

FUNDAMENTOS DE PROGRAMACIÓN





**¿Qué aprendimos
la sesión anterior?**





Introducción a la programación: programas traductores, programación estructurada, Variables y Tipos de datos

Semana 01

ucontinental.edu.pe



Propósito

Crea programas con variables y tipos de datos para dar solución a problemas planteados, haciendo uso del Dev C+.





Agenda del día

- 1 Programas traductores
- 2 Lenguaje de Programación
- 3 Programación Estructurada
- 4 Variables y Tipos de Datos
- 5 Funciones básicas
- 6 Operadores y expresiones
- 7 Estructura de un programa
- 8 Estructura de control





Programas traductores



Programa

Un programa es un conjunto de instrucciones individuales o código fuente ordenado de forma lógica con el objetivo de realizar una tarea, para finalmente obtener una pieza entera de software ejecutable.

```
255 function updatePhotoDescription() {  
256   if (descriptions.length > (page * 9) + (currentImage subtracting))  
257     document.getElementById('bigImageDesc').innerHTML += descriptions[currentImage subtracting] + '  
258   }  
259 }  
260  
261 function updateAllImages() {  
262   var i = 1;  
263   while (i < 10) {  
264     var elementId = 'foto' + i;  
265     var elementIdBig = 'bigImage' + i;  
266     if (page * 9 + i - 1 < photos.length) {  
267       document.getElementById( elementId ).src = 'photos/' + photos[page * 9 + i - 1].src;  
268       document.getElementById( elementIdBig ).src = 'photos/' + photos[page * 9 + i - 1].src;  
269     } else {  
       document.getElementById( elementId ).src = 'photos/' + photos[photos.length - 1].src;  
       document.getElementById( elementIdBig ).src = 'photos/' + photos[photos.length - 1].src;  
     }  
     i++;  
   }  
}
```



Programas traductores

Traducen los programas de código fuente a código máquina

Un código fuente se puede convertir en ejecutable a través de dos formas:



Interpretes.



Compiladores.



a) Intérprete

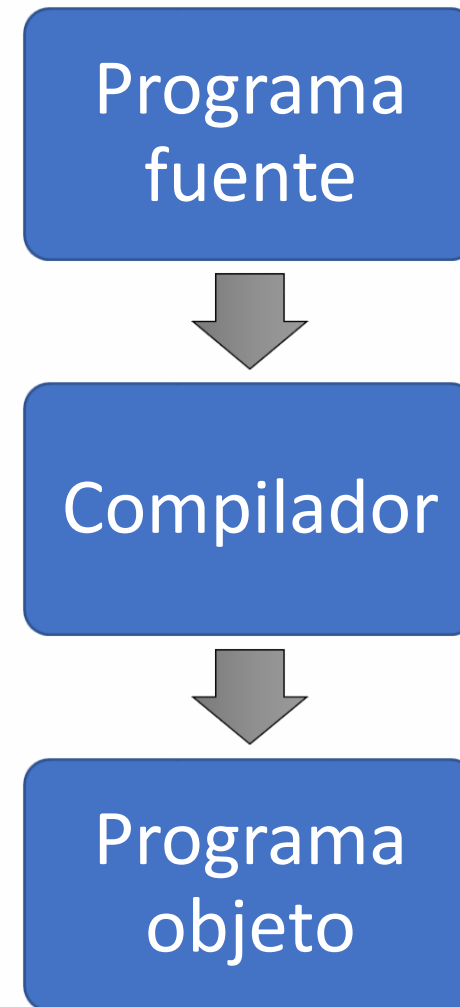
- Analiza el código fuente y lo ejecuta directamente.
- Convierte el código fuente en instrucciones de computadora (lenguaje máquina), y la computadora actúa con esas instrucciones inmediatamente.
- Es como un traductor humano, que conforme a lo que está escuchando va ejecutando, sin generar ningún escrito, es decir que sobre la marcha va traduciendo.





