	Práctica	
	Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

<i>Profesor:</i>	Alejandro Pimentel
<i>Asignatura:</i>	Fundamentos de Programacion
<i>Grupo:</i>	Grupo 3 Bloque 135
<i>No de Práctica(s):</i>	Practica #6
<i>Integrante(s):</i>	Muñoz Reyes Laura Vanessa
<i>No. de Equipo de cómputo empleado:</i>	Luxemburgo 10
<i>No. de Lista o Brigada:</i>	No. de Cuenta 3177522823
<i>Semestre:</i>	2020-01
<i>Fecha de entrega:</i>	30 de septiembre del 2019
<i>Observaciones:</i>	Excelente

CALIFICACIÓN: 10

Practica #6. Entorno de C

INTRODUCCION:

Una vez estudiado los algoritmos y diagramas de flujo como manera de representar procesos es momento de llevarlos al propio entorno de programación. En este curso se utilizara el Lenguaje de C, denominado como un lenguaje de nivel medio, puesto que combina elementos de alto nivel (Fortran,Pascal,Basic) con el funcionamiento del lenguaje ensamblador.

OBJETIVO:

Conocer y usar los ambientes y herramientas para el desarrollo y ejecución de programas en Lenguaje C, como editores y compiladores en diversos sistemas operativos.

MARCO REFERENCIAL:

Para empezar debemos definir que es un programa. Un programa puede entenderse como la forma de expresar la solución a un problema de manera que sea comprensible para el ordenador. En otras palabras, un programa es un conjunto ordenado de instrucciones que se dan a la computadora indicando las operaciones o tareas que se desea llevar a cabo para obtener un resultado. Existen diversos programas dependiendo el objetivo y la complejidad del problema.

C es un lenguaje de programación originalmente desarrollado por Dennis M. Ritchie entre 1969 y 1972 en los Laboratorios Bell, es un lenguaje orientado a la implementación de Sistemas Operativos, concretamente Unix. C es apreciado por la eficiencia del código que produce y es el lenguaje de programación más popular para crear software de sistemas, aunque también se utiliza para crear aplicaciones.

Se trata de un lenguaje de tipos de datos estáticos, débilmente tipificado, de medio nivel pero con muchas características de bajo nivel. Dispone de las estructuras típicas de los lenguajes de alto nivel pero, a su vez, dispone de construcciones del lenguaje que permiten un control a muy bajo nivel. Los compiladores suelen ofrecer extensiones al lenguaje que posibilitan mezclar código en ensamblador con código C o acceder directamente a memoria o dispositivos periféricos

Estructura básica de un programa en C

La mejor forma de aprender un lenguaje es programando con él. El programa más sencillo que se puede escribir en C es el siguiente:

```
main( )  
  
{  
  
}
```

Como nos podemos imaginar, este programa no hace nada, pero contiene la parte más importante de cualquier programa C y además, es el más pequeño que se puede escribir y que se compile correctamente. En él se define la función main, que es la que ejecuta el sistema operativo al llamar a un programa C.

PROCEDIMIENTO / RESULTADO:

1. Investigación texto plano

Un texto plano es un archivo de texto simple, texto sencillo o texto sin formato , es un archivo informático que contiene únicamente texto formado solo por caracteres que son legibles por humanos, careciendo de cualquier tipo de formato tipográfico.

En otras palabras, son archivos que contienen solo texto, pero no hay información sobre el tipo de letra, ni formas, ni tamaños. Técnicamente cualquier archivo puede abrirse como texto plano desde un editor de texto. Obviamente si se abriera un archivo de música MP3, una persona no entendería su contenido. En cambio si se abre un archivo HTML como texto plano, se vería el texto de la página web y todas las etiquetas que, procesadas, le darían un formato.

El clásico programa Bloc de notas (Notepad) de Windows maneja exclusivamente el texto plano. También otros programas como edit (DOS); ed, emacs, vi, vim, Gedit o nano (Unix, Linux), el SimpleText (Mac OS) o TextEdit (Mac OS X).

- **Txt**

Es un archivo informático que contiene únicamente texto formado solo por caracteres, careciendo de cualquier tipo de formato tipográfico.

