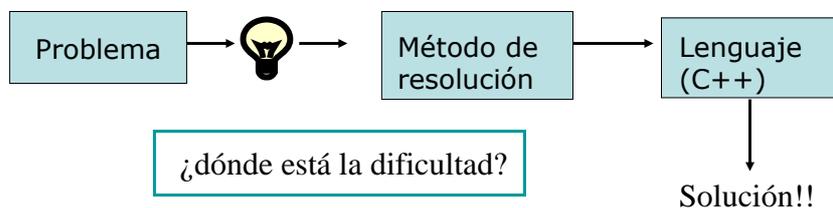


# Fundamentos de Programación (I)

Algoritmos  
Entrada y Salida en C++

## Fundamentos de programación

- **Objetivo:** Traducir nuestras ideas a un nuevo lenguaje de programación para resolver un problema.



...en encontrar el método que permita resolver el problema

## Algoritmos

- **Algoritmo:** conjunto ordenado de operaciones que nos permite resolver un problema.
- Características que debe cumplir:
  - Número finito de pasos.
  - Tiempo finito.
  - Definición precisa de todas las operaciones.
  - Interacción con el entorno. Tiene como mínimo una salida y puede tener entradas.



## Ejemplo de algoritmo sencillo

Ejemplo de Algoritmo "Buenos días", ejecutado por trabajadores/estudiantes todos los días:

1. Despertarse
2. Levantarse
3. Ducharse
4. Ponerse la ropa
5. Desayunar
6. Viajar hasta el trabajo/ universidad

El orden es muy importante

.. Si cambiamos el paso 3 al final llegaríamos empapados !!



La especificación del orden = **CONTROL** del programa

**Algoritmo + léxico → programa** = secuencia de operaciones especificadas en un lenguaje dado.

Un lenguaje de programación viene definido por un léxico, una sintaxis y una semántica.

**Léxico** : Conjunto de símbolos que se pueden utilizar en un lenguaje.

- **Identificadores** : nombres simbólicos que se darán a ciertos elementos del lenguaje (p. e. nombres de variables, tipos)
- **Constantes** : datos que no cambiarán su valor a lo largo del programa.
- **Operadores** : símbolos que representarán operaciones entre variables y constantes.
- **Instrucciones** : símbolos especiales que representarán estructuras de procesamientos
- **Comentarios** : Texto que se utiliza para documentar el programa

## Traducción algoritmo a lenguaje de programación

```
//Algoritmo para calcular precios de
//Pizzas unitario (por unidad de
//superficie)
•Leer dimensiones pizza circular (radio)

•Leer precio pizza circular

•Leer dimensiones pizza rectangular
(largo, ancho)

•Leer precio pizza rectangular
(prec_rect)

•Cálculo del precio unitario de pizza
circular.

•Cálculo del precio unitario de pizza
rectangular.

•Imprimir precio unitario pizza
Rectangular

•Imprimir precio unitario pizza circular
```

```
//Programa que nos dice la mejor elección para comprar una pizza
//Se compara una pizza rectangular con una redonda.
#include <iostream>
#include <stdlib.h>
const int PI 3.141516
int main()
{
//Declaración de variables
float radio, largo, ancho;
int prec_rect, prec_red;
int prec_unirect, prec_unired;
//Presentación del programa al usuario
cout << "Bienvenido a la unidad de consumidores de Pizza\n";
//Lectura de datos
cout << "Teclee el radio de una pizza circular en cm: ";
cin >> radio;
cout << "Teclee el precio de una pizza redonda (pts): ";
cin >> prec_red;
cout << "Teclee el largo y ancho de una pizza rectangular: ";
cin >> largo >> ancho;
cout << "Teclee el precio de una pizza rectangular (pts): ";
cin >> prec_rect;
//Bloque de cálculos
prec_unired = prec_red / (PI* radio * radio);
prec_unirect = prec_rect / (largo * ancho);
//Impresión de resultados
cout << "Precio por unidad de pizza circular: " << prec_unired << endl;
cout << "Precio por unidad de pizza rectangular: " << prec_unirect << endl;
return 0;
}
```

Vamos a completar la definición de algoritmo definiendo claramente los datos de entrada y salida (resultados) necesarios

- Como son los datos de entrada de un algoritmo?
  - Variables de entrada.
- Como es la salida? Como se muestran los resultados?
  - Variables de salida.

... que es una variable?  
... de que tipo pueden ser?

## Entrada y Salida en C++: Introducción

- Flujo de entrada: serie de entradas que alimentan un ordenador para que el programa las utilice.
- Flujo de salida: serie de salidas que el programa genera.
- Suponemos que:
  - Entrada estándar: teclado.
  - Salida estándar: pantalla

## Salidas con cout

- Se puede enviar a la pantalla cualquier combinación de variables y cadenas:

```
cout << num_dulces << "dulces\n";
```

```
cout << num_dulces;  
cout << "dulces" << endl;
```

- Se pueden incluir expresiones aritméticas:

```
cout << "El precio total es:" << (precio1 + precio2);
```

Operador de inserción

## Salidas con cout

- Secuencias de caracteres especiales: todas ellas comienza con el carácter '\'.  
• Algunas secuencias de caracteres:

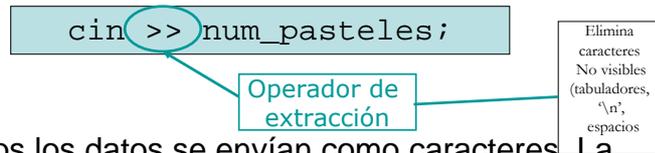
<code>\n</code>	Nueva línea
<code>\t</code>	Tabulación horizontal
<code>\\</code>	Diagonal invertida
<code>\"</code>	Comillas dobles