b) Compilador

- Un compilador analiza el programa y lo traduce al lenguaje maquina.
- Traslada el código en programas, los cuales pueden ejecutarse tiempo después.
- La acción fundamental de los compiladores es equivalente a la de un traductor humano, que toma nota de lo que esta escuchando y reproduce por escrito después.





b) Compilador

La compilación es el proceso de traducción del programa fuente a programa objeto (traducido a código máquina), a través de un programa enlazador, para conducir al programa ejecutable





Lenguaje de Programación



Lenguaje de programación

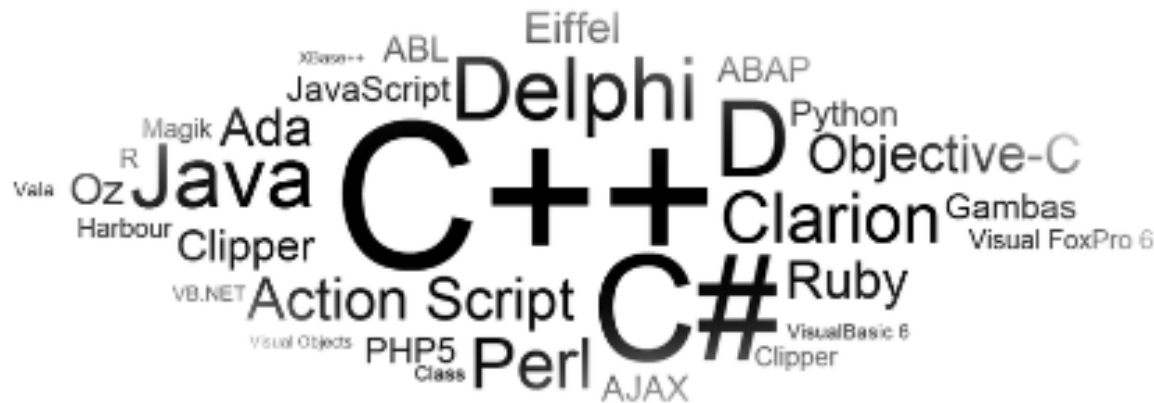
El origen del lenguaje de programación se debe a Ada Lovelace que está considerada como la primera programadora de computadoras. Dando su nombre al lenguaje de programación Ada.

