Laboratorio I

Introducción al Lenguaje “C”

FECyT
Facultad de Ciencias Exactas y Tecnológicas

UNSE
Universidad Nacional de Santiago del Estero

Introducción

- **C** es un lenguaje de programación creado en 1972 por Dennis Ritchie en los Laboratorios Bell como evolución del anterior lenguaje B.
- A pesar de tener casi 50 años de edad, sigue siendo una de las mejores opciones para la programación de los sistemas actuales y el medio más eficiente de aprendizaje para emigrar a los lenguajes dominantes de la actualidad como C++, Java o C#.
- La primera estandarización del lenguaje C fue a través del ANSI, con el estándar X3.159-1989. El lenguaje que define este estándar es conocido como ANSI C.
- La adopción de este estándar es muy amplia por lo que, si los programas creados lo siguen, el código es portátil entre plataformas y/o arquitecturas.

Laboratorio I – Versión Preliminar - Aldo Roldán

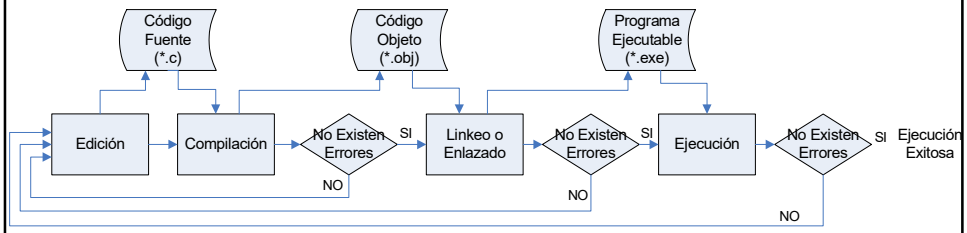
Introducción

- Algunas de las características que lo hacen tan popular son:
  - Es muy portable (transportable entre un gran número de plataformas de hardware y software - sistemas operativos)
  - Existen numerosos compiladores para todo tipo de plataformas sobre los que corren los mismos programas fuentes o con ligeras modificaciones.
  - Es versátil y de bajo nivel, por lo que es idóneo para tareas relativas a la programación del sistema.
  - A pesar de ser un excelente lenguaje para programación de sistemas, es también un eficiente y potente lenguaje para aplicaciones de propósito general.

Introducción

  - Es un lenguaje pequeño, por lo que es relativamente fácil construir compiladores de C y además es también fácil de aprender.
  - Todos los compiladores suelen incluir potentes y excelentes bibliotecas de funciones compatibles con el estándar ANSI.
  - El lenguaje presenta una interfaz excelente para los sistemas operativos Unix y Windows, junto con el ya acreditado Linux.
  - Es un lenguaje muy utilizado para la construcción de: sistemas operativos, ensambladores, programas de comunicaciones, intérpretes de lenguajes, compiladores de lenguajes, editores de textos, bases de datos, utilidades, controladores de red, etc.

Proceso de Creación de Programas



Proceso de Creación de Programas: Edición

- La **edición** es el proceso de crear o modificar el **código fuente** (source code).
- Se denomina **código fuente** de un programa al conjunto de instrucciones que se escriben, en este caso en el Lenguaje de Programación C.
- El mismo se almacena en un archivo de texto, el cual tiene la extensión **.c**.
- Este proceso por lo general se lo lleva a cabo utilizando un editor de texto; o bien mediante un **Entorno de Desarrollo Integrado** o **IDE (Integrated Development Environment)** el cual permite escribir, administrar, desarrollar y probar los programas.

Proceso de Creación de Programas: Compilación

- La **compilación** es el proceso de convertir el código fuente en lenguaje de maquina.
- Este proceso lo lleva a cabo el **compilador**, quien además detecta y reporta los errores que se produjeron durante el proceso de compilación.
- La entrada de esta etapa es el archivo que se produjo durante la edición, el cual es conocido como **código fuente;** y la salida producida por el compilador se conoce como **código objeto** y se almacena en archivos llamados **archivos objeto**, los cuales por lo general en el entorno Windows tienen la extensión .obj., y .o en los entornos *nux.

