

# Clase #02 de 27

## Revisión Trabajo #0 y Proceso de Buildeo

*Abril 12, Lunes*

# Agenda para esta clase

- Revisión Trabajo #0
- Análisis y Síntesis de Hello.c
- Proceso de Buildeo

# Repaso Trabajo #0

“Hello, World!” en C

# Análisis y Síntesis de Hello.c

# Análisis de 'Hello, World!'

```
#include <stdio.h>
```

Incluye información acerca de la biblioteca estándar

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

Define una función llamada *main* que no recibe valores argumento. Las sentencias de *main* se encierran entre llaves

*main* llama a la función de biblioteca estándar *printf* para imprimir esa secuencia de caracteres. `\n` representa el carácter nueva-línea

- Estructura de un programa
  - vs. Pascal
    - Lineal vs. Jerárquico
- Rol de main
- Preprocesador
- Biblioteca estándar: Entrada/Salida
  - vs Framework
- Rol de llaves { }
- Función printf
- Pasaje de argumentos a funciones
- Constantes de cadena o literal cadena
- Secuencia de escape
- Punto y coma como terminador
  - vs. Pascal
- Analizar que es "palabra" del LP y que no
- Identificadores: main y printf --  
¿Cuál es el autómata finito que los reconoce?

C++ source #1

1

2

3

4

5

6

7

8

9

10

11

12

```
1 /* Hello.cpp
2  C11
3  385
4  2015
5  */
6
7 #include <stdio.h>
8
9 int main(void){
10     printf("Hello, World!\n");
11 }
12
```

#1 with x86-64 clang 4.0.0 x

x86-64 clang 4.0.0

-x c -std=c11

11010

LOG

Not

F

Info

1 main: # @main

2 push rbp

3 mov rbp, rsp

4 sub rsp, 16

5 movabs rdi, .L.str

6 mov al, 0

7 call printf

8 xor ecx, ecx

9 mov dword ptr [rbp - 4], ecx # 4-byte Spill

10 mov eax, ecx

11 add rsp, 16

12 pop rbp

13 ret

14

15 .L.str:

16 .asciz "Hello, World!\n"

17

• [www.CompilerExplorer.com](http://www.CompilerExplorer.com)



# Generalización de hello.c

## Programa General

- Un programa es una secuencia de funciones. Forma general:  
main  
f  
g  
...
- Forma general de una función  
Tipo Nombre ( Parámetros )  
{ Cuerpo }
- Cuando se corre (ejecuta) un programa, por convención, main es la primera función invocada por el ambiente de ejecución (e.g., sistema operativo). Todo programa debe tener un main con o sin parámetros

## hello.c

- Este programa define solo la función main
- Como toda función, main puede tener o no parámetros
- Este main invoca a printf con una cadena literal (cadena constante) como argumento. "Entre comillas".

# *Build:* Fases de la Construcción del Ejecutable



# Build: Proceso y Producto

- Procesadores y Procesos

1. Preprocesador
2. Compilador
3. Ensamblador
4. Linker

- Entradas y Resultados

- a. Fuente
- b. Fuente preprocesado
- c. Código ensamblador
- d. Código objeto
- e. Ejecutable (“Build”)

# Términos de la clase #02

Definir cada término con la bibliografía

- Análisis y Síntesis de hello.c
  - main
  - printf
  - Función del #include
  - puts
  - Secuencia de escape
  - Terminador de sentencia de C
  - Separador de sentencias de Pascal
  - Archivo .h (Encabezado)
  - String literal: Cadena Literal, o Literal de cadena, o Constante cadena, o Cadena constante
  - Lenguaje Ensamblador
  - Stack
  - sp
  - bp
- call
- Estructura lineal de un programa C
- Estructura jerárquica de un programa Pascal
- Build
  - Procesadores y Procesos
  - Preprocesador
  - Compilador
  - Ensamblador
  - Linker
  - Fuente
  - Fuente preprocesado
  - Código ensamblador
  - Código objeto
  - Ejectubale (“Build”)

# Tareas para la próxima clase

1. (Opcional) Agregar al readme.md de su repositorio SSL las notas que tomó durante clase sobre el proceso y producto de construcción (“bildeo”) y agregar información complementaria.

# ¿Consultas?

**Fin de la clase**