- Ejemplo:

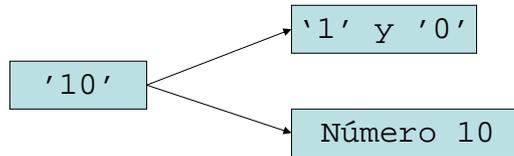
```
//enviar línea en blanco (pantalla  
cout << "\n";
```

## Entrada con cin

- cin es el flujo de entrada estándar (supondremos teclado). Ejemplo:



- Todos los datos se envían como caracteres. La interpretación depende del programa:



## Entrada y salida de caracteres

- Funciones miembro o métodos:
  - **get**: permite leer un carácter de entrada y guardarlo en una variable tipo carácter (char).

```
char siguiente_simbolo;  
cin.get(siguiente_simbolo);
```

- **put**: es análoga a la función miembro get sólo que se emplea para salida. Se envía a la salida un carácter.

```
char siguiente_simbolo='a';  
cout.put(siguiente_simbolo);
```

- **ignore**: ignora num caracteres mientras no se encuentre el carácter delim

```
cin.ignore(int num, int delim);  
cin.ignore(); //ignora un caracter
```

## Entrada y salida de caracteres

- Comparación entre `cin >>` y `cin.get`:

```
cin.get char c1,c2,c3,c4;
//Introduce 4 caracteres separados por espacios
cin.get(c1);
cin.get(c2);
cin.get(c3);
cin.get(c4);

//Imprimo los caracteres leidos
cout << "Los cuatro caracteres leidos son:"<< endl;
cout.put(c1);
cout.put(c2);
cout.put(c3);
cout.put(c4);
```

## Entrada y salida de caracteres

- Comparación entre `cin >>` y `cin.get`:

```
cin >> char c1,c2,c3,c4;
//Prueba con cin
//Introduce 4 caracteres separados por espacios
cin >> c1;
cin >> c2;
cin >> c1;
cin >> c2;

//Imprimo los caracteres leidos
cout << "Los cuatro caracteres leidos son:"<< endl;
cout << c1 << c2 << c3 << c4;
```

## Comparación

```

E:\clases\FundaProg\curso02-03\ejemplos\entrada.exe
Introduce cuatro caracteres separados por espacios
1 2 3 4
Los cuatro caracteres leídos son:
1 2 Presione una tecla para continuar . . . _

E:\clases\FundaProg\curso02-03\ejemplos\entrada2.exe
Introduce 4 caracteres separados por espacios
1 2 3 4
Los cuatro caracteres leídos son:
1234
Presione una tecla para continuar . . . _
    
```

## ¿Podemos sumar dos caracteres?

```

C:\Documents and Settings\Iliok\Escritorio\Clases\F
Dona'm 2 caracteres: + +
+
+-----
U
Presione una tecla para continuar . . . _

#include <iostream>
int main(int argc, char *argv[])
{
    char a, b, res;

    cout << "Dona'm 2 caracteres: ";
    cin >> a >> b;

    res = a + b;

    cout << a << endl;
    cout << b << endl;
    cout << "-----" << endl;
    cout << res << endl;
    system("pause");

    return 0;
}
    
```

ASCII	Hex	Símbolo	ASCII	Hex	Símbolo	ASCII	Hex	Símbolo	ASCII	Hex	Símbolo
0	0	NUL	16	10	DLE	32	20	(espacio)	48	30	0
1	1	SOH	17	11	DC1	33	21	!	49	31	1
2	2	STX	18	12	DC2	34	22	"	50	32	2
3	3	ETX	19	13	DC3	35	23	#	51	33	3
4	4	EOT	20	14	DC4	36	24	\$	52	34	4
5	5	ENQ	21	15	NAK	37	25	%	53	35	5
6	6	ACK	22	16	SYN	38	26	&	54	36	6
7	7	DEL	23	17	ETB	39	27	'	55	37	7
8	8	BS	24	18	CAN	40	28	(	56	38	8
9	9	TAB	25	19	EM	41	29	)	57	39	9
10	A	LF	26	1A	SUB	42	2A	*	58	3A	:
11	B	VT	27	1B	ESC	43	2B	+	59	3B	;
12	C	FF	28	1C	FS	44	2C	,	60	3C	<
13	D	CR	29	1D	GS	45	2D	.	61	3D	=
14	E	SO	30	1E	RS	46	2E	/	62	3E	>
15	F	SI	31	1F	US	47	2F	/	63	3F	?

ASCII	Hex	Símbolo									
64	40	@	80	50	P	96	60	`	112	70	p
65	41	A	81	51	Q	97	61	a	113	71	q
66	42	B	82	52	R	98	62	b	114	72	r
67	43	C	83	53	S	99	63	c	115	73	s
68	44	D	84	54	T	100	64	d	116	74	t
69	45	E	85	55	U	101	65	e	117	75	u
70	46	F	86	56	V	102	66	f	118	76	v
71	47	G	87	57	W	103	67	g	119	77	w
72	48	H	88	58	X	104	68	h	120	78	x
73	49	I	89	59	Y	105	69	i	121	79	y
74	4A	J	90	5A	Z	106	6A	j	122	7A	z
75	4B	K	91	5B	[	107	6B	k	123	7B	{
76	4C	L	92	5C	\	108	6C	l	124	7C	
77	4D	M	93	5D	]	109	6D	m	125	7D	}
78	4E	N	94	5E	^	110	6E	n	126	7E	~
79	4F	O	95	5F	_	111	6F	o	127	7F	o

Tabla de códigos ASCII - Formato de caracteres estándares