Laboratorio I – Versión Preliminar - Aldo Roldán

Proceso de Creación de Programas: Linkeo o Enlazado

- El **linkeo** o **enlazado** es el proceso que combina los distintos módulos generados por el compilador, agrega el código de los módulos de las librerías o bibliotecas proporcionadas como parte del lenguaje, y con todo ello genera un **programa ejecutable**.
- El **linker** es la herramienta encargada de este proceso, también detecta y reporta errores.
- En la practica, los proyectos se componen de varios archivos de código fuente, los cuales deberán ser linkeados para producir un único programa funcional.
- Por su parte, las librerías o bibliotecas de programas soportan y extienden el lenguaje proporcionando rutinas o funciones que lleven a cabo operaciones que no forman parte del lenguaje. Estas también se linkean para formar el ejecutable.

Laboratorio I – Versión Preliminar - Aldo Roldán

Proceso de Creación de Programas: Ejecución

- La etapa de **ejecución** es en la cual Ud. ejecuta u opera el programa ejecutable, una vez que se han completado todos los procesos previos de manera exitosa.
- En cualquiera de las etapas, un error implica por lo general retornar a la etapa de edición para chequear el código fuente.
- El proceso de edición, compilación, linkeo y ejecución son esencialmente los mismos para desarrollar programas en cualquier entorno y con cualquier lenguaje compilado.

Nuestro Primer Programa

```
/*
* Archivo:   main.c
* Descripcion: mi primer programa en C
*/
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[ ]) {
    printf("Bienvenidos!!!\n");
    return 0;
}
```

Análisis: Comentarios

- Observe toda la parte superior del programa.

```
/*
* Archivo:   main.c
* Autor: Aldo
*/
```

- En ellas se implementa un **comentario,** los cuales se utilizan para recordar algo, o documentar alguna cosa que lleva a cabo el programa, y así con el paso del tiempo pueda comprender como lo realizo y como trabaja el programa.
- Cualquier cosa existente entre **/*** y ***/** es tratado como un comentario, y este puede estar en una única línea o bien puede ocupar varias líneas.

Laboratorio I – Versión Preliminar - Aldo Roldán

Análisis: Los archivos cabecera

- Si se sigue analizando el ejemplo, se observan las siguientes líneas:

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

- Las mismas permiten la inclusión de los denominados **archivos cabecera**, los cuales contienen información especifica que el compilador utiliza para integrar cualquier función predefinada u otros objetos globales dentro de un programa, y tienen como extensión **.h**.
- Todo compilador de C que cumpla con el estándar ANSI para el lenguaje, tendrá un conjunto de archivos de cabecera proporcionados con el mismo, los cuales contienen declaraciones relacionadas con las funciones de la **librería estándar** que se encuentran disponibles con C.

Laboratorio I – Versión Preliminar - Aldo Roldán

Análisis: Los archivos cabecera

- En el ejemplo, debido a que sé esta utilizando la función **printf()**, resulta necesario incluir el archivo cabecera **<stdio.h>**, en virtud de que contiene toda la información que el compilador necesita para comprender lo que significa **printf**, así también como otras operaciones que se ocupan de la entrada y salida de información. Como su nombre lo indica, **stdio**, es una abreviatura de **standard input/output**.
- Por su parte, debido a que sé esta utilizando la constante **EXIT_SUCCESS** de la librería estándar, resulta necesario incluir el archivo cabecera **<stdlib.h>**
- Esta constante, cuyo valor es 0, indica al Sistema Operativo que la ejecución de la aplicación finalizo sin errores.

Laboratorio I – Versión Preliminar - Aldo Roldán

Análisis: La función main

- Las siguiente líneas de código definen la **función main()**

```
int main(int argc, char* argv[ ]) {
    printf("Bienvenidos!!!\n");
    return 0;
}
```

- Una **función** es un bloque de código con un nombre que lleva a cabo un conjunto especifico de operaciones.
- **Todo programa en C consta de una o más funciones**, y todo programa debe contener una función llamada **main( )** - ya que el programa comenzará la ejecución a partir del comienzo de esta función.

Laboratorio I – Versión Preliminar - Aldo Roldán

Análisis: La función main

- La primera línea define la función main

  int main(int argc, char* argv[ )]

- Comienza con la palabra clave **int** la cual establece el tipo de valor retornado por la función una vez que finalice su ejecución, que en este caso es un valor entero.
- Luego su nombre, **main**.
- Y por ultimo entre paréntesis los argumentos (datos), que recibe el programa desde el sistema operativo cuando es invocado para su ejecución.

