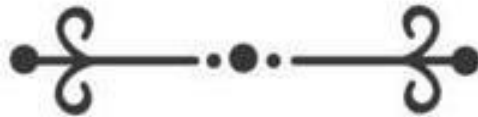


Universidad Nacional Autónoma de México



Facultad de Ingeniería



Estructura de Datos y Algoritmos I

Actividad 1 | Repaso de Fundamentos de Programación

Grupo 15

Matias Zavala Melissa Maruuati

M.I. Marco Antonio Martínez Quintana

24-febrero-2021

Hello, World!

Primeramente era necesario empezar con las bases, que nos servirán para estructurar y tener una guía de cómo hacer nuestro programa, una vez que ya lo tienes en mente eso se hace más mentalmente, pero de inicio se necesita un algoritmo, el cual es una secuencias de pasos a seguir, o también relacionado con las instrucciones para llegar a un objetivo; con lo que debe ir paso a paso hasta cumplir el fin.

Esto va de la mano con el pseudocódigo, siendo este segundo mucho más estructurado, al igual que teniendo palabras clave, con las cuales también nos será posible identificar la forma en un diagrama de flujo y la función que se utilizara en el código fuente.

Ya con los dos principales desarrollos del programa continuamos con el diagrama de flujo, el cual es una forma gráfica de representar eso que ya tenemos escrito, como su nombre lo dice siempre sigue un flujo, de principio hasta cumplir su última tarea, llamado el fin de este. Cada función tiene su figura específica y deben de conectarse con flechas entre sí, acorde a cómo será la dirección que estas van a llevar.

Simbología Estándar: Diagramas de Flujo de Datos.	
Inicio / Fin	
Procesos	
Entrada Datos	
Condición	
Conector	
Cinta Magnética	
Disco Magnético	
Conector de Pagina	
Líneas de Flujo	
Display, Mostrar Datos	
Enviar Datos a Impresora	

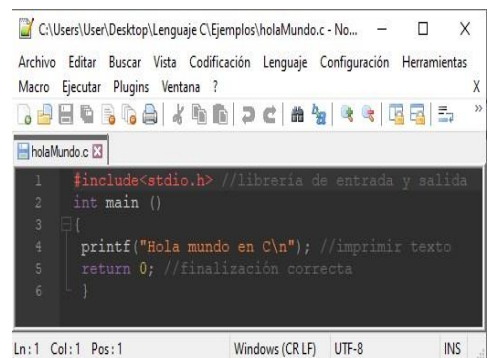
En el curso descargamos las aplicaciones y archivos necesarios para poder usar el compilador y correr los programas mediante el símbolo del sistema. Por lo cual

lo principal fue obtener a MinGW, la cual en el momento me presento un problema, ya que en la principal página de donde lo descargue no tenía el archivo donde tenía el comando de gcc, por lo que tuve que buscarlo por otra parte para que funcionara en el símbolo del sistema.

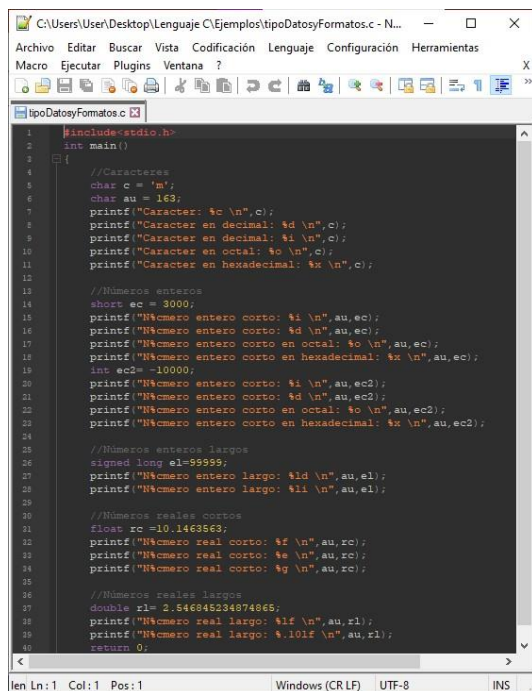
Después de ya tener bien el compilador, entonces siguió descargar la aplicación desde donde escribiríamos el código fuente, el cual tiene como nombre notepad++, con el cual ya no se tuvieron problemas de instalación y ejecución.

También gracias al proyecto final pude ver cómo es que se hace una gráfica de Gantt, la primera vez que lo hice me apoye de youtube, pero la siguiente ya sabía más como y pude darle un mejor formato a esta, quedando mejor acomodado y como un buen cronograma de las actividades realizadas.