Estos archivos están compuestos de bytes que representan caracteres ordinarios como letras, números y signos de puntuación (incluyendo espacios en blanco), también incluye algunos pocos caracteres de control como tabulaciones, saltos de línea y retornos de carro. Estos caracteres se pueden codificar de distintos modos. El sistema de codificación ASCII viene a ser la base primordial y no necesita de un identificador explícito en la comunicación digital.

- **Markdown**

Markdown es un lenguaje de marcado que facilita la aplicación de formato a un texto, empleando una serie de caracteres de una forma especial. En principio, fue pensado para elaborar textos cuyo destino iba a ser la web con más rapidez y sencillez que si estuviésemos empleando directamente HTML. Sin embargo actualmente también podemos emplearlo para cualquier tipo de texto, independientemente de cual vaya a ser su destino. Según uno de sus creadores es realmente dos cosas: por un lado, el lenguaje; y por otro, una herramienta de software que convierte el lenguaje en HTML válido.

- **HTML**

HTML es el lenguaje con el que se define el contenido de las páginas web. Básicamente se trata de un conjunto de etiquetas que sirven para definir el texto y otros elementos que la compondrán. El HTML se creó en un principio con objetivos divulgativos de información con texto y algunas imágenes. No se pensó que llegara a ser utilizado para crear área de ocio y consulta con carácter multimedia (lo que es actualmente la web), de modo que, el HTML se creó sin dar respuesta a todos los posibles usos que se le iba a dar y a todos los colectivos de gente que lo utilizarían en un futuro. El HTML es un lenguaje de marcación de elementos para la creación de documentos hipertexto, muy fácil de aprender, lo que permite que cualquier persona, aunque no haya programado en la vida, pueda enfrentarse a la tarea de crear una web.

- **La Tex**

Es un sistema de composición de textos, orientado a la creación de documentos escritos que presenten una alta calidad tipográfica. Por sus características y posibilidades, es usado de forma especialmente intensa en la generación de artículos y libros científicos que incluyen, entre otros elementos, expresiones matemáticas.

El documento tiene dos cosas llamativas: los comandos manera en la que nos comunicamos con LaTeX y los entornos que son fragmentos donde se aplica un formato.

- **CSV**

Un csv (comma-separated values) es un archivo de texto que almacena los datos en forma de columnas, separadas por coma y las filas se distinguen por saltos de línea.

El formato CSV no está estandarizado. La idea básica de separar los campos con una coma es muy clara, pero se vuelve complicada cuando el valor del campo también contiene comillas dobles o saltos de línea. Las implementaciones de CSV pueden no manejar esos datos, o usar comillas de otra clase para envolver el campo. Pero esto no resuelve el problema: algunos campos también necesitan embeber estas comillas, así que las implementaciones de CSV pueden incluir caracteres o secuencias de escape.

2. **Investigación editores de texto**

Un editor de texto es un programa que permite crear y modificar archivos digitales compuestos únicamente por texto sin formato, conocidos comúnmente como archivos de texto o texto plano.

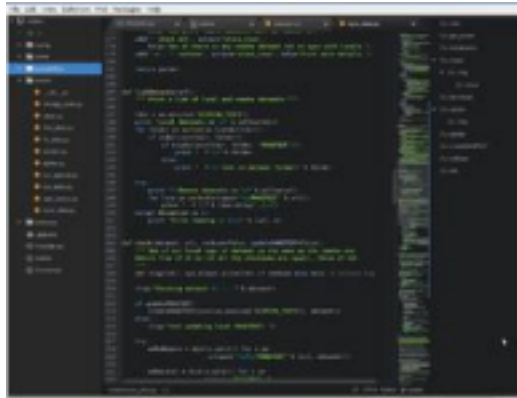
Los editores de texto son incluidos en el sistema operativo o en algún paquete de software instalado y se usan cuando se deben crear o modificar archivos de texto como archivos de configuración, scripts o el código fuente de algún programa. Los editores de textos "planos" se distinguen de los procesadores de texto en que se usan para escribir sólo texto, sin formato y sin imágenes, es decir sin diagramación.