Laboratorio I – Versión Preliminar - Aldo Roldán

Análisis: La función main

- La **estructura general del cuerpo de la función** main() es la siguiente

```
{
printf("Bienvenidos!!!\n");
return 0;
}
```

- El **cuerpo de la función** se encuentra entre las **llaves de apertura y de cierre (bloque de sentencias)** que siguen a la línea en donde aparece el nombre de la función. El cuerpo de la función contiene todas las sentencias que definen lo que realiza la función.
- Es importante destacar la importancia de la alineación de las llaves, afín de que quede reflejado manera clara donde comienza y donde termina un bloque de sentencias.

Laboratorio I – Versión Preliminar - Aldo Roldán

Análisis: La función main

- En el cuerpo de la función main( ) del ejemplo se incluyen la sentencias

  printf("Bienvenidos!!!\n");

  return 0;

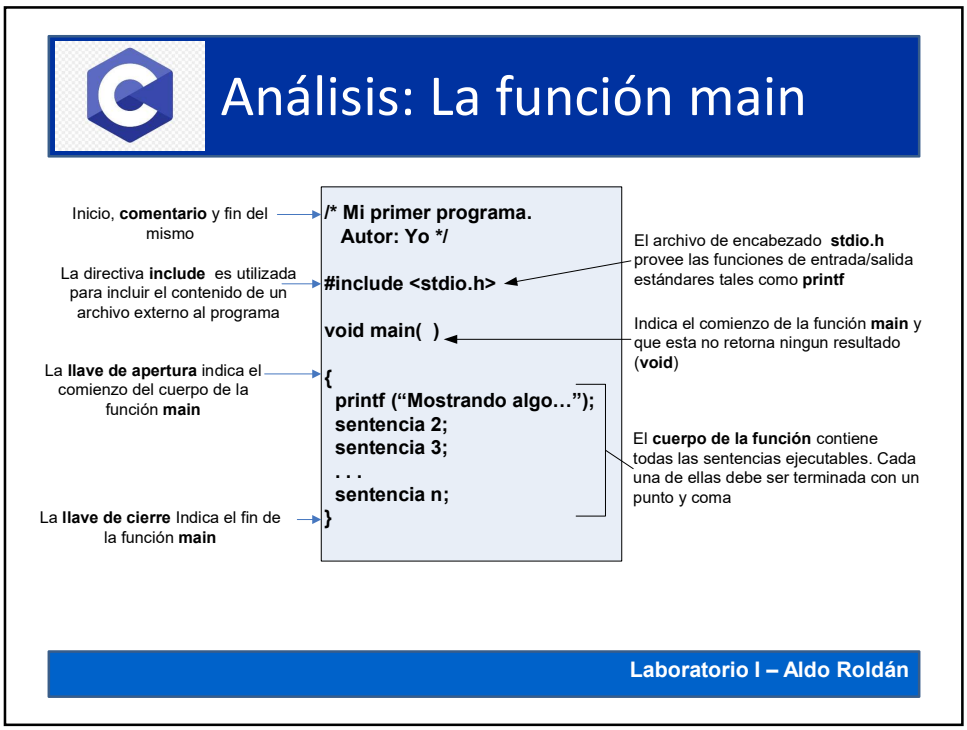
- **printf**() es una función de la librería estándar, y sirve para mostrar información en la pantalla
- En este caso imprime la cadena de caracteres “Bienvenidos!!!!” el cual recibe el nombre de **literal de tipo string**.
- Por su parte, con **return**( ) se finaliza la ejecución de la función main() y retorna el valor 0 al sistema operativo, para indicar que el programa termino de manera normal; mientras que un valor distinto de cero indicara una finalización anormal.