Una vez tenía compilador y el código fuente fue momento de empezar a mostrar una escritura con el lenguaje C, para lo que se ocupa la palabra "printf", la cual ayudaba a mostrar un texto, el cual debía ir con paréntesis y entre comillas, ya que sin esto no lo tomaría como texto, con esto poner el clásico programa llamada "Hola mundo".



```
1 #include<stdio.h> //libreria de entrada y salida
2 int main ()
3 {
4     printf("Hola mundo en C\n"); //imprimir texto
5     return 0; //finalización correcta
6 }
```

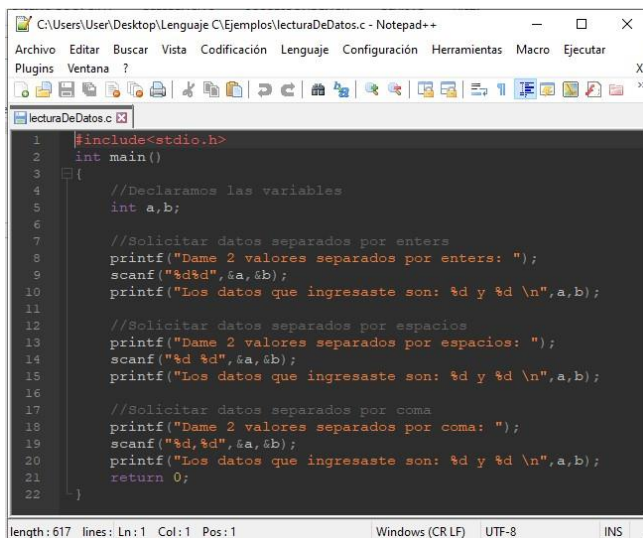


```
1 #include<stdio.h>
2 int main()
3 {
4     //Caracteres
5     char c = 'n';
6     char au = 163;
7     printf("Caracter: %c \n",c);
8     printf("Caracter en decimal: %d \n",c);
9     printf("Caracter en octal: %o \n",c);
10    printf("Caracter en hexadecimal: %x \n",c);
11
12    //Numeros enteros
13    short ec = 3000;
14    printf("Número entero corto: %i \n",au,ec);
15    printf("Número entero corto: %d \n",au,ec);
16    printf("Número entero corto en octal: %o \n",au,ec);
17    printf("Número entero corto en hexadecimal: %x \n",au,ec);
18    int ec2 = -10000;
19    printf("Número entero corto: %i \n",au,ec2);
20    printf("Número entero corto: %d \n",au,ec2);
21    printf("Número entero corto en octal: %o \n",au,ec2);
22    printf("Número entero corto en hexadecimal: %x \n",au,ec2);
23
24    //Numeros enteros largos
25    signed long el=99999;
26    printf("Número entero largo: %ld \n",au,el);
27    printf("Número entero largo: %li \n",au,el);
28
29    //Numeros reales cortos
30    float rc =10.1463563;
31    printf("Número real corto: %f \n",au,rc);
32    printf("Número real corto: %e \n",au,rc);
33    printf("Número real corto: %g \n",au,rc);
34
35    //Numeros reales largos
36    double rl= 2.546845234874865;
37    printf("Número real largo: %lf \n",au,rl);
38    printf("Número real largo: %le \n",au,rl);
39    return 0;
40 }
```

Vimos los diferentes tipos de datos que era posible utilizar en este lenguaje, así como la forma en el que debía escribirse, siendo en todos el signo de porcentaje "%", seguido de la letra por la cual se presentaba dicho formato y estando en el texto del printf, pero al final de las comillas separado por comas seguían la variable que está representada con el tipo de variable, el formato que se haya pedido este.

Otra cosa fue los saltos de renglón representados con "\n", se usa para tabular "\t" y "\a" reproduce un sonido.

