

# Sintaxis y Semántica de los Lenguajes SSL

*K2051*

*2022*

*Esp. Ing. José María Sola, Profesor*

Universidad Tecnológica Nacional  
Facultad Regional Buenos Aires  
Departamento de Ingeniería en Sistemas de Información

# Clase #01 de 27

## Presentación Inicial

*Marzo 28, Lunes*

# Agenda para esta clase

- Bienvenidos a Segundo año y al Curso
- Presentación inicial, conozcámonos
- El contexto de SSL
- La aprobación y la regularización
- Intervalo
- Primer contacto con el compilador
- Trabajo #0

# Presentación Inicial

Conozcámonos

# Docentes del Curso

- Profesor a Cargo
  - Esp. Ing. José María Sola
- Auxiliares
  - Ariel Silva
  - Elizabeth Sosa

# Modalidad Híbrida

- Clases Sincrónicas
  - Presencial Digital en Meet
  - Presencial Física en Campus
    - Estas clases se van a notificar con suficiente antelación
- Clases Asincrónicas
  - Excepcionalmente habrá clases donde se indica lectura y trabajo a realizar



# Clase Presencial Digital

- Micrófono “muteado”
- Horarios del curso
  - Lunes 7:45pm
  - Acceso por *Hangouts Meet* desde *Calendar*
- Consultas antes y después de clase

# Clase Presencial Física en Campus

- Primera fila de banco **libres para el proyector y para otro material** (si el aula no tiene proyector integrado)
- Entrar y salir por puerta trasera (si el aula la tiene)
- Celular, en **silencio o vibrador**
- Horarios del curso
  - 4 horas cátedra con intervalo de 15 minutos
    - 3 horas reloj + intervalo de 15 o 20 minutos
  - De **19:45** hasta antes de las **22:00**, con intervalo
    - Horas dependen de presencia de transporte público y de la vuelta a la prespecialidad física.
- Consultas presenciales, antes y después de clase.



# El Contexto de SSL

# Análisis del Título de la Carrera – Ingeniería en Sistemas de Información

- Ingeniería
  - Aplicación tecnología—ciencia y técnicas—para resolución problemas
  - Construcción de soluciones
- Sistema
  - Conjunto de elementos relacionados con objetivo común
- Información
  - Datos procesados
  - Materia prima para toma de decisiones
  - Dato valor sintáctico
  - Información valor semántico.
- Sistema de Información
  - Sistema manual o automático con
    - personas, máquinas o métodos que
    - procesa información
      - recolecta, transmite, almacena distribuye, presenta y manipula
    - información para sus usuarios
    - en tiempo y forma
  - En general, los sistemas de información tienen grandes partes implementadas con Sistemas Software
  - La programación y la construcción de Sistemas Software.

# Rol de la Materia en las Actividades del Ingeniero en Sistemas de Información



# Integración Vertical: Asignaturas Anteriores

- Algoritmos y Estructura de Datos
- Matemática Discreta
- Sistemas y Organizaciones (no correlativa)

# Matemática Discreta

- Objetivos
  - Aplicar métodos inductivos, deductivos y recursivos en la resolución de situaciones problemáticas y demostraciones matemáticas
  - Comprender los conceptos y procedimientos necesarios para resolver relaciones de recurrencia
  - Aplicar propiedades y funciones definidas en los números enteros y enteros no negativos
  - Caracterizar distintas estructuras algebraicas, enfatizando las que sean finitas y las álgebras de Boole
  - Aplicar propiedades de grafos, dígrafos y árboles en la resolución de situaciones problemáticas
- Contenidos Mínimos
  - Lógica Proporcional Clásica y de Predicados de Primer Orden
  - Teoría de Números
  - Inducción Matemática
  - Relaciones de Recurrencia
  - Estructuras Algebraicas Finitas y Algebra de Boole
  - Grafos, dígrafos y árboles
- Cintenidos Extendidos
  - Lenguajes Formales
  - Autómatas Finitos
  - Expresiones Regulares
  - Gramáticas.