Por lo tanto la diferencia entre un editor de un texto y un procesador es que el primero es un programa que permite crear y modificar archivos digitales compuestos por texto sin formato y el procesador de texto nos permite poner formato al texto.

Entre los principales editores de texto podemos encontrar:

- **Atom**

Es un editor de código de fuente de código abierto para macOS, Linux, y Windows1 con soporte para múltiples plug-in escritos en Node.js y control de versiones Git integrado, desarrollado por GitHub. Atom es una aplicación de escritorio construida utilizando tecnologías web.



- **BlueFish**

Bluefish está dirigido a diseñadores web experimentados y programadores y se enfoca en la edición de páginas dinámicas e interactivas.



- **Brackets**

Brackets es un editor de código fuente con un enfoque principal en el desarrollo web. Creado por Adobe Systems, es un software gratuito y de código abierto con licencia de la licencia MIT, y actualmente se mantiene en GitHub por Adobe y otros desarrolladores de código abierto.

- **Gedit**

Es un editor de textos compatible con UTF-8 para GNU/Linux, macOS y Microsoft Windows. Diseñado como un editor de textos de propósito general, gedit enfatiza la simplicidad y facilidad de uso.

- **Geany**

Geany es un editor de texto pequeño y ligero basado en Scintilla con características básicas de entorno de desarrollo integrado.

- **Emacs**

Es un editor de texto con una gran cantidad de funciones, muy popular entre programadores y usuarios técnicos. GNU Emacs es parte del proyecto GNU y la versión más popular de Emacs con una gran actividad en su desarrollo. El manual de GNU Emacs lo describe como "un editor extensible, personalizable, auto-documentado y de tiempo real."

- **Nano**

GNU nano es un editor de texto minimalista y amigable. Sin embargo, no solo nos permite editar texto, sino que además tiene otras características muy interesantes que lo hacen especialmente útil para modificar archivos de configuración en la terminal, crear lanzadores, y todo este tipo de acciones. Pero no solamente esto, puesto que al soportar coloreado de sintaxis, también puede ser utilizado para escribir código.

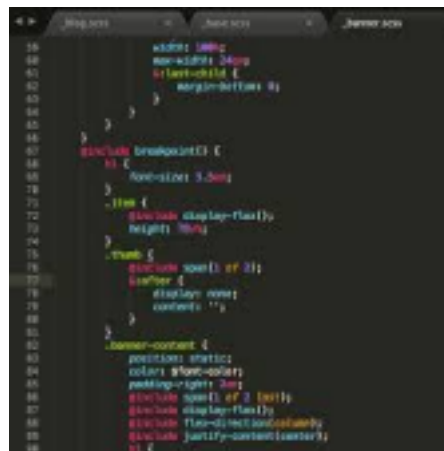


- **Pico**

Pico (Pine composer) es un editor de texto para Unix y sistemas basados en Unix. Está integrado con el cliente de correo electrónico Pine. Fue diseñado por la Oficina de Computación y Comunicaciones de la Universidad de Washington.

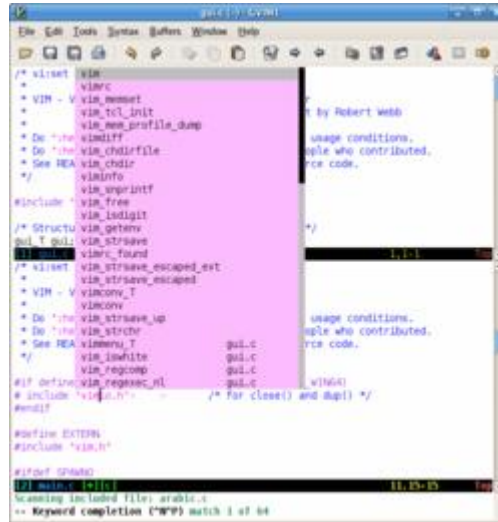
- **Sublime Text**

Sublime Text es un editor de texto y editor de código fuente está escrito en C++ y Python para los plugins. Desarrollado originalmente como una extensión de Vim, con el tiempo fue creando una identidad propia, por esto aún conserva un modo de edición tipo vi llamado Vintage mode.