Una función muy útil para no olvidar la función de cada una de tus líneas o cada apartado es poner los comentarios poner `/**` puede agregar comentarios de un solo renglón, lo cual significa que después de ese puedes escribir simplemente en ese renglón donde lo has puesto, en cambio si te es necesario poner uno de mayor extensión y que abarque 2 o más renglones es necesario poner al principio del primer renglón del comentario `/*` y en el renglón final del mismo se debe poner `*/` lo cual cerrara que hasta ahí es y no afectara a tu programa, cabe destacar que estos no aparecen en la ejecución del programa, sino simplemente en tu código fuente y sirve de guía.



```
1 #include<stdio.h>
2 int main()
3 {
4     //Declaramos las variables
5     int a,b;
6
7     //Solicitar datos separados por enters
8     printf("Dame 2 valores separados por enters: ");
9     scanf("%d%d",&a,&b);
10    printf("Los datos que ingresaste son: %d y %d \n",a,b);
11
12    //Solicitar datos separados por espacios
13    printf("Dame 2 valores separados por espacios: ");
14    scanf("%d %d",&a,&b);
15    printf("Los datos que ingresaste son: %d y %d \n",a,b);
16
17    //Solicitar datos separados por coma
18    printf("Dame 2 valores separados por coma: ");
19    scanf("%d,%d",&a,&b);
20    printf("Los datos que ingresaste son: %d y %d \n",a,b);
21    return 0;
22 }
```

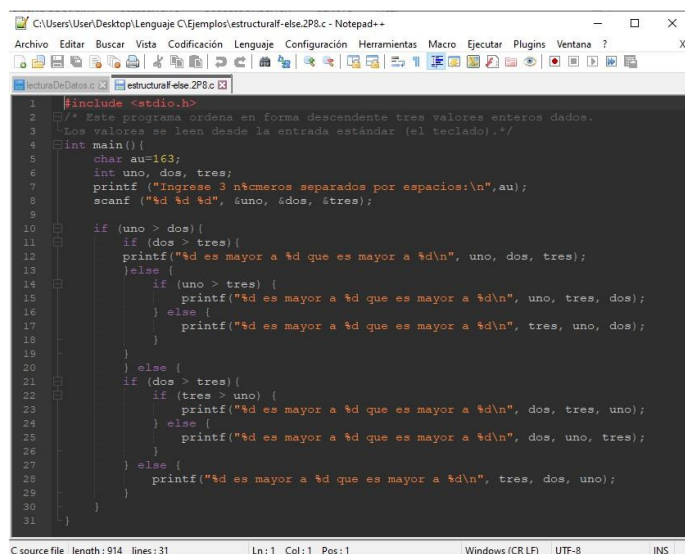
Luego de esto vimos cómo es que podemos escanear o escribir datos desde el teclado para lo cual se ocupa la función `"scanf"`, para ello también es necesario los distintos formatos de datos y la estructura para ponerlas es `"(%d",&x)"`, siendo x la variable en la cual se guardara, por lo que igual se el ampersand `"&"` funciona para que este valor quede registrado.

De igual manera vimos las estructuras de selección, la primera fue la selectiva, representada con `"if"`, que puede agregársele `"else"` (si y sino respectivamente); la estructura de esta es poner esa palabra primero y consecutivamente entre paréntesis sería la condición, donde se ocupan operadores lógicos. Los operadores lógicos son: `==` → Igual a

`!=` → Diferente de

`<`, `<=` Mayor que, Mayor o igual que

`>`, `>=` Menos que, Menos o igual que



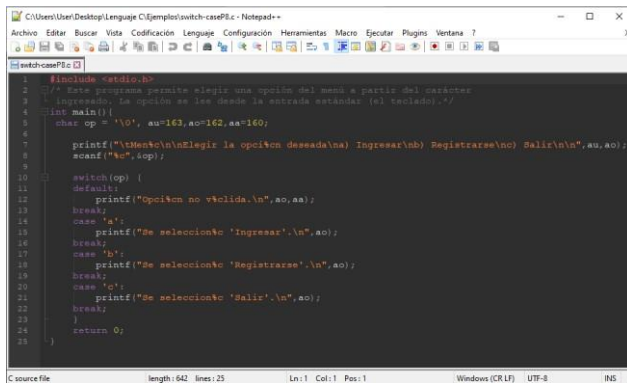
```
1 #include <stdio.h>
2 /* Este programa ordena en forma descendente tres valores enteros dados.
3  Los valores se leen desde la entrada estándar (el teclado). */
4 int main()
5 {
6     char au=163;
7     int uno, dos, tres;
8     printf ("Ingrese 3 números separados por espacios:\n",au);
9     scanf ("%d %d %d", &uno, &dos, &tres);
10
11     if (uno > dos){
12         if (dos > tres){
13             printf("%d es mayor a %d que es mayor a %d\n", uno, dos, tres);
14         } else {
15             if (uno > tres) {
16                 printf("%d es mayor a %d que es mayor a %d\n", uno, tres, dos);
17             } else {
18                 printf("%d es mayor a %d que es mayor a %d\n", tres, uno, dos);
19             }
20         }
21     } else {
22         if (dos > tres){
23             if (tres > uno) {
24                 printf("%d es mayor a %d que es mayor a %d\n", dos, tres, uno);
25             } else {
26                 printf("%d es mayor a %d que es mayor a %d\n", dos, uno, tres);
27             }
28         } else {
29             printf("%d es mayor a %d que es mayor a %d\n", tres, dos, uno);
30         }
31     }
32 }
```