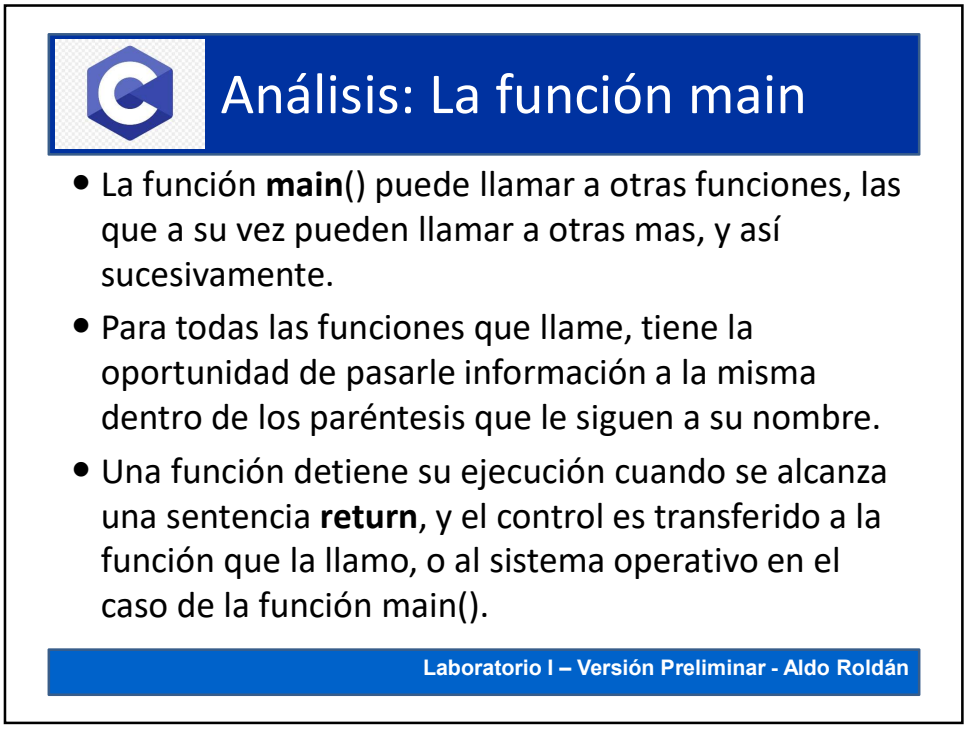
Laboratorio I – Versión Preliminar - Aldo Roldán

Análisis: La función main

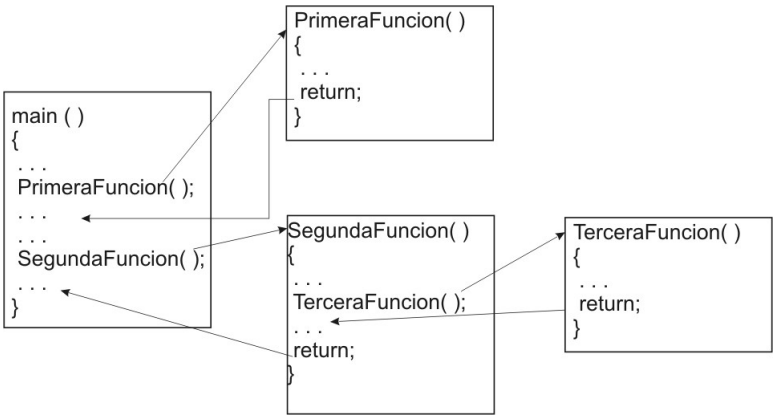
- Los ítems encerrados entre los paréntesis que siguen al nombre de una función, como los que se vieron en la función printf(), se llaman **argumentos.** Estos son los datos que se le pasan a una función para que cumpla con su objetivo. Cuando existe mas de un argumento que resulta necesario pasar a una función, ellos deben ser separados por comas.
- Como **toda sentencia ejecutable en C** (en oposición a las sentencias de definición o las directivas) la línea que contiene el printf() **debe tener un punto y coma al final.**

Laboratorio I – Versión Preliminar - Aldo Roldán


Análisis: La función main
Inicio, comentario y fin del mismo
La directiva include es utilizada para incluir el contenido de un archivo externo al programa
La llave de apertura indica el comienzo del cuerpo de la función main
La llave de cierre Indica el fin de la función main
/* Mi primer programa.
Autor: Yo */
#include <stdio.h>
void main( )
{
printf ("Mostrando algo...");
sentencia 2;
sentencia 3;
. . .
sentencia n;
}
El archivo de encabezado stdio.h provee las funciones de entrada/salida estándares tales como printf
Indica el comienzo de la función main y que esta no retorna ningun resultado (void)
El cuerpo de la función contiene todas las sentencias ejecutables. Cada una de ellas debe ser terminada con un punto y coma
Laboratorio I – Aldo Roldán


Análisis: La función main
La función main() puede llamar a otras funciones, las que a su vez pueden llamar a otras mas, y así sucesivamente.
Para todas las funciones que llame, tiene la oportunidad de pasarle información a la misma dentro de los paréntesis que le siguen a su nombre.
Una función detiene su ejecución cuando se alcanza una sentencia return, y el control es transferido a la función que la llamo, o al sistema operativo en el caso de la función main().
Laboratorio I – Versión Preliminar - Aldo Roldán

Resumen: estructura de un programa



Laboratorio I – Versión Preliminar - Aldo Roldán