- **Vim**

Vim es una versión mejorada del editor de texto vi, presente en todos los sistemas UNIX. Su autor, Bram Moolenaar, presentó la primera versión en 1991. La principal característica tanto de Vim como de Vi consiste en que disponen de diferentes modos entre los que se alterna para realizar ciertas operaciones.



En general, los editores difieren en su modo de uso y en las características que ofrecen.

- | | |
|---|---------------------------------------|
| → Resaltado de palabras clave | → Integración de compilador |
| → Autocompletado | → Integración de control de versiones |
| → Lista de elementos definidos | → Integración de terminal |
| → Autosangrado | → Búsquedas avanzadas |
| → Identificación de pares de paréntesis | → etc. |

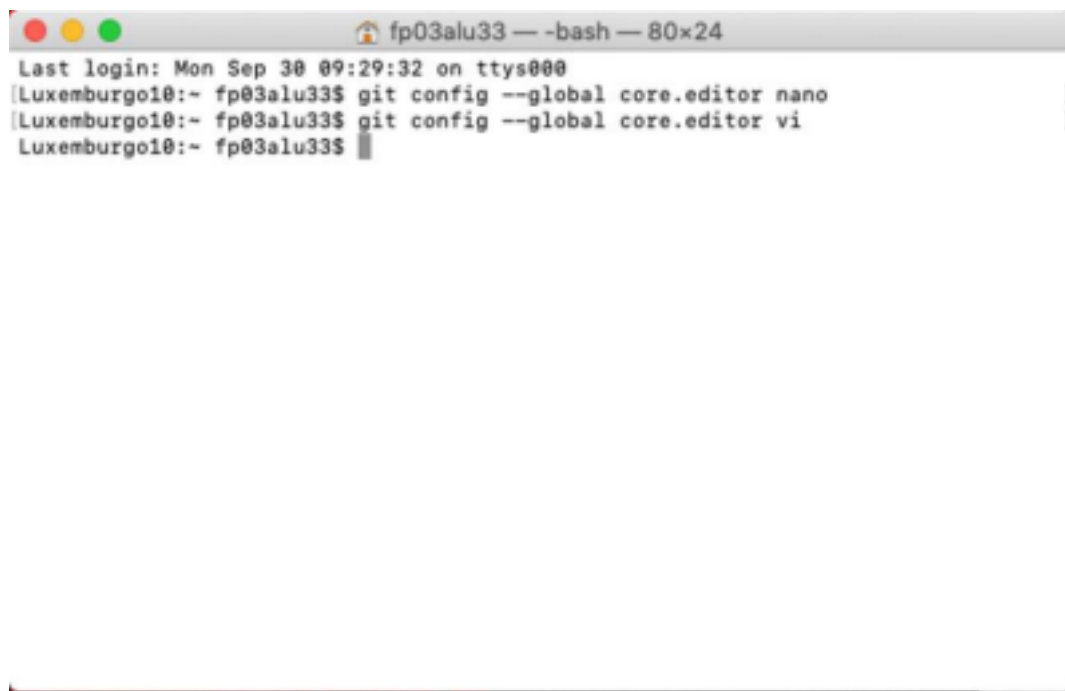
3. **Editor Nano**

Nano es un editor que se abre directamente en la terminal, por lo que percibí su uso es un poco más complicado que notepad o sublime text debido a que su diseño no es a lo que estamos acostumbrados normalmente.