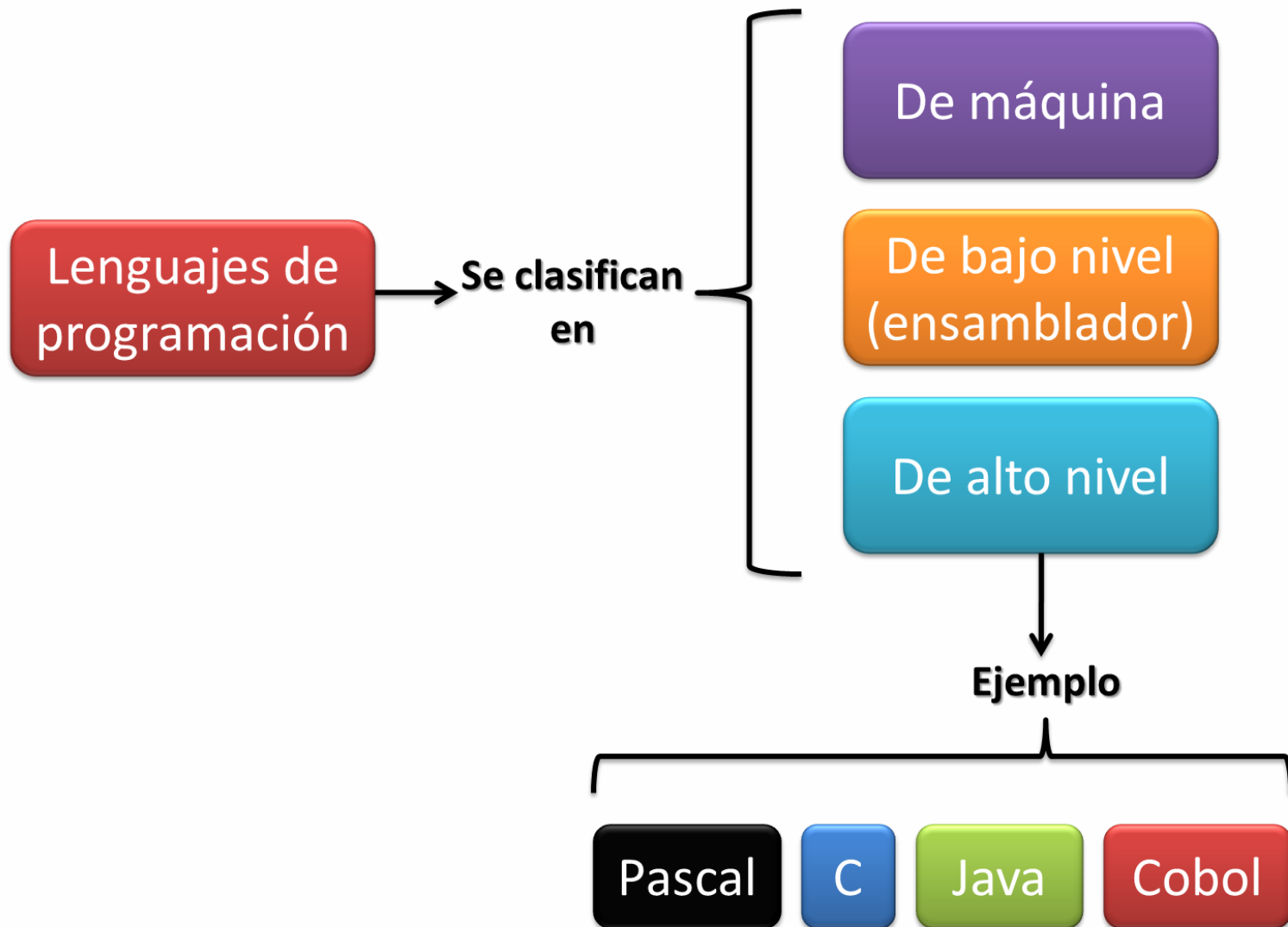




Lenguaje de programación

El lenguaje de programación tiene la capacidad de especificar, de forma precisa, cuáles son los datos que debe trabajar un equipo informático, de qué modo deben ser conservados o transferidos dichos datos y qué instrucciones debe poner en marcha la computadora ante ciertas circunstancias.







C++

Es un lenguaje de programación propuesto en la década de 1980 por Bjarne Stroustrup. La intención de su creación fue el extender al lenguaje de programación C mecanismos que permiten la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido.