# Algoritmos y Estructuras de Datos

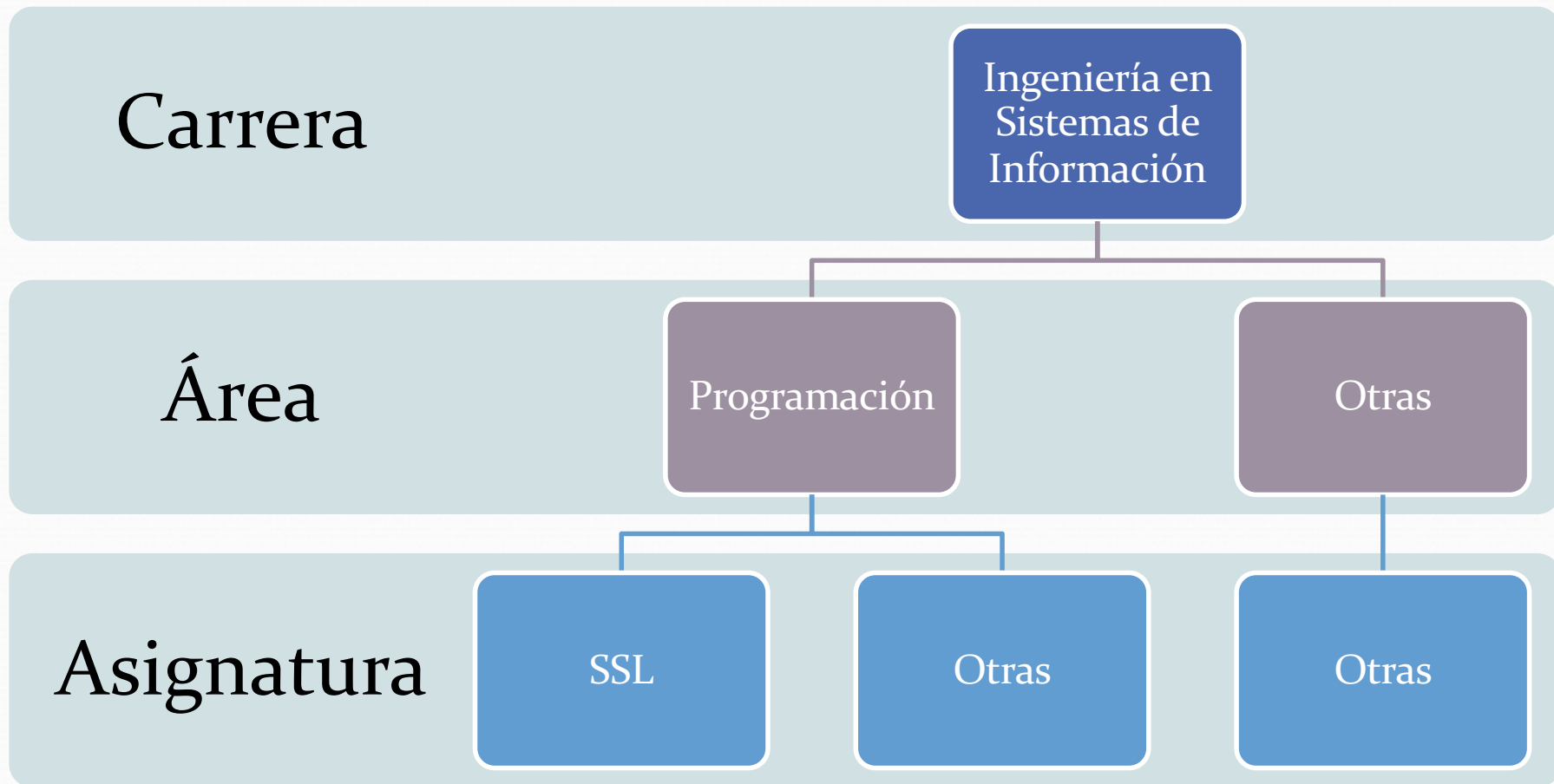
- Objetivos
  - Identificar problemas algorítmicos.
  - Conocer el proceso de diseño e implementación de software
  - Aplicar las herramientas fundamentales representativas de los procesos, integrando la sintaxis elemental de un lenguaje de programación en el laboratorio asociado
- Contenidos Mínimos
  - Concepto de Dato
  - Tipos de Datos Simples
  - Tipo Abstracto de datos
  - Estructuras de Control Básicas: secuencial, condicional, cíclica
  - Estrategias de Resolución
- Estructuras de Datos: registros, arreglos y archivos
- Abstracciones con procedimientos y funciones
- Pasaje de Parámetros
- Estructuras de Datos lineales (Pilas-Colas)
- Algoritmos de Búsqueda, Recorrido y Ordenamiento
- Archivos de Acceso Secuencial y Aleatorio: organizaciones y accesos.
- Procesamiento Básico
- Recursividad
- Nociones de Complejidad Computacional
- Noción de Orden de Complejidad.



# Repaso de Conceptos de Asignaturas Anteriores

- Área de Sistemas de Información
  - Sistemas y Organizaciones (no correlativa)
    - Sistema
    - Organización
    - Información
    - Dato
    - Proceso
- Área de programación
  - Matemática Discreta (correlativa)
    - Función
    - Autómata
    - Lógica
    - Números aleatorios
    - Grafos
  - Algoritmos y Estructura de Datos (correlativa)
    - Algoritmo
    - Dato
    - Estructura de Datos
    - Función
    - Programa
    - Lenguaje
    - Proceso
    - Procedimiento
    - Parámetro
    - Argumento.

# SSL en la Carrera



# Área Programación

- Objetivos
  - Formar e informar acerca de metodologías, técnicas y lenguajes de programación, como herramientas básicas para el desarrollo de software y el estudio de disciplinas que permitan crear nuevas tecnologías
- Asignaturas (640 hs)
  - 1. Matemática Discreta (96 hs)
  - 2. Algoritmos y Estructuras de Datos (160 hs)
  - 3. Sintaxis y Semántica de los Lenguajes (128 hs)
  - 4. Paradigmas de Programación (128 hs)
  - 5. Gestión de Datos (128 hs).

