



Universidad Nacional Autónoma de México
Facultad de Ingeniería



Estructura de Datos y Algoritmos I

**Actividad 1: Repaso de lo que aprendí en la asignatura de
Fundamentos de Programación**

Sánchez Hernández Marco Antonio

Fecha: 26/febrero/2021

Introducción

Fundamentos de Programación es una asignatura correspondiente al primer semestre, del plan de estudios 2016, de la carrera en ingeniería en computación, impartida en la Facultad de Ingeniería de la Universidad Nacional Autónoma de México (UNAM). El objetivo expuesto dentro del plan de estudios de la asignatura es que, *“el alumno resolverá problemas aplicando los fundamentos de programación para diseñar programas en el lenguaje estructurado C, apoyándose en metodologías para la solución de problemas”* (anónimo, 2016), esto se pretende lograr a través de un curso teórico-práctico.

Para la parte práctica de la asignatura se cuenta con un manual de prácticas, el cual se compone de 13 prácticas en las cuales se cubre todo el temario teórico. Una característica importante de este manual es el hecho de que, las herramientas utilizadas, los conocimientos y habilidades que se pretende adquirir con este, son estándares en el campo laboral de la computación.

Otro punto que considero fue de mucha importancia al momento de adquirir conocimientos dentro de esta asignatura, fue la forma de trabajar y evaluar del profesor, pues permitía profundizar en los temas del mundo del cómputo que cada uno de los alumnos quisiera, ganando habilidades y conocimientos extra a los vistos durante clase. En este presente trabajo se expondrán, de forma breve y general, los conocimientos adquiridos durante esta asignatura, incluyendo algunos extra.

Conocimientos adquiridos

GitHub

Una parte fundamental en la programación es realizar modificaciones al código con el fin de optimizarlo, corregirlo, mejorarlo, etc., sin embargo, los cambios hechos no siempre resultan bien, por lo que es necesario regresar a estados anteriores del código, una forma de hacerlo es creando diferentes archivos y guardando cada uno de ellos, pero una forma más rápida y eficiente de realizar esto es a través de un control de versiones. GitHub se ha vuelto la herramienta estándar en este ámbito, pues no solo es utilizada para respaldar información, sino también para trabajos en equipo, compartir software, colaboración en proyectos de software libre, manejar páginas web, etc.

GitHub funciona mediante repositorios, donde será almacenada la información deseada, dentro de estos repositorios se encuentra por defecto una rama llamada main, en la cual se espera que se guarde el archivo final de un proyecto. Antes de realizar cambios sobre un archivo es importante analizar que el cambio a realizar sea correcto, así como observar que es lo nuevo de este archivo, para ello se crean más ramas, que sirven para , temporalmente, almacenar posibles cambios que puedan ser agregados al archivo final. Una vez se da el visto bueno a un cambio, se pueden guardar los cambios en el archivo final, un punto importante de GitHub es que lleva un historial de todos los cambios realizados, por lo que se puede volver a un estado más antiguo del proyecto.

GitHub también es una buena herramienta para llevar un historial de todos los proyectos que haz realizado, esto con el fin de tener evidencia de experiencia al momento de buscar trabajo en el campo laboral.

Arch Linux es un sistema operativo Linux que se caracteriza por sobre todos estos sistemas, por utilizar un método especial para descargar ciertas aplicaciones que no se encuentran directamente disponibles

dentro del sistema operativo, esto es mediante Arch User Repository, el cual utiliza principalmente GitHub, para facilitar la descarga de todo lo necesario para compilar un programa y ejecutarlo.

Búsquedas avanzadas

Una parte importante de la investigación y búsqueda de información, es la puntualidad de la búsqueda y la seriedad de las fuentes de consulta. Realizar búsquedas en Google suele ser tedioso, pues muchas de las primeras páginas que aparecen dentro del buscador, no cuentan con una seriedad o sustento científico o profesional que otorgue seguridad al momento de utilizar la información contenida en ellas, además que, la existir miles de páginas, no resultaría práctico revisar página por página para encontrar la información requerida. La búsqueda avanzada permite a los usuarios realizar búsquedas puntuales, desde mostrar solo resultados que contengan cierta palabra en el título, hasta buscar única y exclusivamente en una cierta página.