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int input_var = 0;
6      // Entra a un Ciclo Do-While el cual se mantiene hasta que
7      // se introduce un numero o -1 (salir del programa)
8      do {
9          cout << "Ingrese un numero o -1 (para Salir)";
10         if (!(cin >> input_var)) {
11             // CIn Acepta una entrada por el teclado si esta entrada no es igual
12             // a un numero entero imprime el siguiente mensaje y termina el programa
13             cout << "Has introducido un caracter no numerico " << endl;
14             break;
15             //Sale del ciclo y termina el programa
16         }
17         if (input_var != -1) {
18             // Si el Valor introducido por el teclado es un numero entero
19             // imprime el numero introducido
20             cout << "Numero ingresado es: " << input_var << endl;
21         }
22     } while (input_var != -1);
23     //Condicion al escribir -1 el programa terminara;
24     cout << "Terminado" << endl;
25     return 0;
26 }
```





Programación estructurada



Programación estructurada

Esta metodología de programación utiliza un número limitado de estructuras de control que minimizan la complejidad de los programas y por consiguiente reducen los errores. Los programas deben estar dotados de una estructura.





Programación estructurada

La programación estructurada incorpora:

Recursos abstractos:

- Descompone acciones complejas en acciones simple.

Diseño descendente:

- Descompone el problema en etapas o estructuras jerárquicas.

Estructuras de control:

- Uso de estructuras secuenciales, de selección y repetición.



Programación orientada a objetos

- La programación orientada a objetos (POO), es una filosofía de programación que se basa en la utilización de objetos. El objetivo de la POO es imponer una serie de normas de desarrollo que aseguren y faciliten la mantenibilidad y reusabilidad del código.
- Para efectos de esta asignatura, no se desarrollará a profundidad este tema.





Variables y Tipos de Datos



Variable

Es un espacio de memoria de computadora para guardar información, en la cual se puede grabar un valor y que se puede recuperar más tarde.

Ejemplo:

Variable	Datos
nombre	"María" "Juan" "Carlos"
sexo	'M' 'F'
precio	100.56 1200.456
cantidad	10 24 100



Tipos de datos

Tipo de dato	Tipo de dato en C++	Tamaño	Rango	Datos
Entero	int	2 bytes	-32768 a 32767	10 25 -32768
Carácter	char	1 byte (8 bits)	---	'A' 'B' '1'
Real	float	4 bytes	3.4E-38 a 3.4E38	3.559 80095.500089
Real doble	double	8 bytes	1.7E-308 a 1.7E308	1.7×10^{308} 0.56666 -8563089.6



Ejemplo de declaración de variables

Ejemplo	Descripción
int nota;	La variable nota almacena el tipo de dato int
char sexo;	La variable sexo almacena el tipo de dato char, puede ser 'M' o 'F'
double precio;	La variable precio almacena el tipo de dato double, para valores decimales
float prom;	La variable prom almacena el tipo de dato float, para valores decimales
int x=0;	La variable x almacena el tipo de dato int, y se inicializa con el valor de 0, en la misma línea de instrucción
int a,b;	La variable a y b almacena el tipo de int
int x; x=0;	La variable x almacena el tipo de dato int, y se inicializa con el valor de 0 , en dos líneas de instrucción.



Funciones básicas



Funciones básicas

Las funciones siguientes emplean la librería `#include <math.h>`

Sintaxis de funciones propias de c++	Expresión matemática	Ejemplo en c++
pow (base,exponente)	$Y = 2^3$	y=pow(2,3)
	$Y = x^3$	Y=pow(x,3)
	$\sqrt{16} = 16^{1/2}$	z=pow(16,1/2.0)
sqrt(numero)	$Z = \sqrt{16}$	z=sqrt(16)
	$Z = \sqrt{x + y}$	w=sqrt(x+y)



Operadores y expresiones



Operadores aritméticos

Operador aritmético	Operador en C++	Ejemplo
Suma	+	<code>s=n1+n2;</code>
Resta	-	<code>r=n1-n2;</code>
Multiplicación	*	<code>m=n1*n2;</code>
División	/	<code>d=n1/n2;</code>
Módulo (Residuo)	%	<code>resto=n1%n2;</code>



Operadores relacionales

Operador relacional	Operador en C++	Ejemplo
Igualdad	<code>==</code>	<code>n1==n2</code>
Diferente	<code>!=</code>	<code>n1!=n2</code>
Mayor que	<code>></code>	<code>n1>n2</code>
Menor que	<code><</code>	<code>n1<n2</code>
Mayor o igual que	<code>>=</code>	<code>n1>=n2</code>
Menor o igual que	<code><=</code>	<code>n1<=n2</code>



Operadores lógicos

Operador lógico	Operador en C++	Ejemplo
And – Y	&&	(n>0)&&(n<=100)
Or – O		(n<=0) (n>100)
Not – No	!	!(n==0)



Operadores de asignación

Operador de asignación	Operador en C++	Ejemplo
Incremento	++	<pre>x=5; x++; //x=x+1 //x=6</pre>
Decremento	--	<pre>x=5; x--; //x=x-1 //x=4</pre>
Asignación simple	=	<pre>x=5 x=x+3 // x+3 se asigna a x // x=8</pre>



Expresiones

- Una expresión es un conjunto de términos que representan una cantidad algebraica o algún tipo de cálculo que puede contener operadores.

Total=a+c/d-b

No es lo
mismo

Total=(a+c)/(d-b)

Fórmula	Ejemplo
$z = \frac{a+c}{d-b}$	<code>z = (a+c) / (d-b);</code>
$z = \frac{a+c^3}{d-5}$	<code>z = (a+pow(c,3))/(d-5);</code>
$\sqrt{x+2} + \sqrt[3]{x^5}$	<code>z = sqrt(x+2)+pow(x,5/3.0);</code>



Palabras reservadas

- Existen algunas palabras reservadas para instrucciones propias del C++ o de sus librerías.
- Estas palabras clave usadas por el compilador, NO se deben usar como variables y otros.
- Algunas de estas palabras claves son: **if**, **while**, **for**, **main**, entre otras.



Secuencias de escape

Sec. de escape	Descripción	Sec. de escape	Descripción
\a	Alerta	\t	Tabulación horizontal
\b	Backspace	\v	Tabulación vertical
\f	Suministro de papel	\'	Comilla simple
\n	Salto de línea	\"	Comilla doble
\r	Caracter de regreso	\\	Barra inversa



Mecanismos de salida

Son aquellos mensajes que el programa utiliza para comunicarse con el exterior o con el usuario. El objeto “**cout**” se utiliza para la salida de datos y normalmente se envía a la pantalla.