```
fp03alu33 — nano — 80x24
GNU nano 2.8.6      New Buffer

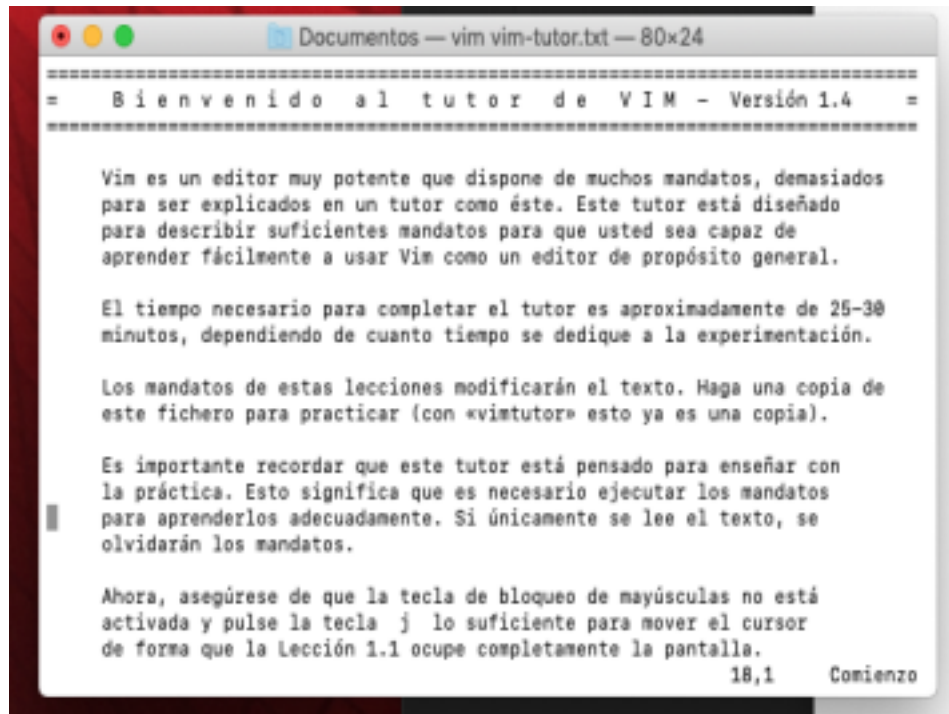
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```



```
fp03alu33 — -bash — 80x24
Last login: Mon Sep 30 09:29:32 on ttys000
Luxemburgo10:~ fp03alu33$ git config --global core.editor nano
Luxemburgo10:~ fp03alu33$ git config --global core.editor vi
Luxemburgo10:~ fp03alu33$
```


4. Editor Vim

Seguir el tutor de Vim



```
=====
=  Bienvenido al tutor de VIM - Versión 1.4  =
=====

Vim es un editor muy potente que dispone de muchos mandatos, demasiados
para ser explicados en un tutor como éste. Este tutor está diseñado
para describir suficientes mandatos para que usted sea capaz de
aprender fácilmente a usar Vim como un editor de propósito general.

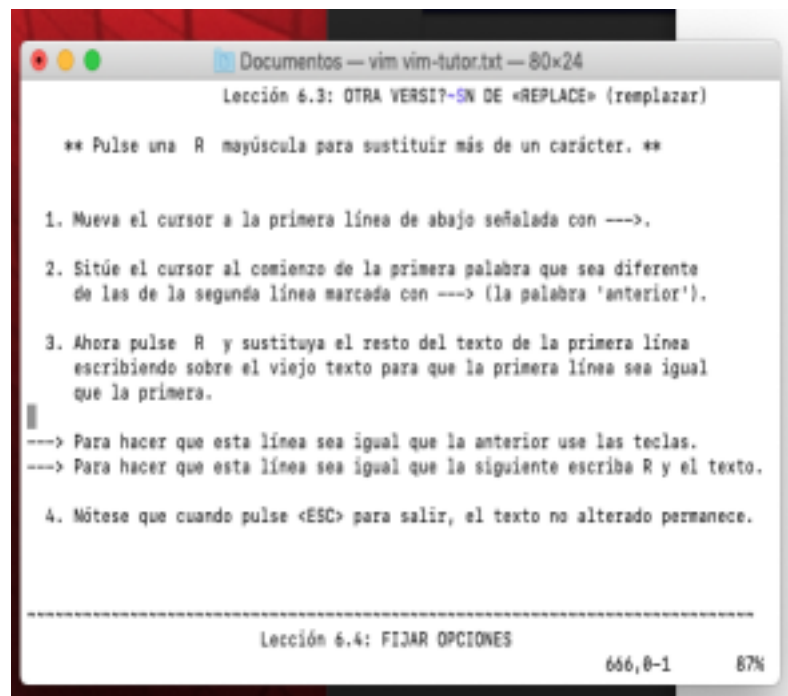
El tiempo necesario para completar el tutor es aproximadamente de 25-30
minutos, dependiendo de cuanto tiempo se dedique a la experimentación.