Estas búsquedas son realizadas a través de comandos escritos en el buscador junto con la búsqueda, dentro de la práctica número 1, del manual de laboratorio, se muestran algunos comandos útiles para el motor de búsqueda Google Search, sin embargo, muchos de estos comandos funcionan de igual manera en otros buscadores, como DuckDuckGo. Además de esto, otra opción importante para realizar búsquedas con carácter y sustento científico es Scholar Google, donde toda la información mostrada contará con una cita y la mayor parte de esta proviene de páginas con alto rigor científico y profesional.

GNU y el software libre

El software libre es una de las partes más importantes de la computación hoy en día, pues ha permitido el desarrollo de nuevas tecnologías con mayor rapidez, y que los usuarios tengan un mayor acercamiento al desarrollo de software. Hoy en día existe una contraparte “free and open source” prácticamente para cualquier software propietario, uno caso particular es LibreOffice como alternativa a Microsoft Office, así como para cada necesidad que el usuario tenga.

Lo primero, y más importante, a comprender del software libre, es el significado de “libre”, pues muchas veces el concepto de software o “free software” en inglés, genera una pequeña confusión entre las personas. Que un software sea libre, no significa que sea gratuito, aplicaciones como Arduor o Reaper son software libre, sin embargo, tienen un costo monetario. El software libre tiene sus bases en cuatro principios:

- Libertad de ejecutar un programa.
- Libertad de estudiar y modificar un programa.
- Libertad de redistribuir copias.
- Dar acceso al código fuente.

A partir de estos cuatro principios, se entiende como software libre todo aquel software, gratuito o de paga, que su código sea de acceso público para cualquier persona, y que esta persona tenga la libertad de editar y redistribuirlo. Aunque es importante mencionar que la redistribución de software, el que se puede editar y que no, está regulado bajo la Licencia Pública General de GNU.

GNU es un sistema operativo basado en Unix, que utiliza generalmente un kernel Linux. GNU ha sido un proyecto que ha se ha encargado de hacer crecer el uso de software libre y la distribución de este.

Dentro de su página se puede encontrar un enorme listado de software que cumple con ser software libre. Otra parte importante a mencionar de GNU y el software libre, es que dentro de la lista anteriormente mencionada, solo hay software que, además de cumplir con los principios, respeta plenamente la privacidad de los usuarios. Es por esta razón que, el software libre ha representado una clave para la privacidad cibernética.

Otra parte importante del software libre es la Free Software Foundation (FSF), fundada por Richard Stallman, la cual se encarga de toda la parte legal, es decir, la Free Software Foundation es la que ha hecho que el software libre sea una realidad, y que hoy en día lo siga siendo. Una de las metas principales planteadas este último año es que el software libre sea parte de la educación en todas las instituciones.

El sistema operativo Linux

La práctica número 2, del manual de laboratorio de la asignatura, consistía en hacer uso de los comandos básicos de un sistema operativo Linux. Debido a que la modalidad de las clases era a distancia, las prácticas debían ser realizadas a través de una conexión remota a un servidor de la Facultad de Ingeniería, esto limitó el contacto con el sistema operativo, impidiendo ver el potencial que este presentaba. Sin embargo, el profesor expuso la opción de instalar una distribución Linux, este caso Ubuntu, virtualmente por medio de una máquina virtual.

Llevar a cabo esta opción representó una enorme adquisición de nuevos conocimientos y un verdadero acercamiento al software libre. El mayor conocimiento a destacar es el uso de un sistema operativo Linux, pues además de utilizar comandos, era posible navegar por todo el sistema operativo, sin restricción, descargando y utilizando software.

