

# Clase #02 de 27 El Compilador & el Lenguaje de Programación C

*Abril 6, Lunes*

# Agenda para esta clase

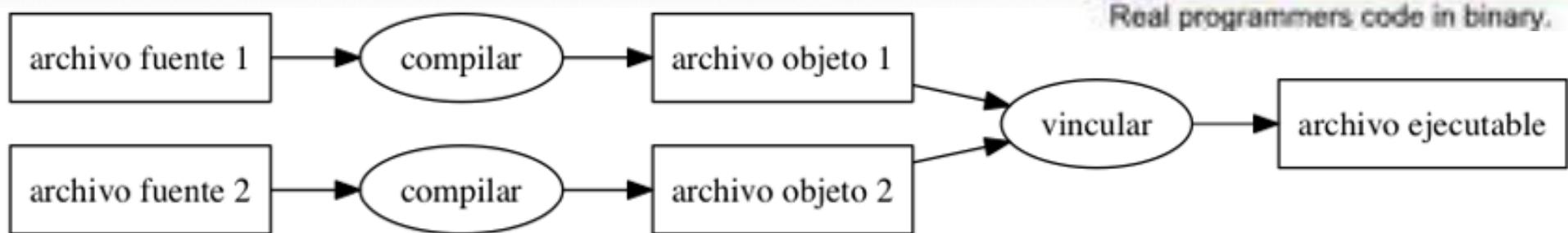
- Primer contacto con el compilador
- Intervalo
- Trabajo #0
- Introducción al Lenguaje de Programación C

# **Primer Contacto con el Compilador**

Lenguajes y Herramientas de Desarrollo

# ¿Qué es un Compilador?

- Programa que hace programas, un meta programa
- Traductor
- Baja de Nivel de Abstracción
- Función de Lenguaje a Lenguaje:  $C: L_1 \rightarrow L_2$
- Familia de Compiladores
- Par ( $L_1, L_2$ )
- Proceso, en etapas: Front End y Back End
- Compilaciones separadas, luego vinculadas