# Objetivos de cada Asignatura del Área

- MD, AyEdD, SSL
- Paradigmas de Programación
  - Comprender los fundamentos de los paradigmas de programación básicos que son utilizados por los lenguajes de programación actuales
  - Conocer el modelo formal o semiformal subyacente de cada paradigma y la forma en que el mismo es incorporado en un lenguaje de programación concreto.
  - Aplicar los diferentes paradigmas en la resolución de problemas
- Gestión de Datos
  - Desarrollar los conceptos de estructuración de los datos en dispositivos de almacenamiento.
  - Describir metodologías para el modelado de datos.
  - Conocer modelos actuales para la persistencia de grandes volúmenes de datos.
  - Desarrollar los conceptos relacionados con la consistencia, integridad y seguridad de la información. Aplicar técnicas y métodos para el tratamiento concurrente de los datos.

# Integración con otras Asignaturas y Conceptos Principales

- Integración vertical: Anteriores
  - Área de Programación
    - Matemática Discreta
    - Algoritmos y Estructuras de Datos
- Integración Horizontal: Paralelas
  - Área de Programación
    - Paradigmas de Programación
  - Computación
    - Sistemas Operativos
  - Sistemas de Información
    - Sistemas y Organización
    - Análisis de Sistemas
    - Diseño de Sistemas
- Integración vertical: Posteriores
  - Área de Programación
    - Gestión de Datos
    - Ingeniería en Software
- Abstracción -- Concepto fundamental
  - Separación, dejar de lado los detalles para enfocar en lo importante
- Tipo de Dato
  - Conjunto de Valores y conjunto de operaciones sobre ese conjunto de valores
- Orientación a Objetos
  - Objeto: entidad con comportamiento y que mantiene un estado.



# La Aprobación y la Regularización

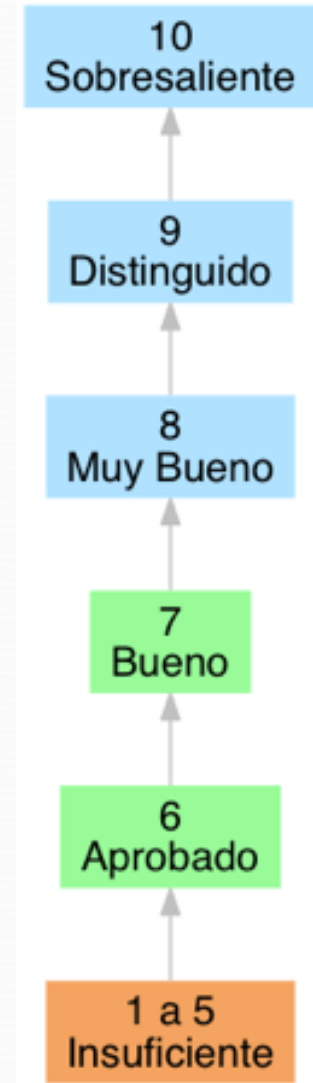


# Dinámica de Trabajo

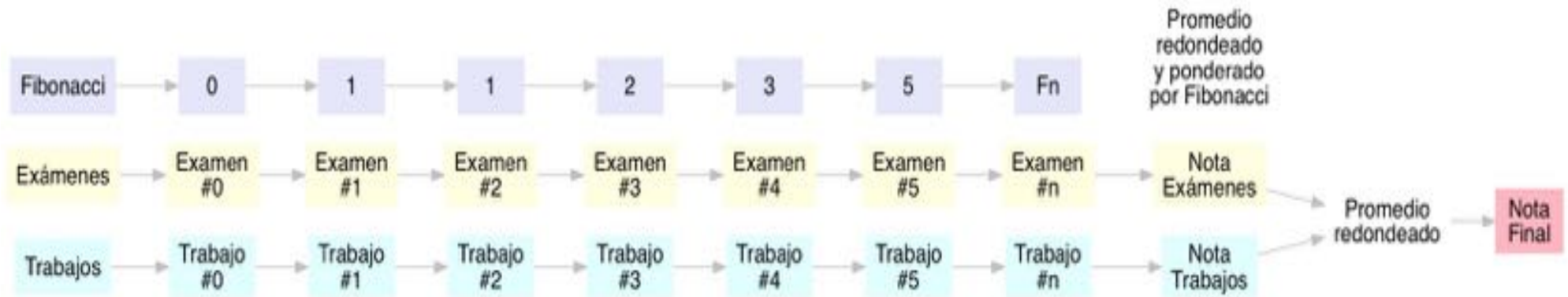
- Indicación de lecturas y ejercicios obligatorios para siguiente clase
- Profundización de conceptos en clase
- Clase interactiva y participativa
  - Se recomienda seguir la clase con la bibliografía y tomar apuntes a la par
- Ejercicios de aplicación en clase
- Consultas
  - Durante clase, y justo antes de iniciar y finalizar la clase
  - En otros horarios, a través del Foro.