Los mandatos de estas lecciones modificarán el texto. Haga una copia de
este fichero para practicar (con «vimtutor» esto ya es una copia).

Es importante recordar que este tutor está pensado para enseñar con
la práctica. Esto significa que es necesario ejecutar los mandatos
para aprenderlos adecuadamente. Si únicamente se lee el texto, se
olvidarán los mandatos.

Ahora, asegúrese de que la tecla de bloqueo de mayúsculas no está
activada y pulse la tecla  j  lo suficiente para mover el cursor
de forma que la lección 1.1 ocupe completamente la pantalla.

                                           18,1      Comienzo
```



```
Documentos — vim vim-tutor.txt — 80x24
Lección 6.3: OTRA VERSIÓN DE «REPLACE» (reemplazar)

** Pulse una  R  mayúscula para sustituir más de un carácter. **

1. Mueva el cursor a la primera línea de abajo señalada con ---->.
2. Sitúe el cursor al comienzo de la primera palabra que sea diferente
   de las de la segunda línea marcada con ----> (la palabra 'anterior').
3. Ahora pulse  R  y sustituya el resto del texto de la primera línea
   escribiendo sobre el viejo texto para que la primera línea sea igual
   que la primera.
----> Para hacer que esta línea sea igual que la anterior use las teclas.
----> Para hacer que esta línea sea igual que la siguiente escriba R y el texto.

4. Nótese que cuando pulse <ESC> para salir, el texto no alterado permanece.

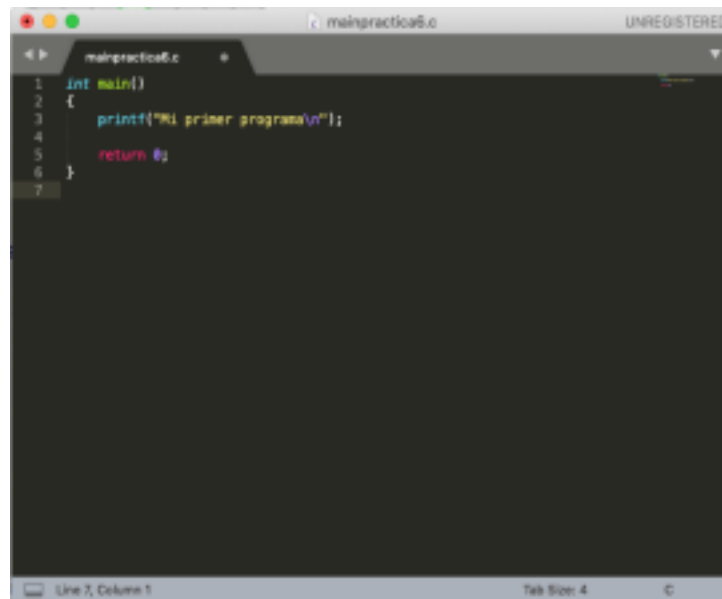
=====
Lección 6.4: FIJAR OPCIONES
                                           666,0-1      87%
```

Al principio me costó mucho trabajo mover el cursor solo con las letras J, K, L y H pero después me fui adaptando y efectivamente si resultaba más fácil navegar en el texto. Después al entrar y salir de vim no tuve ningún problema sin embargo resultaba un poco tedioso tener que repetir todos los pasos. Una de las cosas que me costó trabajo es irlo haciendo con práctica y no memorizando ya que tendía a regresarme a aprenderme cómo hacerlo.

Hay mandatos que son muy útiles como el de borrar una línea completa y que puedas manipular lo borrado de esa manera. De igual manera creo que los mandatos de ayuda en línea no deben pasar desapercibidos pues pueden sacarnos de apuros en muchos casos.

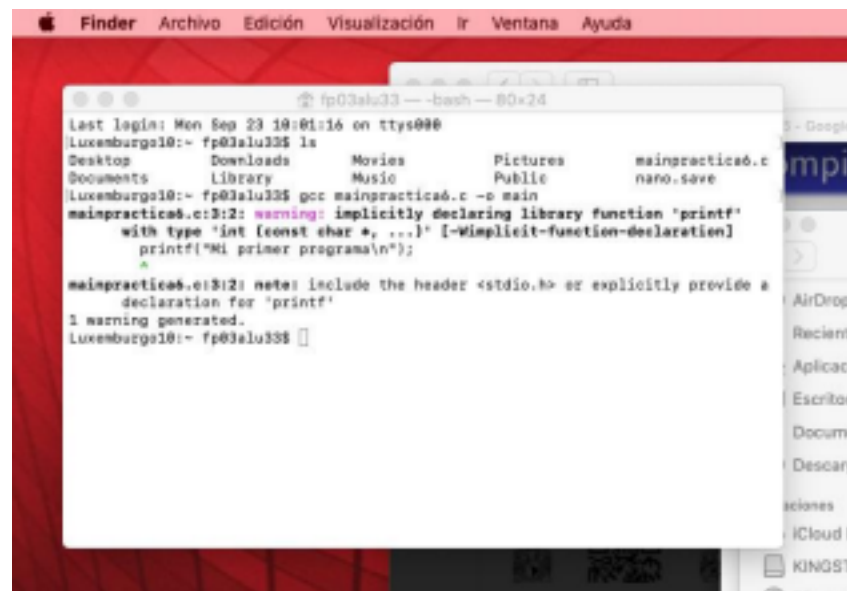
5. Notepad++ / Sublime Text

Como la práctica fue realizada en las computadoras Mac del laboratorio se utilizó Sublime Text en lugar de Notepad