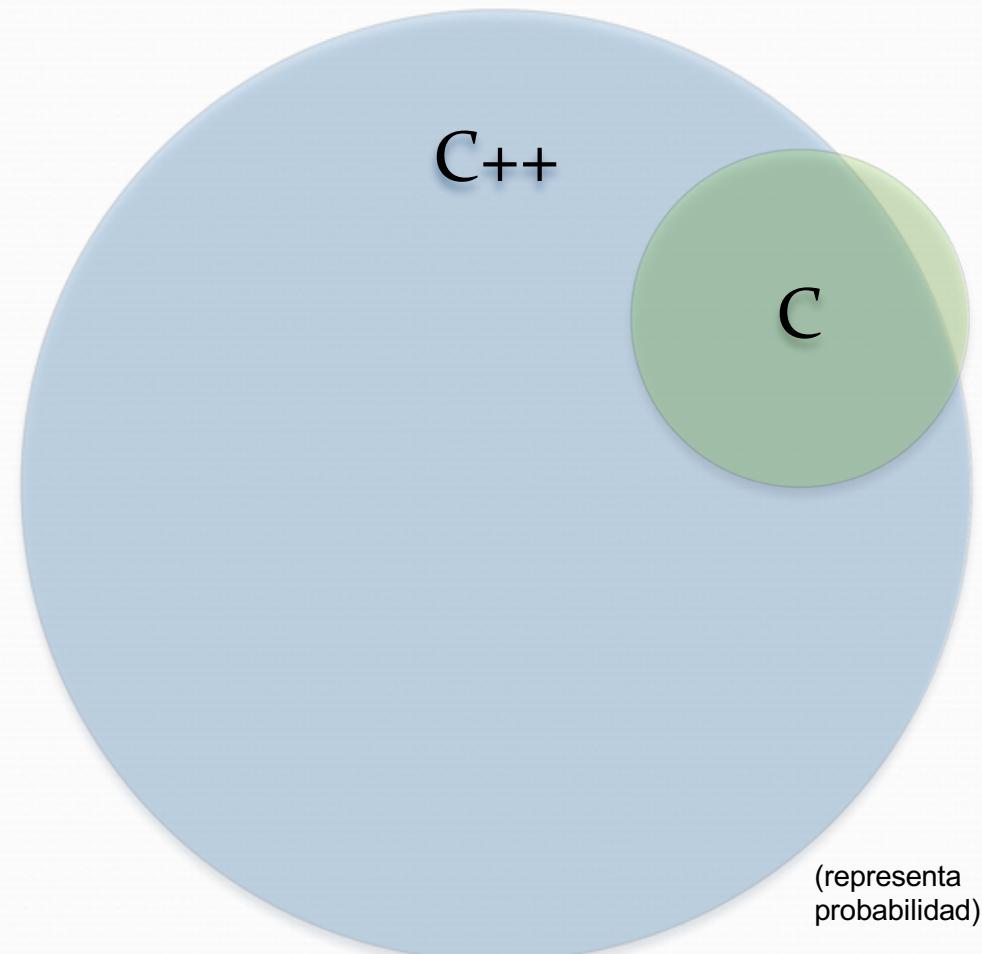


# Sobre los Lenguajes C y C++

## Historia

- 1970's
  - C
  - C With Classes
- 1980's
  - Comienza standard C
  - C++
- 1990's
  - Standard C90
  - Standard C++98
- 2000's
  - Standard C99
  - Standard C++03
- 2010's
  - Standard C11, C18
  - Standard C++11, 14, 17, 20

**Conjuntos de infinitos  
programas válidos de C++ y C**



# "Hello, World!"

```
/* Hello world
JMS
20150402
*/
#include <stdio.h>

int main(void){
    printf("Hello, world!\n");
}
```

- Propósito
- Comentario encabezado
  - Qué
    - Título descriptivo
  - Quién
    - Número de Equipo e integrantes
  - Cuándo
    - Se actualizó por última vez

- 1.1 [K&R1988]
- [https://en.wikipedia.org/wiki/"Hello,\\_World!"\\_program](https://en.wikipedia.org/wiki/)

# Proceso básico para desarrollar programas

1. **Escribir** el programa con un editor de texto (e.g., vi, Notepad, TextPad, Sublime, TextMate, Notepad++, Notepad2). Es convención para los archivos fuente de C la extensión sea .c (e.g., hello.c)
2. **Compilar** el archivo fuente para producir el programa objeto (e.g., cc hello.c) ...  
... y **Vincular** (link) el programa con las bibliotecas para crear el programa ejecutable; generalmente ocurre junto con el punto anterior.
3. **Ejecutar** el programa (e.g., hello.exe ó ./a.out)
4. ¿Error en 2 ó 3? Volver a 1 y repetir.

# Ejemplo desde línea de comando

## Mac OS X C18

1. Desde la línea de comando
  1. > vi hello.c      crear el fuente
  2. > cc hello.c -std=c18 -Weverything  
                      crear el ejecutable  
                      en realidad: Preprocesador → Compilador → Linker
  3. > ./a.out      ejecutar  
                      Hello, World!      salida
2. Si hay un error en el paso 2 ó 3, volver al 1 y repetir 2 y 3

Otra versión para gcc es:

> cc hello.c -std=c18 -Wall -pedantic-errors

```
josemariasola:CHelloWorld> cc hello.c -std=c11 -Weverything
josemariasola:CHelloWorld> ./a.out
Hello, World!
josemariasola:CHelloWorld> █
```

# Ejemplo desde línea de comando Compilador Microsoft (ejemplo en C++, no C)

1. Desde la línea de comando
  1. > **notepad hello.c** crear el fuente
  2. > **cl hello.c** crear el ejecutable  
en realidad: Preprocesador → Compilador → Linker
  3. > **hello.exe** ejecutar  
Hello, World! salida
2. Si hay un error en el paso 2 ó 3, volver al 1 y repetir 2 y 3

The screenshot shows a 'Developer Command Prompt for VS2013' window. The command line output is as follows:

```
C:\Samples>cl Hello.cpp
Microsoft (R) C/C++ Optimizing Compiler Version 18.00.21005.1 for x86
Copyright (C) Microsoft Corporation. All rights reserved.

Hello.cpp
C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\INCLUDE\xlocale(337) : warning C4530: C++ exception handler used, but unwind semantics are not enabled. Specify /EHsc
Microsoft (R) Incremental Linker Version 12.00.21005.1
Copyright (C) Microsoft Corporation. All rights reserved.

/out:Hello.exe
Hello.obj

C:\Samples>Hello
Hello World!

C:\Samples>
```