# Aprobación – Dos Formas

- Requisito para aprobación
  - Regularización y aprobación del examen final ó
  - Aprobación Directa
- Examen final
  - Luego de un ciclo lectivo
    - Correlativas aprobadas
      - Algoritmos y Estructuras de Datos
      - Matemática Discreta
  - Cuatro oportunidades



# Requisitos para la Regularización (i.e., Firma)



- Bedelía: **75% de Asistencia**
- Cátedra: Evaluaciones
  - Frecuentes, Grupales ó individuales, de aplicación y conceptuales
  - **Dos conjuntos de Evaluaciones**
    - **#1 Trabajos**
    - **#2 Exámenes**
    - Fechas establecen durante el curso
    - **Cada conjunto tiene su nota**
    - **Dos recuperatorios por evaluación en Dic y Feb**
  - Al finalizar el curso **evaluación individual oral ("coloquio")** basada en trabajos
  - **Nota Final:** promedio entre conjunto **#1 Exámenes** y conjunto **#2 Trabajos**
  - **Para regularizar, ambas notas mayores o iguales a 6(seis).**

# Niveles de Competencia

Nivel	Descripción	Calificaciones
No alcanza	No se observa capacidad de entendimiento.	1, 2, 3
En desarrollo	Logra comprender algunos conceptos, pero no todos.	4, 5
Competente	Comprende todos los conceptos.	6, 7
Promovido	Puede explicar los fundamentos detrás de los conceptos.	8, 9
Avanzado	Propone nuevos fundamentos o conceptos.	10

# Aprobación Directa

- No haber sido reincorporado
- Nota Trabajos y Nota Exámenes, **ambas** mayor o igual a 8
- Hay tres instancias de determinación del estado de Aprobación Directa:
  - En de Noviembre, antes de finalizar la cursada
  - En de Diciembre
  - En de Febrero.



# Playlist del Curso

- Agreguen a la playlist [SSL'22](#) los temas para antes de clase y para el intervalo



# Intervalo

15 minutos

# Primer Contacto con el Compilador

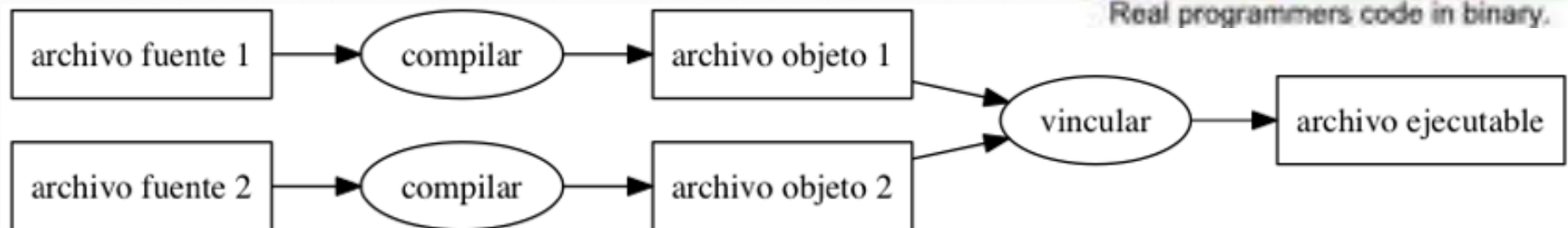
Lenguajes y Herramientas de Desarrollo

# ¿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



Real programmers code in binary.

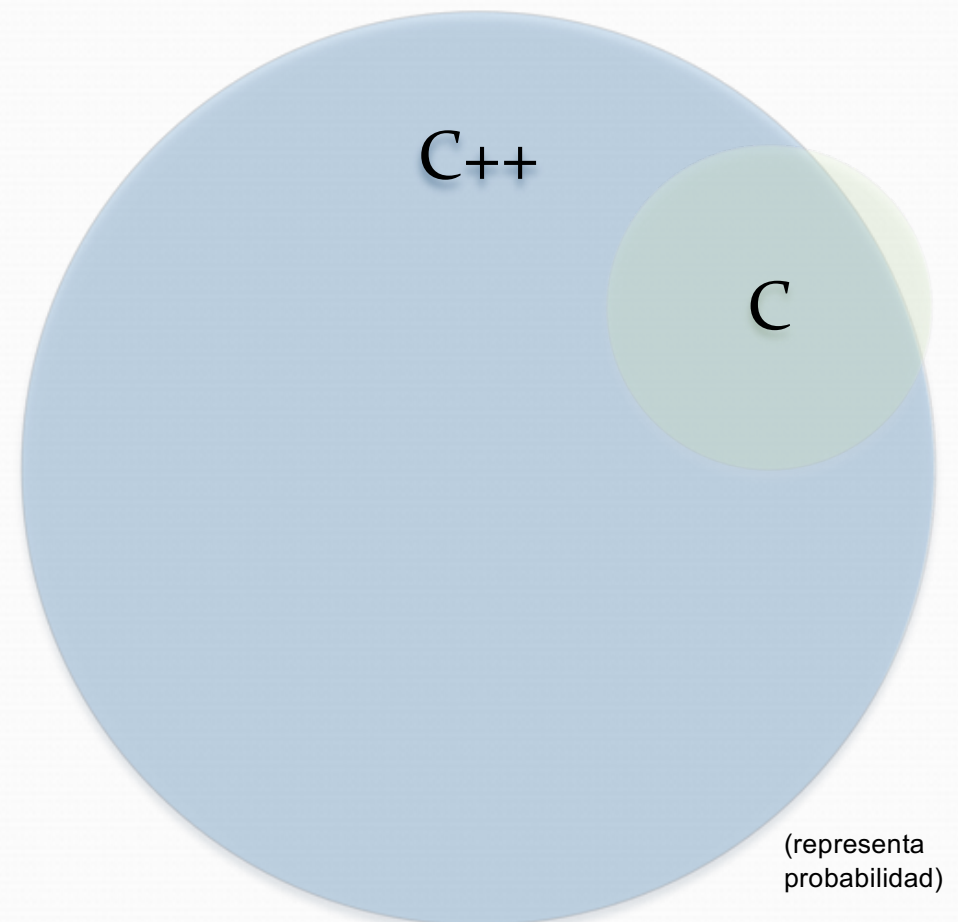


# Sobre los Lenguajes C y C++

## 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



# 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**



```
CHelloWorld — -bash — 60x8
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