```
//Ejemplo de mecanismos de salida  
#include<iostream>  
using namespace std;  
int main()  
{  
    cout<<"Universidad Continental"<<endl;  
    cout<<"Mi primer programa"<<endl;  
    return 0;  
}
```



Mecanismos de entrada

Nos permiten la interacción entre el usuario y el programa. El objeto “**cin**” se utiliza para extraer valores del flujo y almacenarlos en variables. Normalmente la entrada procede del teclado.

```
//Ejemplo de mecanismos de entrada y salida
#include<iostream>
using namespace std;
int main()
{
    int n1,n2;
    cout<<"Ingrese el primer numero: "<<endl;
    cin>>n1;
    cout<<"Ingrese el segundo numero: "<<endl;
    cin>>n2;
    cout<<"La suma es: "<< n1+n2 <<endl;
    return 0;
}
```



Estructura de un programa



Estructura de un programa

- Un programa esta formado por la cabecera y el cuerpo o módulo principal
 - En la cabecera se incluyen algunas rutinas predefinidas que hacen a la programación más sencilla, existen una gran cantidad de librerías disponibles para hacer un trabajo más estandarizado.
 - El cuerpo contiene la función principal, las funciones adicionales y las clases que se necesiten en el programa.



Sintaxis de un programa en C++

Cabecera

Cuerpo
o módulo
principal

```
1  /*PROGRAMA EJEMPLO
2   Calcula el area de un triangulo
3  */
4  #include<iostream>
5  using namespace std;
6  int main()
7  {    //Declaración de variables
8      int b,h;
9      double s;
10     //Entrada de datos
11     cout<<"Ingrese base: ";
12     cin>>b;
13     cout<<"Ingrese altura: ";
14     cin>>h;
15     //Proceso
16     s=(b*h)/2.0;
17     //Salida de datos
18     cout<<"El area del triangulo es : "<<s<<"\n";
19     system("pause");
20     return 0;
21 }
```

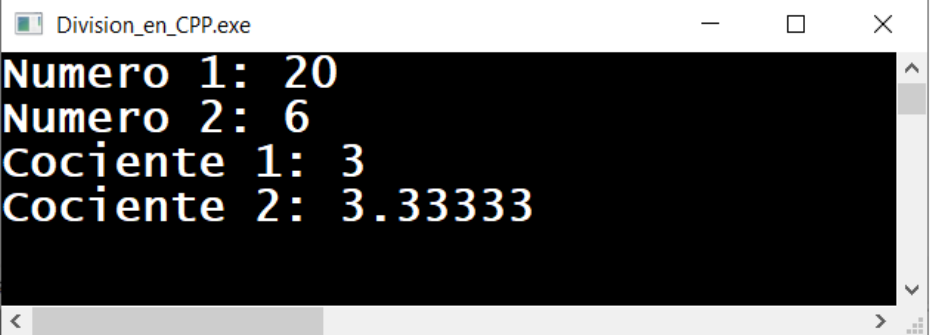
Comentario de más de una línea se declara entre:
/* comentario
De más de una línea*/

Comentario de una línea se declara:
//comentario 1 de una línea
//comentario 2 de una línea



Ejemplo: Elaborar un programa que solicite el ingreso de dos números enteros. Mostrar la división de los dos números en decimal y en entero.