# Ejemplo desde línea de comando

## Compilador Borland

1. Desde la línea de comando
  1. > **notepad hello.c** crear el fuente
  2. > **bcc32 hello.c** crear el ejecutable  
en realidad: Preprocesador → Compilador → Linker
  3. > **hello.exe** ejecutar  
Hello, World! salida
2. Si hay un error en el paso 2 ó 3, volver al 1 y repetir 2 y 3

The screenshot shows a Windows Command Prompt window titled 'C:\WINDOWS\System32\cmd.exe'. The command line shows the steps of the build process:

```
C:\Program Files\Borland\BCC55\Bin>bcc32 hello.c
Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland
hello.c:
Turbo Incremental Link 5.00 Copyright (c) 1997, 2000 Borland

C:\Program Files\Borland\BCC55\Bin>hello
Hello, World!

C:\Program Files\Borland\BCC55\Bin>
```

# Herramientas de Desarrollo: Sobre el Compilador y el IDE

- Con IDE (*Integrated Development Environment*, Entorno Integrado de Desarrollo)
  - Ejemplos
    - Apple Xcode
    - Microsoft Visual Studio
    - Eclipse
- Sin IDE
  - Editor
  - Compilador.

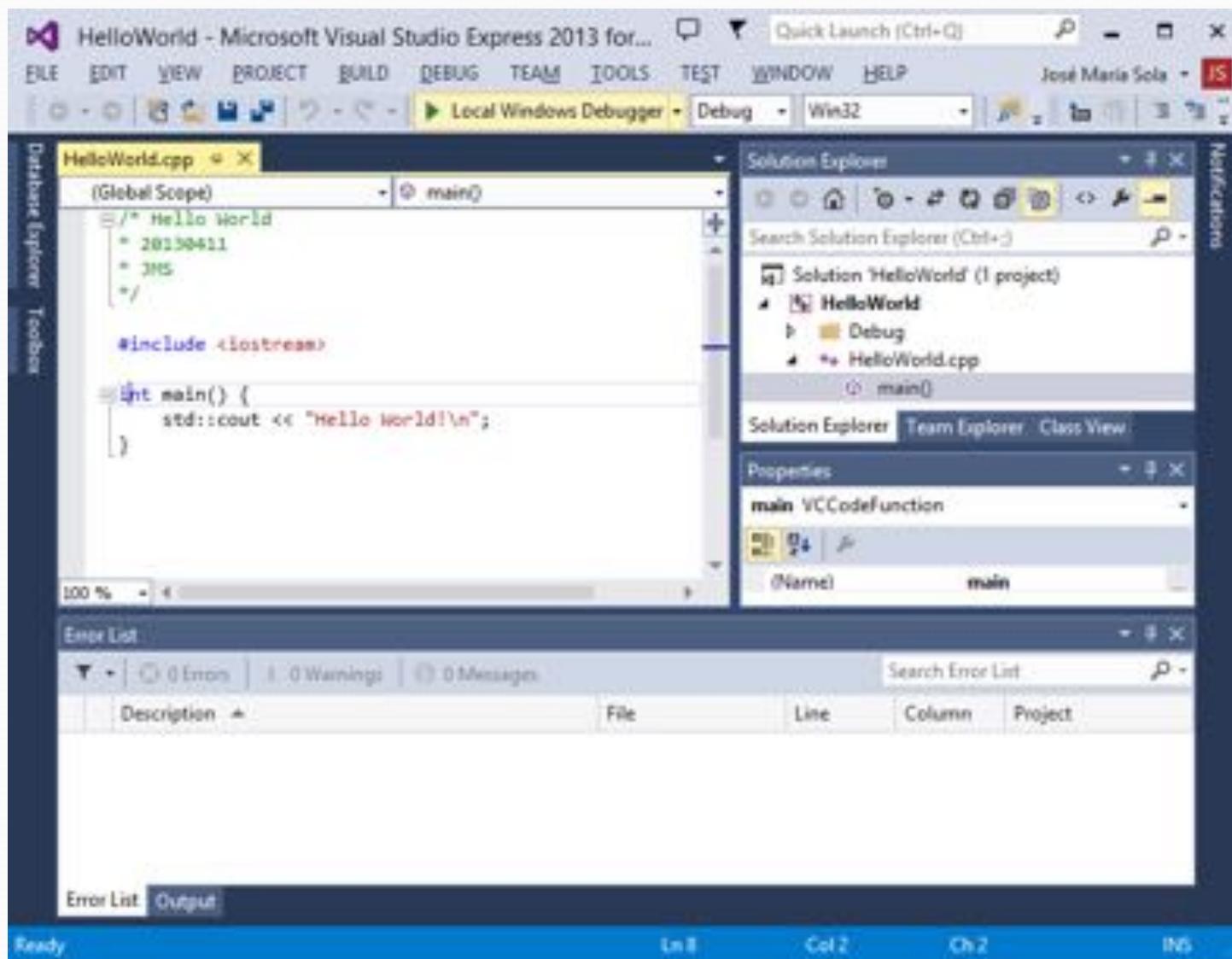
## Con IDE

- Editor
- Depurador
- Gestor de Proyectos y de configuraciones
- Ayuda
- y más...