```
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(117) : warning C4538: 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

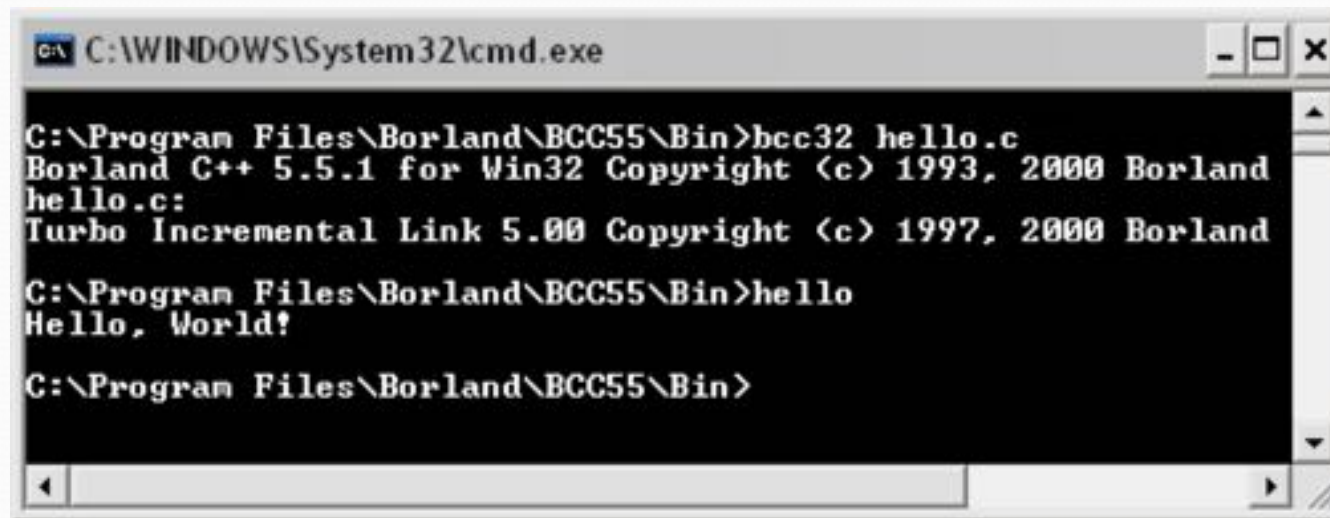
C:\Samples>hello
Hello World!

