PROGRAMACIÓN AVANZADA

ING. INDUSTRIAL

FAC. DE INGENIERÍA

UNIVERSIDAD NACIONAL DE JUJUY

Introducción C++

- Estructura de un programa en C++.Elementos de programas C++.Tipos de datos de C++. Expresiones. Entrada y Salida. Estructuras de control

Lenguaje de Programación C

- Fue diseñado e implementado por Brian Kernighan y Dennis Ritchie en 1972, a partir de los lenguajes BCPL (1967) y B (1970). Su desarrollo está estrechamente vinculado al del sistema operativo UNIX.
- Combina características de los lenguajes de alto nivel (sentencias de control y manipulación de datos) y de los lenguajes de bajo nivel (manejo de bits).
- Es independiente del hardware.

Lenguaje de Programación C++

- Es una extensión de C inventado por Bjarne Stroustrup a principio de la década de 1980 en los laboratorios Bell.
- Es un lenguaje de programación híbrido. Permite usar tanto la técnica de programación estructurada y como la técnica de programación orientada a objetos.

Enfoques de Programación

- Programación estructurada (evolucionó en los sesenta y setenta)
- Programación orientada a objetos (OOP: Object- Oriented Programming) (evolucionó en los ochenta)

Programación Estructurada

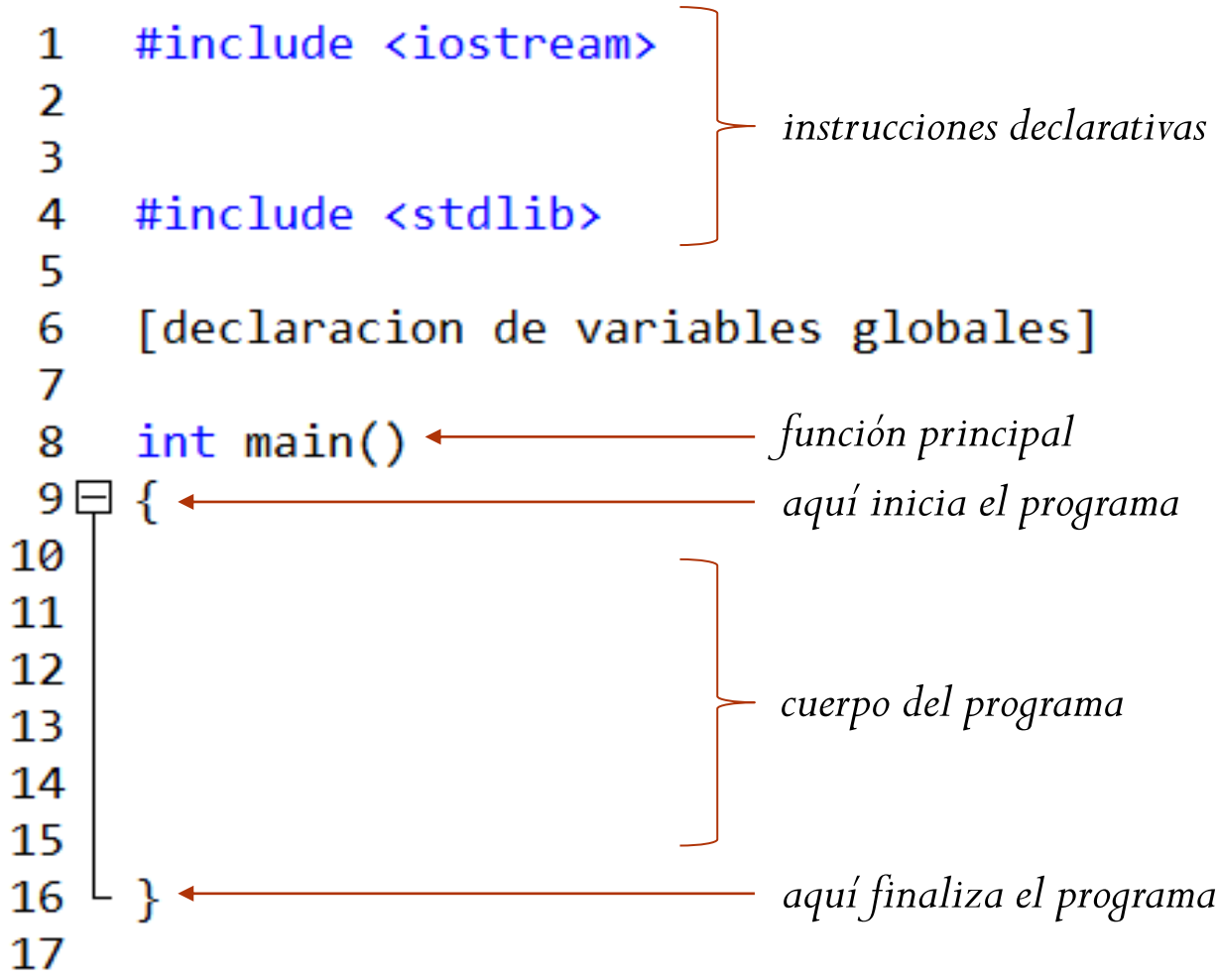
- Enfoque disciplinado que permite escribir programas estructurados, utilizando las siguientes tres estructuras de control bien definidas:
  - Secuencial (asignación, lectura, escritura)
  - Decisión o selección (simple, doble, múltiple)
  - Repetición (repita-mientras, hacer-mientras, repita-para)
- Los programas estructurados son fáciles de probar, depurar y modificar.
- Programación orientada a acciones donde la unidad básica es la función.