```
mainpractica6.c
1 int main()
2 {
3     printf("Mi primer programa\n");
4
5     return 0;
6 }
7
```

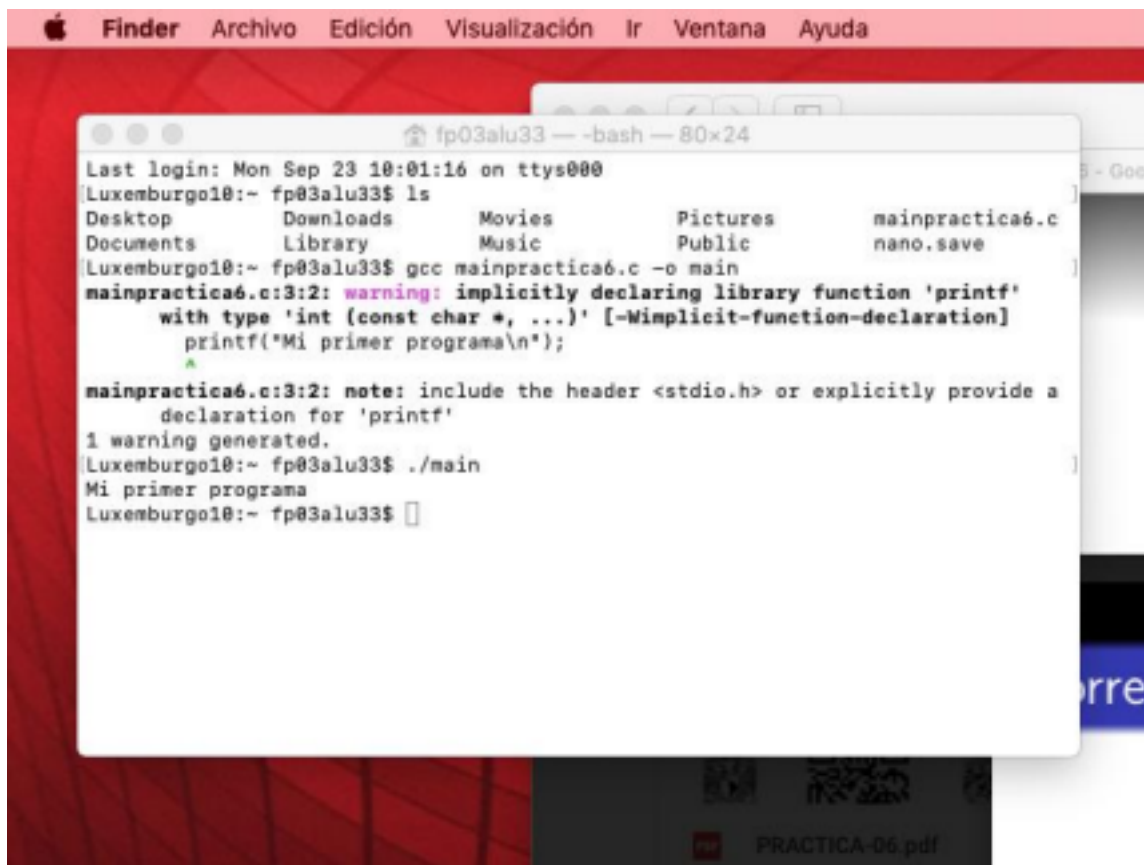
6. Compilar