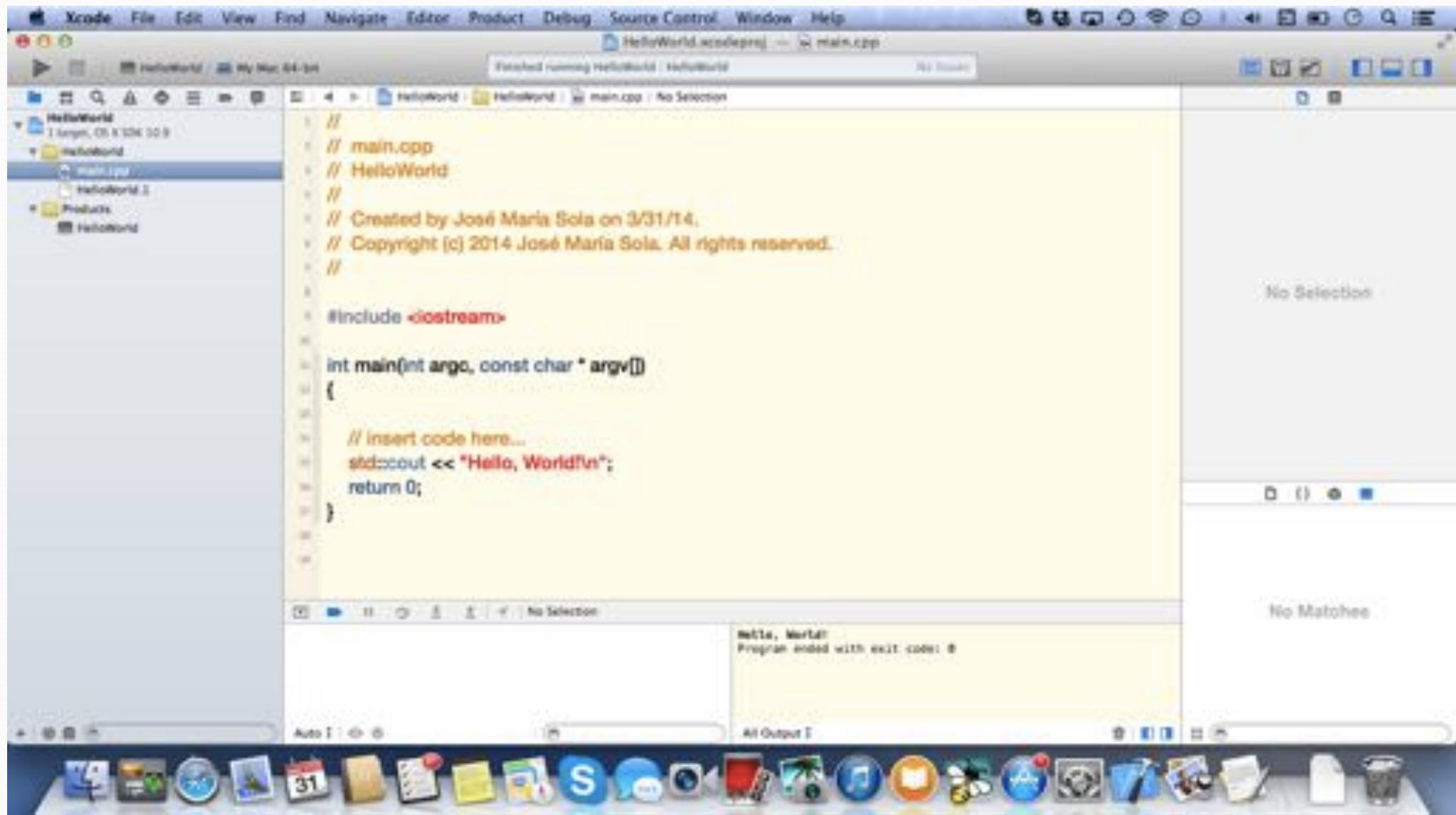
**Sin IDE**  
Requiere editor

Compilador de C/C++

# Ejemplo con IDE Microsoft Visual Studio Express for Windows Desktop (Ejemplo en C++, no C)



# Ejemplo con IDE Apple Xcode (ejemplo en C++, no C)



# Ejemplo con IDE Microsoft Visual Studio Code

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- Title Bar:** hello.c — Untitled (Workspace)
- File Explorer:** Shows a file icon.
- Toolbar:** Includes icons for DEBUG (play), settings, and other workspace operations.
- Left Sidebar:** Contains sections for VARIABLES, WATCH, CALL STACK, and BREAKPOINTS. The WATCH section has a red dot at line 10, indicating a breakpoint or a watched variable.
- Code Editor:** Displays the following C code:

```
/* Hello.cpp
C11
JMS
2015
*/
#include <stdio.h>
int main(void){
    printf("Hello, World!\n");
}
```
- Bottom Status Bar:** Shows the file path master\*, line Ln 9, column Col 6, spaces Spaces: 2, encoding UTF-8, line endings LF, character set C, Mac, and notification count 1.

# Make (C)

- Nuestro objetivo (*goal*) es construir ó hacer (*make*) la versión ejecutable de `hello.c`
- Desde la línea de comando podemos lograrlo con el comando `make`, pasándole como argumento el nombre del *goal*, en nuestro caso `hello`
- El comando `make` sabe *makear* un ejecutable a partir de un fuente
- En sistemas *Windows* el análogo es el comando `nmake.exe`, aunque también es posible utilizar el `make.exe` ó `mingw32-make.exe` si instalamos *MinGW*.



```
$ make hello
cc -std=c17 -Weverything -pedantic-errors    hello.c   -o hello
$ ./hello
Hello, World!
$
```

# Links a Compiladores C/C++

Usar unos de estos compiladores o cualquier otro, siempre y cuando se lo configure para **C18**

- Con IDE y Línea de Comandos
  - Microsoft Code
    - <https://code.visualstudio.com/Download>
  - Apple Xcode