Programación Orientada a Objetos

- En el paradigma orientado a objetos, el programa se organiza como un conjunto finito de **objetos** que contienen datos y operaciones (funciones miembro en C++) que llaman a esos datos y que se comunican entre sí mediante **mensajes**.

Estructura básica de un programa en C/C++

Estructura de un programa en C++



Estructura de un programa en C++

- **#include<librería_solicitada>**
  - con esta directiva de preprocesador se incorporan las bibliotecas de funciones que se van a utilizar
  - Las librerías más utilizadas son:
    - <iostream> contiene las funciones para ingresar y mostrar datos
    - <math.h> contiene las funciones matemáticas comunes
    - <time.h> contienes las funciones para tratamiento y conversión entre formatos de fecha y hora
    - <stdio.h> contiene los prototipos de las funciones, macros y tipos para manipular datos de E/S
    - <stdlib.h> contiene tipos, macros y funciones para la conversión numérica, generación de memoria y tareas similares
    - <string.h> contiene los prototipos de las funciones y macros de clasificación de caracteres

Estructura de un programa en C++

- **Variable**
  - Una variable es un espacio reservado en la computadora para contener valores que pueden cambiar durante la ejecución de un programa. Toda variable tiene un tipo de dato, el cual determina los valores que se pueden representar y las operaciones que se pueden realizar con ellos.
  - Una **variable global** es válida para todas las funciones que componen el programa.

Estructura de un programa en C++

- main()
  - Es la función principal de un programa en C++.
  - Todo programa en C++ comienza con una función main() que es única.
  - Los paréntesis que le siguen contienen lo que se le va a mandar a la función
  - Entre las llaves { } se incluirá el bloque de instrucciones/sentencias que resuelve el problema. Las sentencias:
    - definen la lógica de un programa o subprograma
    - manipulan los datos para producir el resultado deseado por el usuario del programa

Estructura de un programa en C++

- Además de la función main(), un programa C++ consta de una colección de subprogramas (en C++ siempre son funciones).
- Todas las sentencias de C++ situadas en el cuerpo de la función main( ), o de cualquier otra función, deben terminar en punto y coma.

Estructura de un programa en C++

- COMENTARIOS
  - Un comentario es cualquier información añadida al archivo fuente e ignorada por el compilador. En C estándar los comentarios comienzan por `/*` y terminan con la secuencia `*/`. En C++ se define una línea de comentario comenzando con una doble barra inclinada (`//`). Todo lo que viene después de la doble barra inclinada es un comentario y el compilador lo ignora.

Estructura de un programa en C++

- Ejemplo de un programa básico en C++

```
#include <iostream>

using namespace std;

int main() {
    cout << "Hola mundo!" << endl;
  return 0;
}
```

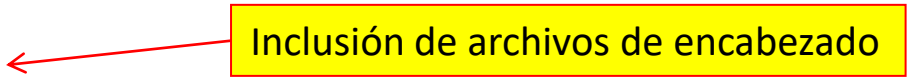
Estructura de un programa en C++

- Ejemplo de un programa básico en C++

```
#include <iostream>

using namespace std;

int main() {
    cout << "Hola mundo!" << endl;
  return 0;
}
```



Estructura de un programa en C++

- Ejemplo de un programa básico en C++

```
#include <iostream>

using namespace std;

int main() {
    cout << "Hola mundo!" << endl;
  return 0;
}
```

Estructura de un programa en C++

- Ejemplo de un programa básico en C++

```
#include <iostream>

using namespace std;

int main() {
    cout << "Hola mundo!" << endl;
  return 0;
}
```

Estructura de un programa en C++

- Ejemplo de un programa básico en C++

```
#include <iostream>

using namespace std;

int main() {
    cout << "Hola mundo!" << endl;
  return 0;
}
```

Código a ejecutar por la función principal

Elementos de un programa en C++

Unidad 1

Elementos de un programa en C++

- Los elementos básicos de un programa C++ son:
  - identificadores;
  - palabras reservadas;
  - comentarios;
  - signos de puntuación;
  - separadores, y
  - archivos cabecera.