```
1 // División en C++
2
3 #include <iostream>
4 using namespace std;
5
6 int main(){
7     // Variables
8     int n1, n2;
9     double cociente1, cociente2;
10    // Entrada de datos
11    cout << "Numero 1: "; cin >> n1;
12    cout << "Numero 2: "; cin >> n2;
13    // Proceso
14    cociente1 = n1 / n2;
15    cociente2 = (n1 * 1.0) / n2;
16    // Salida
17    cout << "Cociente 1: " << cociente1 << endl;
18    cout << "Cociente 2: " << cociente2 << endl;
19    return 0;
20 }
```



```
Division_en_CPP.exe
Numero 1: 20
Numero 2: 6
Cociente 1: 3
Cociente 2: 3.33333
```




Ejemplo: Elaborar un programa que almacene el nombre, sexo, dos notas. Luego debe mostrar los datos ingresado y el promedio de las dos notas

```
1 // Ejemplo de Entrada y Salida
2 #include <iostream>
3 using namespace std;
4 int main(){
5     // Variables
6     string nombre;
7     double nota1, nota2, promedio;
8     char sexo;
9     // Entrada
10    cout << "\nINGRESO DE DATOS\n";
11    cout << "Nombre: ";    cin >> nombre;
12    cout << "Sexo (M/F): "; cin >> sexo;
13    cout << "Nota 1: ";    cin >> nota1;
14    cout << "Nota 2: ";    cin >> nota2;
15    // Proceso
16    promedio = (nota1 + nota2) / 2;
17    // Salida
18    cout << "\nREPORTE" << endl;
19    cout << "Nombre: " << nombre << endl;
20    cout << "Sexo: " << sexo << endl;
21    cout << "Nota 1: " << nota1 << endl;
22    cout << "Nota 2: " << nota2 << endl;
23    cout << "Promedio: " << promedio << endl;
24    return 0;
25 }
```

```
Entrada_Salida.exe

INGRESO DE DATOS
Nombre: Gustavo
Sexo (M/F): M
Nota 1: 18
Nota 2: 19

REPORTE
Nombre: Gustavo
Sexo: M
Nota 1: 18
Nota 2: 19
Promedio: 18.5
```



Ejemplo:

Calcular la siguiente expresión $z = \sqrt{x + y^3}$

```
1 // Evaluar Expresión
2
3 #include <iostream>
4 #include <math.h>
5 using namespace std;
6
7 int main(){
8     // Variables
9     int x, y;
10    double z;
11    // Entrada
12    cout << "\nENTRADA\n";
13    cout << "Valor de X: "; cin >> x;
14    cout << "Valor de Y: "; cin >> y;
15    // Proceso
16    z = sqrt( x + pow(y,3) );
17    // Salida
18    cout << "\nREPORTE\n";
19    cout << "Valor de Z: " << z << endl;
20    return 0;
21 }
```

```
EvaluarExpresion.exe
ENTRADA
Valor de X: 1
Valor de Y: 2
REPORTE
Valor de Z: 3
```



Estructuras de control



Estructuras de control

- Son aquellas estructuradas que nos permitirán evaluar instrucciones de forma secuencial, selectiva y repetitiva.
- Entre las estructuras básicas tenemos:
 - ✓ if - else
 - ✓ switch
 - ✓ while
 - ✓ do – while
 - ✓ for



Preguntas





Reflexionemos



ucontinental.edu.pe