    - <https://developer.apple.com/xcode>
  - Replit: IDE On-Line
    - <https://repl.it/>
  - Microsoft Visual Studio Community 2017
    - <https://www.visualstudio.com/vs/features/cplus/>
  - CodeLite
    - <https://codelite.org>
  - Eclipse IDE for C/C++ Developers
    - <https://www.eclipse.org/downloads/packages/release/2020-03/r/eclipse-ide-cc-developers-includes-incubating-components>
  - Más antiguos
    - Code::Blocks
      - <http://www.codeblocks.org/downloads/>
    - Dev-C++
      - <http://www.bloodshed.net/devcpp.html>
- Sin IDE, solo Línea de Comandos
  - Si tu sistema es un UNIX (macOS, GNU, Linux) es probable que incluya un compilador, probá los comandos **cc** y **gcc** desde la línea de comandos
  - GNU C Compiler (ahora GNU Compiler Collection)
    - <http://gcc.gnu.org/install/binaries.html>
    - Para plataformas Windows
      - <http://www.mingw.org>
      - <http://mingw-w64.org/doku.php>
  - Clang
    - <http://releases.llvm.org/download.html>
  - Embarcadero Free C++ Compiler
    - <https://www.embarcadero.com/free-tools/ccompiler>
  - Más antiguos
    - Borland C++ Compiler version 5.5 Free Download
      - <http://edn.embarcadero.com/article/20633>
        - Using the Borland 5.5 Compiler and command-line tools
      - <http://edn.embarcadero.com/article/20997>
        - Borland C++ 5.5 Free Command-line Tools Supplementary Information
      - <http://edn.embarcadero.com/article/21205>

# Compiladores, Editores y Entornos de Desarrollo: Instalación, Configuración y Prueba

- Introducción a compilador, entornos de desarrollo
- Amar de entorno de desarrollo para C/C++ bajo un entorno Windows, basado en el compilador MinGW y el editor de código fuente Visual Studio Code.
- Paper disponible en:

<https://josemariasola.wordpress.com/ssl/papers/>

# Intervalo

## 20 minutos

# Trabajo #0

hello.c: "Hello, World!"