Las prácticas de elaboración de programas debían ser realizadas en lenguaje C, para ejecutar código en este lenguaje se debe contar con un compilador, como gcc, el cual podía resultar algo tedioso de instalar y configurar en el sistema operativo Windows o MacOs, algo que en Linux no ocurría, pues al estar escrito en lenguaje C, contaba con un paquete que contenía el compilador gcc, el cual solo requería de ejecutar una sola línea de comando para instalarlo y usarlo, por esta última razón, decidí realizar todas las prácticas de programación dentro de algún sistema operativo Linux.








El acercamiento con Linux y el software libre, despertaron en mi un enorme interés por aprender más acerca de este sistema operativo, al grado cambiar completamente de Windows a Linux, y utilizar software libre en el día a día. Gracias esto, pude comprender con mayor claridad, qué es el software libre, las ventajas, desventajas, limitaciones y problemáticas que hoy en día presenta.

Algoritmos, diagramas de flujo y pseudocódigo

Un algoritmo es un conjunto finito de pasos que se deben seguir para dar solución a un problema, los algoritmos son la parte más importante en el desarrollo de software, pues estos garantizarán que la solución dada sea eficiente y pueda ser duplicada siguiendo este. Un algoritmo debe ser claro y evitar ambigüedades.

Antes de realizar un algoritmo es importante identificar los datos de entrada y de salida del problema a resolver. Un algoritmo ayuda a diseñar una solución general a un problema, el cual puede ser aplicado a cualquier lenguaje de programación deseado.

Los algoritmos son una excelente herramienta para encontrar la solución más eficiente a un problema, pero puede no ser lo suficientemente claro y visual para observar el flujo que debe tener la información, es aquí cuando se hace uso de los diagramas de flujo. Un diagrama de flujo es una representación gráfica de un algoritmo que a base de simbología representa las acciones a realizar para llegar al final de este.

SIMBOLOGÍA			
SÍMBOLO	NOMBRE	SÍMBOLO	NOMBRE
	Flecha de flujo.		Subproceso
	Comentario o anotación		Documento/ Impresora
	Inicio o finalización		Salida en Pantalla
	Proceso		Datos o Entrada/Salida
	Decisión		Referencia en página
			Referencia a otra página

Arguelles J. (2020). Simbología de diagramas de flujo. Figura [4]. Recuperado de <https://platzi.com/tutoriales/1444-pensamiento-logico/6052-que-es-un-diagrama-de-flujo/>

Un diagrama de flujo nos ayuda a tener una visión más clara de como es que podemos realizar el código de un programa para resolver un determinado problema.

Otra herramienta importante para representar el flujo de un algoritmo, previo a desarrollar un programa, es la creación de un pseudocódigo, este ayudará a visualizar la estructura básica que tendrá el código del programa, siendo útil para observar que tan eficiente es la forma en la que hemos decidido diseñar el programa.

El uso de estos tres elementos previo a desarrollar un programa ahorra tiempo al momento de programar el funcionamiento básico del programa, y ayuda a consumir la mínima cantidad de tiempo y recursos necesarios para resolver este problema, cabe resaltar que no todos los problemas actualmente son computables, pues hay algoritmos que representan problemas incluso para las computadoras. Un famoso problema matemático que ejemplifica esto es el problema P vs NP.

Programación en lenguaje C

El lenguaje C puede parecer difícil y tedioso para algunos usuarios, pues al trabajar directamente con la memoria de la computadora, es factible cometer errores en el almacenamiento, manejo de datos, e incluso presentar problemas de compatibilidad entre sistemas operativos, este último punto me ocurrió

durante el desarrollo de mi proyecto final, donde el programa, por utilizar una librería especial e Linux, no podía ser ejecutado con facilidad en Windows o MacOS, así como también presentó problemas entre distribuciones Linux, pues por defecto, Manajaro Linux, no cuenta con las preferencias del sistema latinas, ocasionando que ciertos caracteres especiales no fuesen impresos en pantalla.

Sin embargo, C también es considerado como el mejor lenguaje de programación por muchos, pues ha sido utilizado para crear la mayor parte de las aplicaciones que hoy en día se utilizan, incluyendo sistemas operativos.