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



```
C:\WINDOWS\System32\cmd.exe

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 Enviroment*, Entorno Integrado de Desarrollo)
  - Ejemplos
    - Apple Xcode
    - Microsoft Visual Studio
    - Eclipse
- Sin IDE
  - Editor
  - Compilador.

## Con IDE

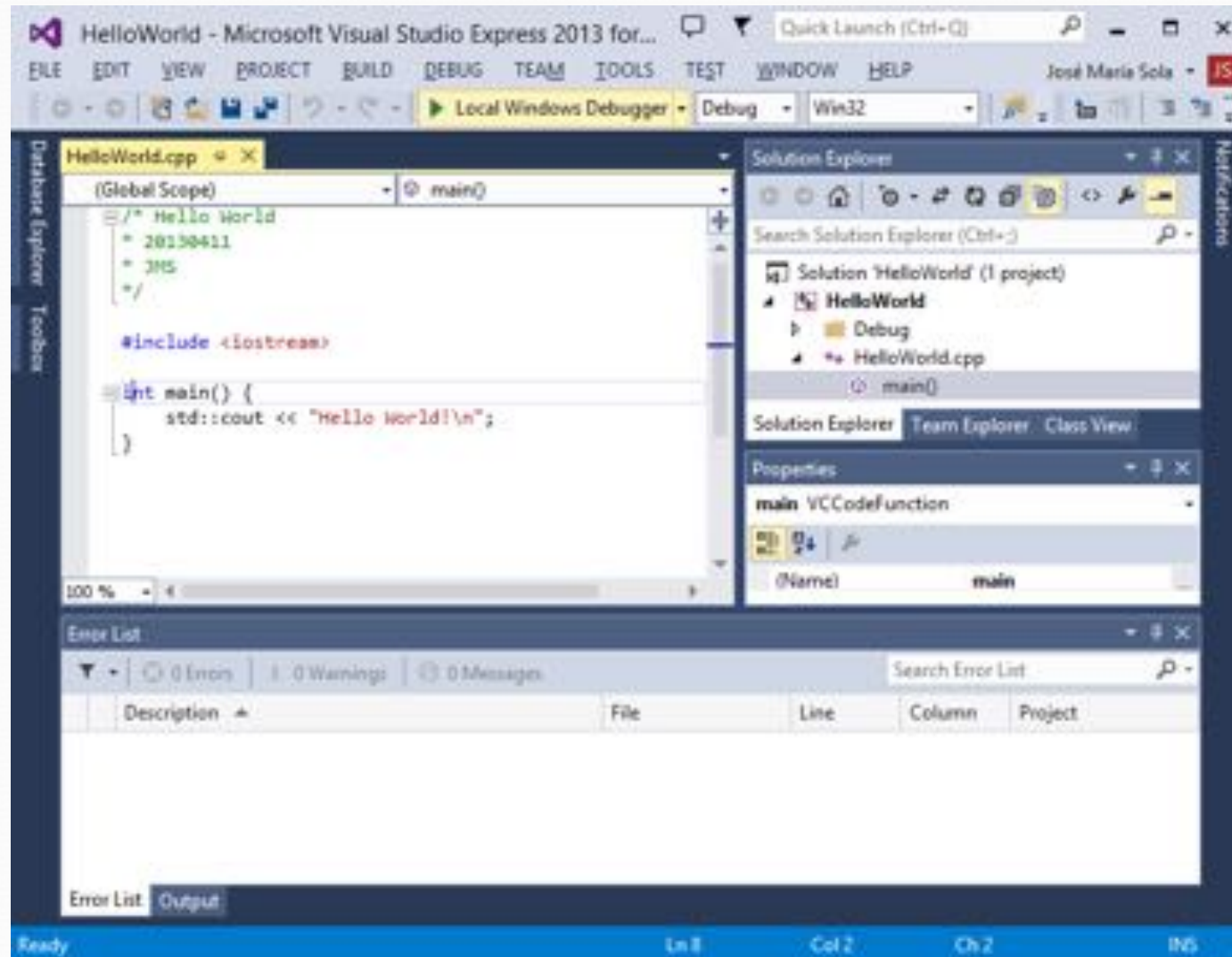
- Editor
- Depurador
- Gestor de Proyectos y de configuraciones
- Ayuda
- y más...

**Sin IDE**  
Requiere editor

Compilador de C/C++



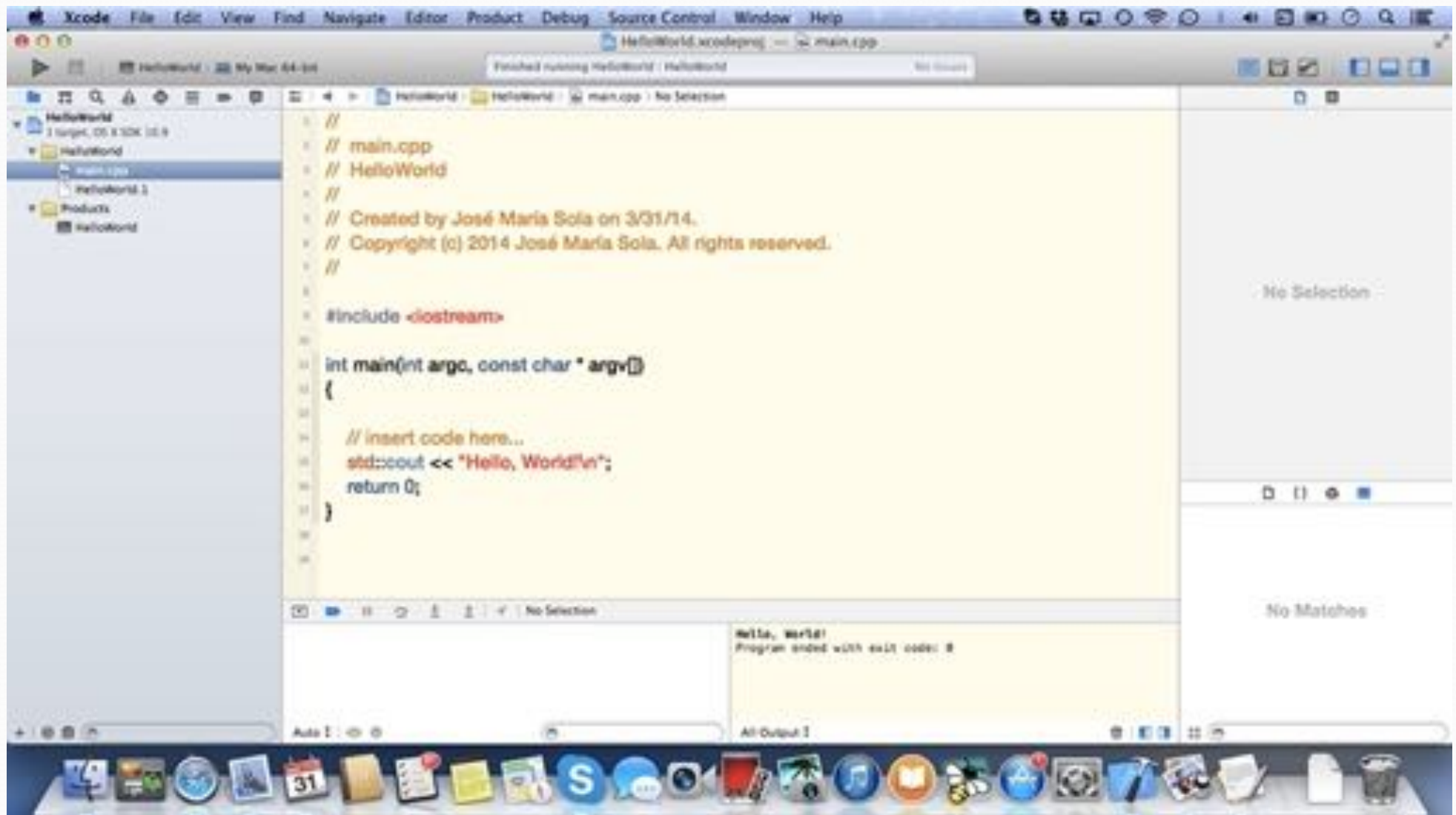
# Ejemplo con IDE Microsoft Visual Studio Express for Windows Desktop (Ejemplo en C++, no C)





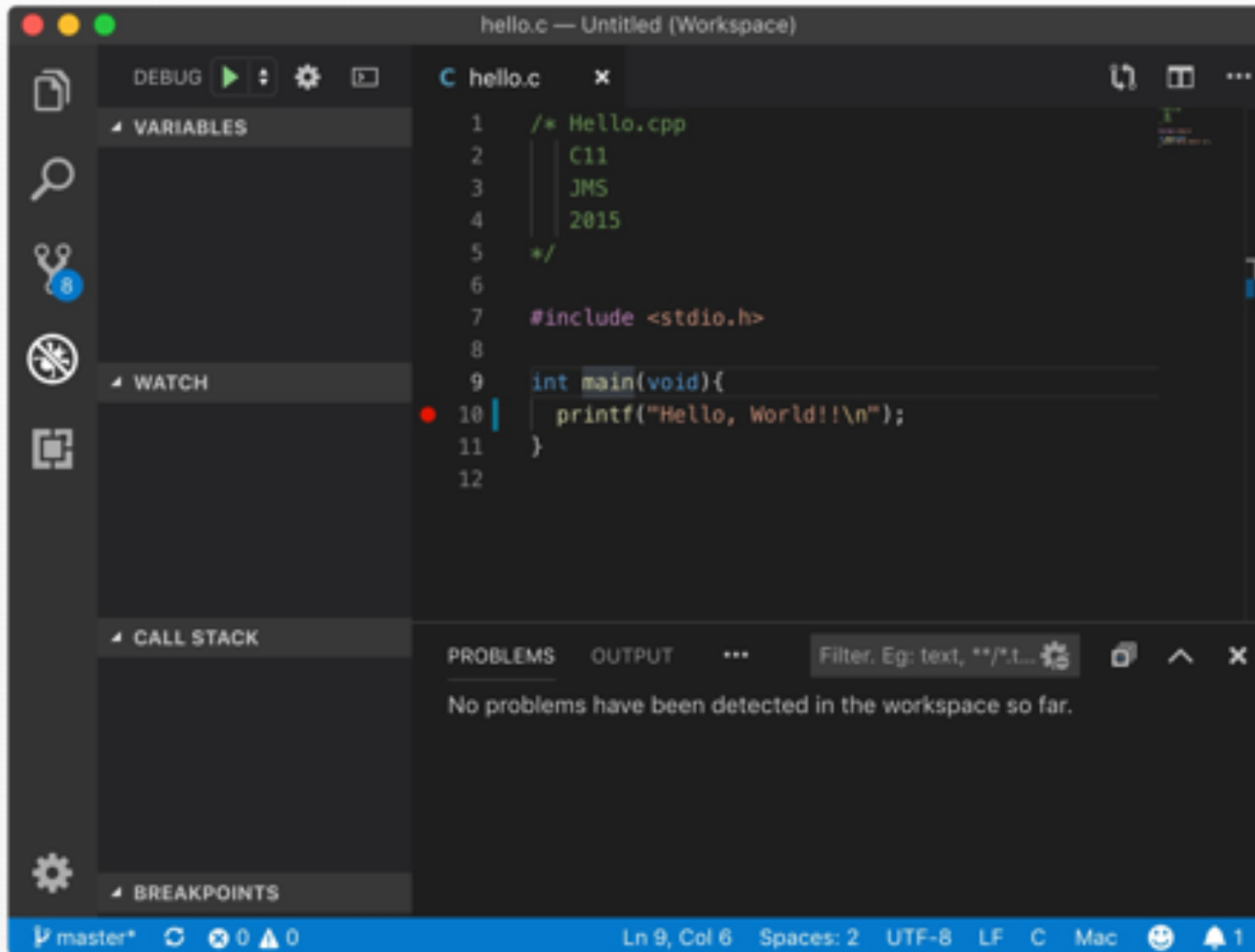
# Ejemplo con IDE

## Apple Xcode (ejemplo en C++, no C)

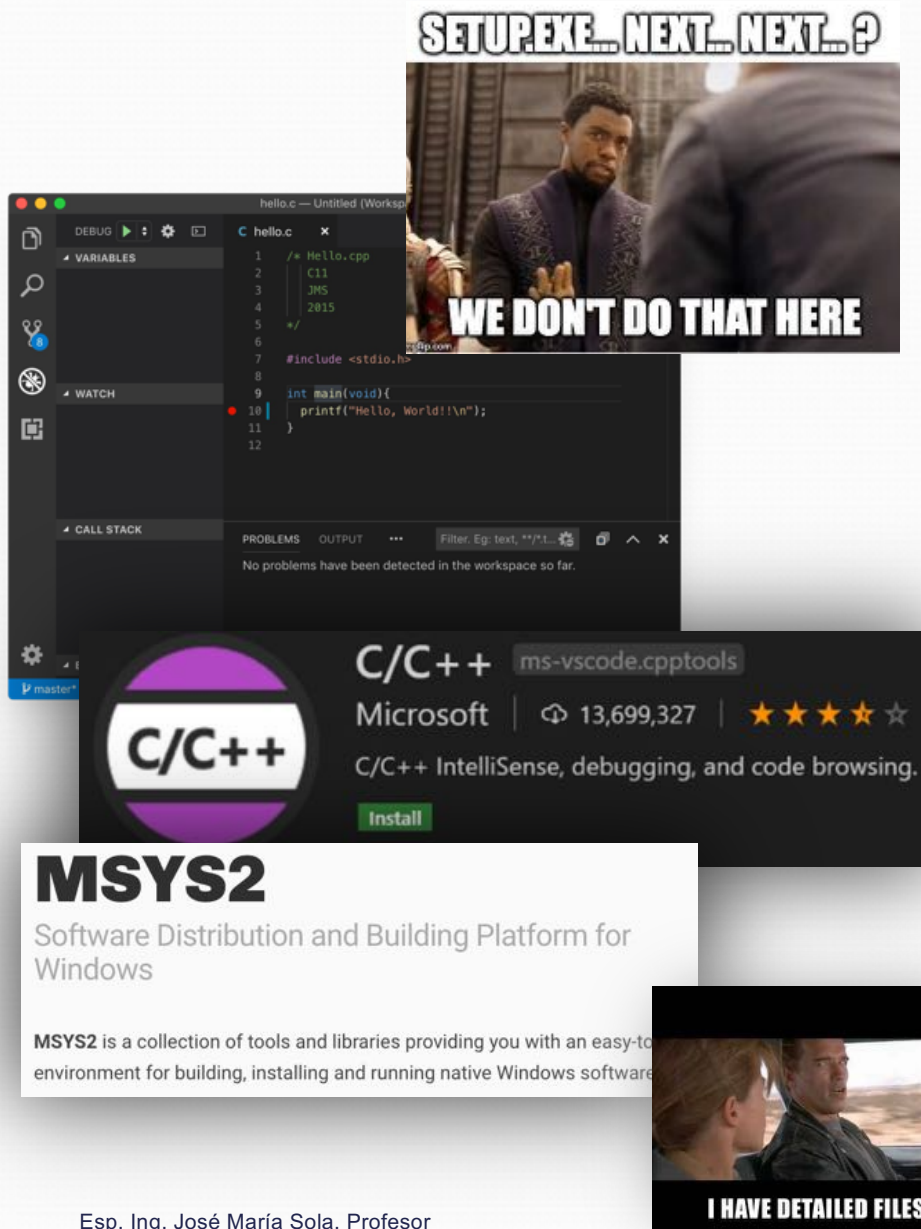


# Ejemplo con IDE

## Microsoft Visual Studio Code



# Instalación de Microsoft Visual Studio Code



- Es más que un editor de texto, es un editor de código fuente
- Hoy en día es muy popular
- Agrega funcionalidad mediante un sistema de extensiones, para nuestro caso: C/C++
- La extensión no incluye un compilador ☹, por lo que hay que conseguir uno, por ejemplo gcc, clang, o cl
  - Los sistemas Windows pueden obtener gcc mediante MinGW (the Minimalist GNU for Windows)
  - MinGW se puede obtener desde [MSYS2](#)
- La extensión C/C++ tiene documentación detallada sobre la instalación de todo el tool-chain

# 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-ake.exe` si instalamos *MinGW*.