&& → And (Y)

|| → Or (O)

! → Not (No)

Así como esa, está la estructura de control selectiva switch-case, en el cual es posible poner antes el que se vea el menú para que te muestre las distintas opciones, una vez puesto esto, entonces se pone primero "switch", seguido del

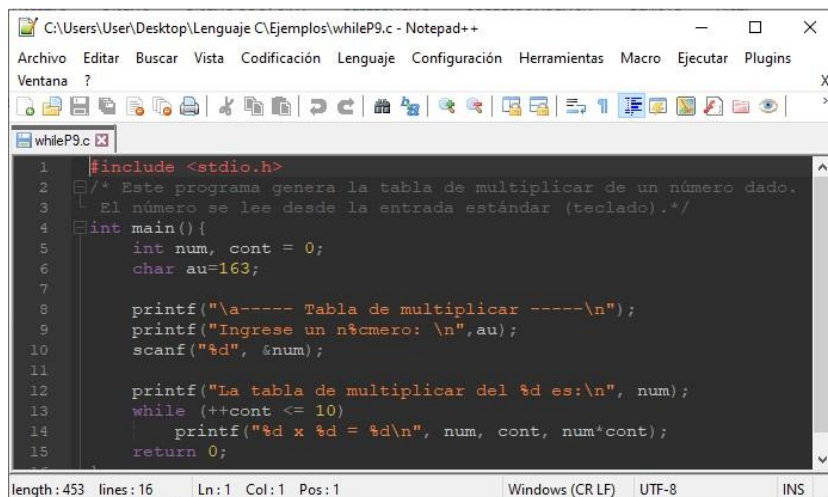


```
1 #include <stdio.h>
2 /* Este programa permite elegir una opción del menú a partir del carácter
3 ingresado. La opción se lee desde la entrada estándar (el teclado). */
4 int main()
5 {
6     char op = '0', au=163, ao=162, aa=160;
7
8     printf("\tMenú\nElegir la opción deseada\nIngresar\nRegistrar\nSalir\n\n", au, ao);
9     scanf("%c", &op);
10
11     switch(op) {
12         default:
13             printf("Opción no válida.\n", ao, aa);
14             break;
15         case 'a':
16             printf("Se seleccionó 'Ingresar'.\n", ao);
17             break;
18         case 'b':
19             printf("Se seleccionó 'Registrar'.\n", ao);
20             break;
21         case 'c':
22             printf("Se seleccionó 'Salir'.\n", ao);
23             break;
24     }
25     return 0;
26 }
```

nombre de la variable que se escaneo, así es como después de enuncian los distintos casos con "case" y el número, luego de ello las acciones que se realizarán en este, finalmente aquí debes de agregar también la opción "default", indispensable para que no se tome en cuenta una de las

opciones que estén ahí.

Otra sería estructura de control repetitiva, la cual es con la palabra "do" seguido de las instrucciones que seguirá y "while" una vez acabadas estas, acatando al hecho de que realizaran una y otra vez hasta que cumplan la condición que se enuncie seguida de la palabra de while, aunque también si se pone primero la palabra "while" con la condición, se repetirá las acciones que se pongan después de esta hasta que se cumpla esta.