Identificadores

- Un *identificador* es una secuencia de caracteres, letras, dígitos y subrayados (_) que comienza siempre por un carácter. Las letras mayúsculas y minúsculas son diferentes. Pueden tener cualquier longitud, pero el compilador ignora a partir del 32. No pueden ser palabras reservadas.
- Ejemplos de identificadores:

Nombre_Alumno	LetraIndice	Dia	Mayor	menor
Fecha_Compra	Alfa	habitacion24	i	j

Palabras reservadas

- C++, como la mayoría de los lenguajes, tiene reservados algunos identificadores con un *significado especial*, que sólo pueden ser usados con ese cometido.
- Una *palabra reservada*, tal como void (ausencia de tipo, o tipo genérico), es una característica del lenguaje C++.
- Una *palabra reservada* no se puede utilizar como nombre de identificador, objeto o función.

Palabras reservadas

```
asm
auto
bool
break
case
catch
char
class
const
continue
default
delete
do
double
else
enum
explicit
extern
float
for
friend
goto
if
inline
int
longmutable
namespace
new
operator
private
protected
public
register
return
short
signed
sizeof
static
struct
switch
template
this
throw
try
typedef
union
unsigned
virtual
void
volatile
wchar_t
while
```

Signos de Puntuación y Separadores

- Todas las sentencias de C++ deben terminar con un punto y coma (;). Los separadores son espacios en blanco, tabulaciones, retornos de carro y avances de línea. Otros signos de puntuación son:

!	-	]
%	+	\
^	=	;
&	{	‘
*	}	:
(	~	<
)	[	>
?	,	.
	/	

Archivos de Cabecera

- Un *archivo de cabecera* es un archivo especial que contiene las declaraciones de objetos y funciones de la biblioteca que son añadidos en el lugar donde se insertan.
- Un *archivo de cabecera* se inserta con la directiva #include. El nuevo **ANSI C++** ha cambiado el convenio (notación) original de los archivos de cabecera. Es posible utilizar sólo los nombres de las bibliotecas sin el sufijo **.h**; es decir, se puede usar iostream, cmath, cassert y cstlib en lugar de iostream.h, math.h, assert.h y stdlib.h respectivamente.

Tipos de Datos en C++

Unidad 1

Tipos de datos en C++

- Los tipos de datos simples o básicos de C++ son: ***enteros***; ***números de coma flotante*** (reales) y ***caracteres***

Tipo básico	Tipo	Ejemplo	Tamaño en bytes	Rango. Mínimo..Máximo
Carácter	char	'C'	1	0..255
Entero	short	-15	2	-128..127
	int	1024	2	-32768..32767
	unsigned int	42325	2	0..65535
	long	262144	4	-2147483648..2147483637
Real	float	10.5	4	$3.4*(10^{-38})..3.4*(10^{38})$
	double	0.00045	8	$2.7*(10^{-308})..2.7*(10^{308})$
	long double	1e-8	8	igual que double

El Tipo de Dato bool

- La mayoría de los compiladores de C++ incorporan el tipo de dato **bool** cuyos valores posibles son: "verdadero" (true) y "falso" (false). El tipo **bool** proporciona la capacidad de declarar variables lógicas, que pueden almacenar los valores verdadero y falso.
- Si en el compilador de C++ no está disponible el tipo **bool**, deberá utilizar el tipo de dato int para representar el tipo de dato **bool**.
- C++ utiliza el valor entero 0 para representar falso y cualquier valor entero distinto de cero (normalmente 1) para representar verdadero. De esta forma, se pueden utilizar enteros para escribir expresiones lógicas de igual forma que se utiliza el tipo **bool**.