```
CHelloWorld — -bash — 70x9
[$ 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/cpp-lusplus/>
  - CodeLite
    - <https://codelite.org>
  - Eclipse IDE for C/C++ Developers
    - <https://www.eclipse.org/downloads/package-s/release/2020-03/r/eclipse-ide-cc-developers-includes-incubating-components>
  - Más antiguos
    - Code::Blocks
      - <http://www.codeblocks.org/downloads/>
    - Dev-C++
- 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

- <https://josemariasola.wordpress.com/papers/#CompiladoresInstalacion>
- 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.

# Trabajo #0

"Hello, World!" en C

# Trabajo #0 — "Hello, World!" en C

- Enunciado en <https://josemariasola.wordpress.com/ssl/assignments/>
- 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 el usuario usuario GitHub en la lista indicada en el curso.
- Restricciones
  - La fecha y hora límite de entrega se publica en el calendario
  - La evaluación se hace con lo publicado en GitHub.

# Términos de la clase #01

Definir cada término con la bibliografía

- El Contexto de SSL
  - Ingeniería
  - Sistema
  - Información
  - Sistema de Información
  - Ingeniería en Sistemas de Información
  - Software
  - Programación
  - Problema (Necesidad)
  - Solución
  - Análisis
  - Diseño
  - Implementación
  - Desarrollo
  - Proyecto
  - Organización
  - Función
  - Autómata
  - Abstracción
- Tipo de Dato
- Orientación a Objetos
- Primer Contacto con el 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 Enviroment, Entorno Intedrado de Desarrollo)
  - Utilidad Make

# Tareas para la próxima clase

1. Comenzar a trabajar sobre el Trabajo #0



# ¿Consultas?



**Fin de la clase**