# Trabajo #0 – "Hello, World!" en C

- Enunciado en [josemariasola.wordpress.com](http://josemariasola.wordpress.com)
- Secuencias de Tareas
  - Si no posee una cuenta GitHub, crearla
  - Crear un repositorio público llamado SSL
  - Escribir el archivo readme.md que actúa como front page del repositorio personal
  - Crear la carpeta oo-CHelloWorld.
  - Escribir el archivo readme.md que actúa como front page de la resolución.
  - Seleccionar, instalar, y configurar un compilador C18
  - Indicar en readme.md el compilador seleccionado.
  - Probar compilador con hello.c que envíe a stdout la línea Hello, World! o similar
  - Ejecutar el programa, y capturar su salida en un archivo de texto output.txt
  - Publicar en repositorio personal SSL \ oo-CHelloWorld: readme.md, hello.c, y output.txt
  - Informar por email a [UTNFRBASSL@yahoo-groups.com](mailto:UTNFRBASSL@yahoo-groups.com) el usuario usuario GitHub.
- Restricciones
  - La fecha y hora límite de entrega se publica en el calendario
  - La evaluación se hace con lo publicado en GitHub.

# Introducción al Lenguaje de Programación C

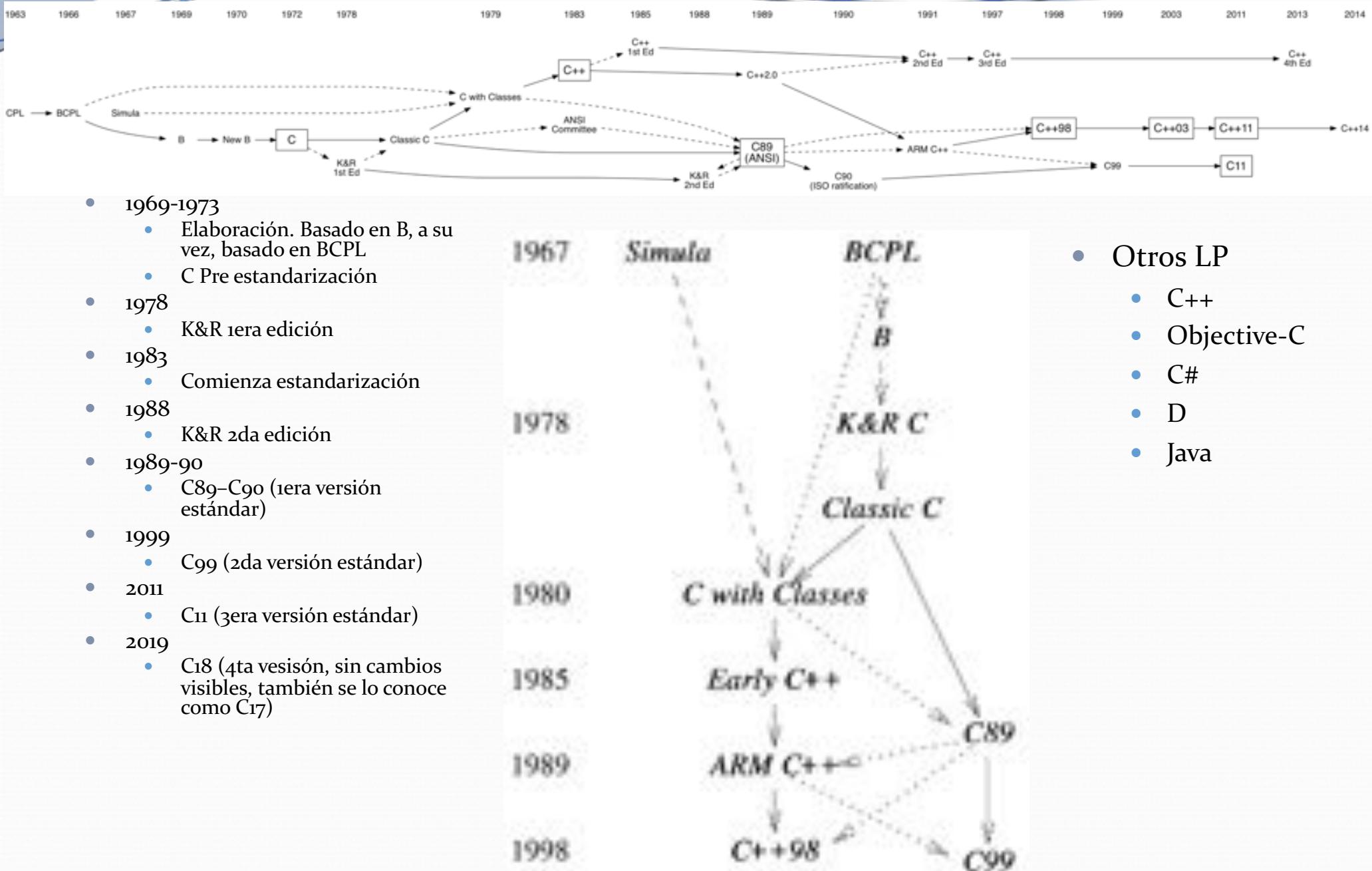
# Descripción general

- LP de propósito general, no está especializado
- Economía en las expresiones, pero expresivo
  - Poco texto, mucha información
  - Variabilidad en texto, variabilidad en significado.
- Control de flujo (*¿de qué?*)
- Estructuras de datos
- Gran cantidad de operadores
- flexible
- No es de muy alto nivel (*¿de qué?*)
- No es grande (*¿en qué sentido?*)
- Su falta de restricciones y su generalidad lo hacen efectivo
- Independiente de máquina, portable (procesador y sistema operativo)
- El lenguaje de programación de Unix
- Primer lenguaje de alto nivel eficiente y portable
  - En el momento, menos problemas que
    - Basic, PL/I, Fotran, Cobol, Pascal
  - Comparado con Lisp
    - Vinculación
    - Más rápido
    - Con GC, lo cual no es apropiado para programación de sistemas