Para iniciar un programa en C, debemos llamar a librerías que nos brinden las funciones que requerimos, dentro de la librería estándar de C, existe una librería llamada “stdio.h” la cual es la librería estándar de entrada y salida, es decir, esta librería nos permite manejar las entradas y salidas de datos, tomando como entrada estándar el teclado, y como salida estándar la pantalla. La función utilizada en C para manejar las salidas de datos es printf(), y la función utilizada para manejar la entrada de datos es scanf().

Es importante mencionar que toda línea de código escrita en C, debe terminar con un punto y coma en caso de no terminar en corchetes.

La sintaxis para la declaración de variables en C es la siguiente:

```
tipoDato nombre;
```

Es importante mencionar que toda línea de código que no sea un encabezado o no termine en llaves, debe terminar con punto y coma.

En lenguaje C existen dos formas de declarar una constante:

```
#define nombre=valor;
```

```
const tipoDato nombre;
```

La diferencia radicaría en que, utilizar #define establece un valor, mientras que const permite que el usuario fije un valor, pero en ambos este no podrá ser cambiado una vez asignado.

C, como todo lenguaje, posee diferentes tipos de datos, los principales vistos en clase fueron los siguientes:

-int: entero.

-long int: entero largo.

-char: carácter.

-bool: booleano, solo puede tomar uno de dos valores, verdadero o falso.

-float: punto flotante, se refiere a los números reales.

-double: se refiere igualmente a los números reales.

Para asignar un valor leído a una variable se utiliza la siguiente sintaxis

```
scanf(“%q”, &name);
```

Donde la q corresponde al tipo de dato que fue leído, ejemplo

```
int num;  
scanf("%d", &num);
```

En C existen 5 operadores aritméticos que permiten realizar operaciones:

+: suma.
-: resta.
*: multiplicación.
/: división.
%: resto.
=: asignación

Otras operaciones pueden ser llevadas a cabo en C con la llamada a otra librería, como math.h

Secuencias de escape en C:

-\n: salto de linea.
-\t: tabulador horizontal.
-\a: alarma
-\r: retroceso de carro.
-\\: \.
-\'0\': carácter nulo.
-\: tabulador vertical.

Para construir programas más complejos se requiere de la toma de decisiones y de expresiones que cambien el flujo de un programa dependiendo si una condición se cumple o no.

if-else es la una comparación que ejecutará un bloque de código si se cumple una cierta condición, de lo contrario continuará con el flujo del programa, o ejecutará otro bloque de código en caso de utilizar else.

```
if (condición){  
    bloque de código  
}  
  
else {  
    bloque de código  
}
```

Para realizar estas comparaciones se utilizan los operadores lógicos:

==: igual a
!=: diferente de

<: menor que
>: mayor que
<=: menor igual que
>=: mayor igual que
!: no.
&&: y.
||: o.

Algunas ocasiones se deben comparar distintas condiciones, y anidar múltiples condicionales if-else, puede resultar bastante tedioso y poco práctico, para este tipo de problemas, existe la estructura de selección switch, la cual ejecutará una cierta línea de código dependiendo de la condición que se haya seleccionado.

```
switch(variable){  
    case <condición>:  
        bloque de código  
    break;  
}
```

Otro tipo de estructuras de control son las estructuras de repetición, las cuales repetirán un cierto conjunto de código mientras una condición se cumpla. En C, existen tres estructuras de repetición.

-for: estructura de repetición utilizada cuando conocemos el número de veces que el código se debe ejecutar.

```
for (<variable>; <condición>; <incremento>){  
    bloque de código  
}
```

-while: estructura de repetición utilizada cuando no se conoce el número de veces que el código se debe ejecutar.

```
while (<condición>){  
    bloque de código  
    Nota: el incremento se debe realizar dentro del bloque  
}
```

-do-while: estructura de repetición utilizada principalmente en la creación de menús, la particularidad y ventaja de esta estructura es que solo se ejecuta una vez y repite todo el bloque de código hasta que la condición sea falsa.


```
do{  
  
    bloque de código  
  
}while(<condición>);
```

Una parte importante en la escritura de código, es ejecutar el código para observar el correcto funcionamiento de este, a veces los errores cometidos son mostrados por el compilador porque se trata de errores de sintaxis, pero otras veces los errores son valores no esperados, o violaciones de segmento que se desconocen. Los depuradores son una herramienta utilizada para la ejecución de código y observar el funcionamiento de este, creando puntos de control (puntos donde el programa se detendrá), o mostrando el valor que la variable toma en ese momento, ayuda a los programadores a corregir y mejorar el código, la herramienta utilizada en clase fue gdb.