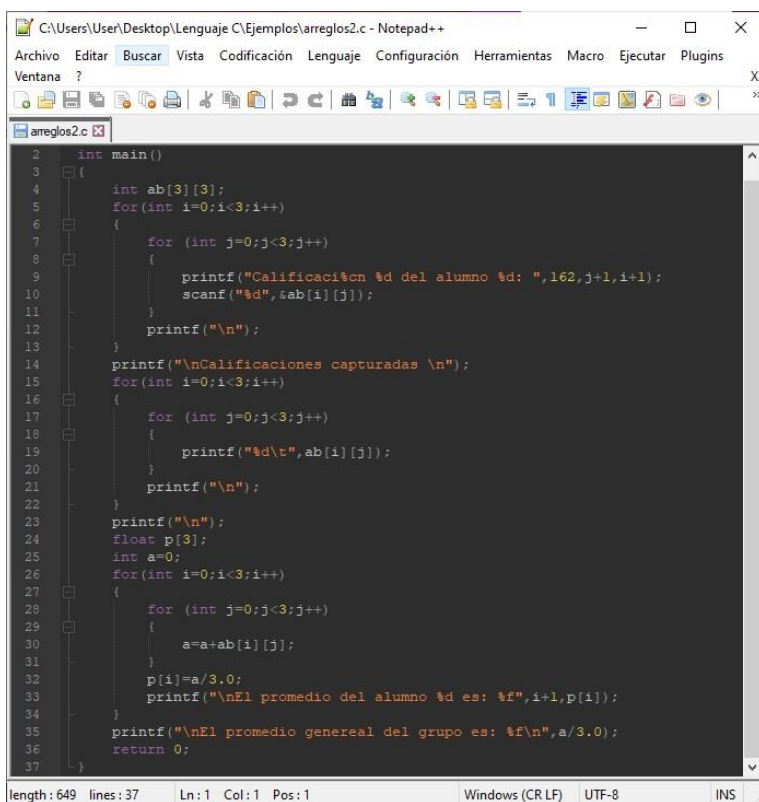
```
1 #include <stdio.h>
2 /* Este programa genera la tabla de multiplicar de un número dado.
3 El número se lee desde la entrada estándar (teclado). */
4 int main()
5 {
6     int num, cont = 0;
7     char au=163;
8
9     printf("\n----- Tabla de multiplicar ----- \n");
10    printf("Ingrese un número: \n", au);
11    scanf("%d", &num);
12
13    printf("La tabla de multiplicar del %d es:\n", num);
14    while (++cont <= 10)
15        printf("%d x %d = %d\n", num, cont, num*cont);
16    return 0;
17 }
```



También vi la estructura de control de repetición "for", este es un poco más preciso y tiene un contador, para ello entre paréntesis inicializas el valor de tu primera variable, también es posible declararla desde ahí, luego con punto y coma, se pone la condición con respecto a un valor numérico o el valor ya puesto en una variable, finalmente se pone a la variable junto con "++", esto siendo para que se vaya sumando el valor de esa variable de uno en uno.

```
for (int indice = 0 ; indice < enteroNumAlumnos ; indice++)
```

Algo más que vimos fue el depurador, cosa que no me encanto, ya que era necesario interpretar lo que ocurría, cosa que se me dificultaba el ver los errores que se presentaban en los programas que nos daban, así como se me facilita más ver cuando tienen error en la principal al intentar compilar, ya que te señalan igual el número de renglón en donde se presenta.



```
1  // Ejemplo de arreglos multidimensionales
2  int main()
3  {
4      int ab[3][3];
5      for(int i=0;i<3;i++)
6      {
7          for (int j=0;j<3;j++)
8          {
9              printf("Calificaci%cn %d del alumno %d: ",i+1,j+1,i+1);
10             scanf("%d",&ab[i][j]);
11         }
12         printf("\n");
13     }
14     printf("\nCalificaciones capturadas \n");
15     for(int i=0;i<3;i++)
16     {
17         for (int j=0;j<3;j++)
18         {
19             printf("%d\t",ab[i][j]);
20         }
21         printf("\n");
22     }
23     printf("\n");
24     float p[3];
25     int a=0;
26     for(int i=0;i<3;i++)
27     {
28         for (int j=0;j<3;j++)
29         {
30             a=a+ab[i][j];
31         }
32         p[i]=a/3.0;
33         printf("\nEl promedio del alumno %d es: %f",i+1,p[i]);
34     }
35     printf("\nEl promedio genereal del grupo es: %f\n",a/3.0);
36     return 0;
37 }
```

Conocí los arreglos, para los cuales se me facilitaron los unidimensionales, siendo básicos, pero los multidimensionales tenían una mayor dificultad, más en el aspecto de incluir los for para que se pudiera repetir la acción y guardar los datos en el arreglo respectivo, aunque siendo de mucha ayuda para crear un inventario o para capturar muchos datos. Este conocimiento también lo aplique al querer traducir código morse a palabras comunes del habla en español.

De igual forma con el

proyecto final fuimos viendo cómo es que cada detalle y cada conocimiento que se tiene te ayudan al momento de tener un empleo que te paguen más, también ver por nuestra cuenta aproximadamente cómo cuanto valdría nuestro proyecto, siendo el tiempo que le invertiste, que tan complicado es y lo estructurado.

### **Referencias:**

Diagrama de flujo. (s. f.). [Ilustración]. <https://bit.ly/3kkI7La>

Manual de prácticas del laboratorio de Fundamentos de programación. Facultad de Ingeniería UNAM. Recuperado de: <http://lcp02.fib.unam.mx/poll/login/>