  - C es la mejor abstracción de una computadora existente, no de un dispositivo imaginario
  - Suficientes estructuras de control y de datos para resolver problemas, limitadas para que se pueda implementar el compilador.

# Frases sobre C

- C no es un LP grande, y no le queda bien un libro grande (K&R)
- C tiene vueltas, falencias y un enorme éxito (Ritchie)
- C es un arma filosa, con la se pueden hacer programas eficientes y elegantes o una “carnicería” (Pike)
- C mejora a medida que uno gana experiencia con C (K&R) (Curva de aprendizaje empinada).

# Historia de C y LP relacionados



- 1969-1973
  - Elaboración. Basado en B, a su vez, basado en BCPL
  - C Pre estandarización
- 1978
  - K&R 1era edición
- 1983
  - Comienza estandarización
- 1988
  - K&R 2da edición
- 1989-90
  - C89-C90 (1era versión estándar)
- 1999
  - C99 (2da versión estándar)
- 2011
  - C11 (3era versión estándar)
- 2019
  - C18 (4ta versión, sin cambios visibles, también se lo conoce como C17)

- Otros LP
  - C++
  - Objective-C
  - C#
  - D
  - Java

# Términos de la clase #02

Definir cada término con la bibliografía

- Compilador
  - Función del compilador
  - Proceso de compilación
  - Proceso básico para desarrollar programas
  - Lenguaje máquina (bajo nivel de abstracción)
  - Lenguaje de Alto Nivel de Abstracción
  - C
  - C++
  - Hello World (Kernighan)
  - IDE (Integrated Development Environment, Entorno Intedrado de Desarrollo)
  - Utilidad Make
- Introducción al Lenguaje de Programación C
  - Nivel de abstracción
  - Independiente de Máquina
  - Control de flujo de ejecución
  - Historia de C y de ANSI C
  - Relación entre C y C++
  - ANSI C, C89 ó C90
  - C99
  - C11
  - C18, también conocido como C17

# Tareas para la próxima clase

1. Leer de [K&R1988] *desde la tapa hasta la sección 1.1 Comenzado inclusive*
2. Entrega Trabajo #0



# ¿Consultas?



# Fin de la clase