Los arreglos permiten almacenar una secuencia de datos del mismo tipo, pueden ser unidimensionales o multidimensionales, aunque el almacenamiento en memoria sería el mismo, lo que cambia es el índice. El índice se refiere a la posición que le corresponde a un cierto dato dentro del arreglo.

```
tipoDato nombre[<tamaño>];
```

Dentro del curso solo se utilizó un tamaño fijo, sin embargo, existe la posibilidad de tener un tamaño dinámico.

Las funciones son la base de toda la programación, existen funciones definidas en la librería estándar y algunas son creadas directamente por el usuario. Las funciones sirven para crear bloques de código que van a ser utilizados en diferentes momentos de un programa, ahorrando así líneas de código, algunos programadores prefieren separar las funciones creadas del archivo de código principal y guardarla en otro código.

```
tipoDato nombre (<parámetros>){  
  
    bloque de código  
  
}
```

Los parámetros son los valores de entrada que tendrá la función, pudiendo o no, retornar algún valor al código principal.

Si una función es declarada dentro de alguna función, el nombre de esta puede ser reutilizado en alguna otra sin generar algún conflicto, mientras que si la función es declarada fuera de toda función será considerada como global, y podrá ser usada en cualquier función.

Un tipo especial de dato que acompaña a las funciones es void, el cual no retornará valor alguno una vez ejecutado el código.

La información utilizada en un programa es temporal, por lo que, una vez terminada su ejecución, la información se perderá, una forma de almacenar la información es en un archivo de texto, esto se logra a través de apuntadores. Un apuntador hace referencia a la localidad en memoria de una variable.

Para el caso de los archivos, se pueden abrir en diferentes modos, como lectura “r”, escritura, “w”. esto a través de la siguiente función.

```
Name = fopen (“nombreDeArchivo.txt”, “<modo>”);
```

```
fclose(Name);
```

Cabe mencionar que la variable Name debe ser un apuntador, y al hacer referencia a un archivo, será de tipo FILE

```
FILE * Name;
```

Al realizar el proyecto final de la asignatura obtuve conocimientos de una librería extra del lenguaje C, ncurses, la cual permite crear ventanas dentro de la terminal, utilizando apuntadores del tipo WIN.

Conclusión

El amplio conocimiento reflejado adquirido en esta materia de debió a la forma en la que el profesor impartió su clase, pues además de darnos el temario y los conocimientos solicitados en el plan de estudios, contaba experiencias laborales, aplicaciones, y, finalmente, el proyecto final era de tema libre, no limitando las herramientas a usar, e incluso el lenguaje de programación a usar.

Referencias

Arguelles J. (2020). Platzi. *¿Qué es un diagrama de flujo?* Recuperado el 25 de febrero de 2021, de <https://platzi.com/tutoriales/1444-pensamiento-logico/6052-que-es-un-diagrama-de-flujo/>

Facultad de Ingeniería. (2016). Universidad Nacional Autónoma de México. *Proyecto de modificación del plan de estudios de la licenciatura de Ingeniería en Computación*. Recuperado de https://www.ingenieria.unam.mx/programas_academicos/licenciatura/Computacion/2016/asignaturas_computacion_2016.pdf

Solano J., García E., Nakayama A., Arteaga T., Castañeda M. (2018). Laboratorio de Computación Salas A y B. *Manual de prácticas del laboratorio de Fundamentos de programación*. Recuperado de <http://lcp02.fi-b.unam.mx/>