Variable

- En C++ una variable es una posición de memoria a que se le asocia un nombre (identificador) en el que se almacena un valor del tipo de dato del que se ha definido. El valor de una variable puede cambiar a lo largo de la ejecución del programa, siendo manipulada por los operadores aplicables al tipo del que ha sido definida la variable.
- **Declaración**. Una declaración de una variable es una sentencia que proporciona información de la variable al compilador C++.c
- Es preciso declarar las variables antes de utilizarlas. La sintaxis de declaración es:

  tipo lista de variables;

- Siendo *tipo* el nombre de un tipo de dato conocido por C++ y *lista de variables* una sucesión de nombres separadas por comas, y cada nombre de la lista un identificador de C++.
- **Inicialización**. Las variables, pueden ser inicializadas al tiempo que se declaran. El formato general de una declaración de inicialización es:

  tipo lista de inicialización;

- Siendo *lista de inicialización* una sucesión nombre_variable = expresión. Además, expresión es cualquier expresión válida cuyo valor es del mismo tipo que tipo.
- *Nota: los dos formatos de declaración pueden combinarse entre sí.*

Variables

Duración de una variable

Variables Locales

Son aquellas que se definen dentro de una función

Variables Globales

son variables que se declaran fuera de las funciones y por defecto son visibles a cualquier función incluyendo main( ).

Entrada y Salida

Unidad 1

Entrada de datos desde consola

- El flujo de entrada **`cin`** permite introducir datos desde el teclado y almacenarlos en variables.

- Ejemplo:

```
int main() {
  int x;
  cout << "Escribe un numero entero: ";
  cin >> x;
  cout << "El numero que escribiste es: ";
  cout << x << endl;
}
```

Salida de datos a consola

- En C++ se puede usar el flujo **`cout`** para imprimir información en la pantalla, utilizando el operador de salida de flujo <<

- La constante `endl` imprime un retorno de carro

- Ejemplos:

```
cout << "Lenguaje C++";
cout << "El valor de x es " << x << endl;
cout << "El area del circulo es " << 3.14159 * r
  * r;
cout << endl << endl << endl;
```

Expresiones

Unidad 1

Variables y asignación

- Para declarar variables en C/C++ se escribe primero el tipo de variable, seguido del nombre de una o más variables (separados por comas).

- Ejemplos:

```
int a;
int i, j, k;
float x, y;
```

Asignación de variables

- Para asignar un valor a una variable se utiliza el operador =
- Es posible asignar valores al momento de declarar las variables;
- Ejemplos:

```
x = 10;
float a = 3.4, b = 5.6, c = -1.7;
```

Operadores aritméticos

- En C/C++ se pueden formar expresiones aritméticas con los operadores comunes:
  + (suma), - (resta), * (multiplicación), / (división), **% (resto)**

- Debe tenerse siempre en cuenta que los operadores actúan de acuerdo al tipo de datos de los operandos involucrados. Ejemplo:

```
x = 5 / 2;          // el resultado es 2
x = 5.0 / 2; // el resultado es 2.5
```

- C/C++ no cuenta con un operador de potencia, pero se puede utilizar la función `pow()` de la librería math.h

Operadores Incrementales

- C también tiene el *operador incremental* unario, ++, y el *operador decremental* unario -- que mostramos a continuación.

Operadores	Expresión de muestra	Explicación
++	++a	Se incrementa a en 1 y a continuación se utiliza el nuevo valor de a en la expresión en la cual resida a
++	a++	Utiliza el valor actual de a en la expresión en la cual reside a, y después se incrementa a en 1
--	--b	Se decrementa en 1 y a continuación se utiliza el nuevo valor de b en la expresión en la cual reside b
--	b--	Se utiliza el valor actual de b en la expresión en la cual reside b, y después se decrementa a b en 1

Expresiones booleanas

- Una expresión booleana es aquella que después de evaluarse puede tomar uno de dos posibles resultados: **verdadero** o **falso.**
- En lenguaje C/C++, los valores verdadero y falso se representan, respectivamente, con los números 1 y 0.
- Sin embargo, cualquier valor distinto de cero es considerado como “verdadero”.

Operadores de comparación

- Una manera de formar expresiones booleanas es utilizando los operadores de comparación:

a == b	- igualdad
a < b	- menor a
a > b	- mayor a
a != b	- distinto de
a <= b	- menor o igual a
a >= b	- mayor o igual a

Operadores booleanos

- Las expresiones booleanas pueden combinarse mediante los operadores booleanos clásicos:

&&	- conjunción (and)
||	- disyunción (or)
!	- negación (not)

- Notar que estos operadores difieren de los operadores lógicos binarios: &, |, ~

- Notar que, en muchos casos, los operadores && y || pueden reemplazarse, respectivamente, por el producto y la suma.

Práctica #1

- Escriba un programa que pida al usuario la base y la altura de un rectángulo y calcule y muestre su superficie.