```
Finder Archivo Edición Visualización Ir Ventana Ayuda
fp03alu33 - bash - 80x24
last login: Mon Sep 23 10:01:16 on ttys000
Luxemburgo10:~$ fp03alu33$ ls
Desktop Downloads Movies Pictures mainpractica6.c
Documents Library Music Public nano.save
Luxemburgo10:~$ fp03alu33$ gcc mainpractica6.c -o main
mainpractica6.c:3:2: warning: implicitly declaring library function 'printf'
      with type 'int (const char *, ...)' [-Wimplicit-function-declaration]
      printf("Mi primer programa\n");
      ^
mainpractica6.c:3:2: note: include the header <stdio.h> or explicitly provide a
      declaration for 'printf'
1 warning generated.
Luxemburgo10:~$ fp03alu33$
```

Al momento de compilar tuve un problema pues en lugar de warning me salía error y revisaba todo el procedimiento y estaba bien. Entonces decidí por cambiar el archivo de carpeta y así ya funciona.

7. Correr



The screenshot shows a macOS desktop with a red background. A terminal window titled 'fp03alu33 -- bash -- 80x24' is open. The terminal output shows the user 'Luxemburgo10' logging in and listing files in the current directory. The files listed are Desktop, Downloads, Movies, Pictures, mainpractica6.c, Documents, Library, Music, Public, and nano.save. The user then compiles the program 'mainpractica6.c' using 'gcc' with the output file 'main'. The compiler shows a warning about an implicitly declared library function 'printf' and a note to include the header '<stdio.h>' or provide a declaration for 'printf'. The user then runs the program './main', which outputs 'Mi primer programa'.

```
fp03alu33 -- bash -- 80x24
Last login: Mon Sep 23 10:01:16 on ttys000
Luxemburgo10:~ fp03alu33$ ls
Desktop      Downloads    Movies       Pictures     mainpractica6.c
Documents    Library      Music        Public       nano.save
Luxemburgo10:~ fp03alu33$ gcc mainpractica6.c -o main
mainpractica6.c:3:2: warning: implicitly declaring library function 'printf'
      with type 'int (const char *, ...)' [-Wimplicit-function-declaration]
  printf("Mi primer programa\n");
  ^
mainpractica6.c:3:2: note: include the header <stdio.h> or explicitly provide a
      declaration for 'printf'
1 warning generated.
Luxemburgo10:~ fp03alu33$ ./main
Mi primer programa
Luxemburgo10:~ fp03alu33$
```

CONCLUSIONES

Resulta difícil acostumbrarte ahora a un lenguaje C después de realizar los algoritmos en papel, sin embargo es mejor ya que la probabilidad de error es menor pues ya en el lenguaje C puedes probarlos al correrlos y así mismo verificarlos.

Fuentes

<https://drive.google.com/drive/folders/1up6yhEu5PQFLY4fJ7wBbk5shOvkrVODB>

<https://es.wikipedia.org/wiki/Vim>

www.tiendasvirtualesycomercioweb.com

lolap.wordpress.com

desarrolloweb.com