

Universidad Nacional Autónoma de México

Facultad de Ingeniería

Estructuras de Datos y Algoritmos I

Actividad #1 | Repaso

Ramírez Pérez Daniela Itzel

24/Febrero/2021

## Repaso

Para comenzar, los temas más destacados que vimos el semestre pasado fueron los siguientes:




- **Solución de problemas y Algoritmos**

En este se comenzó los análisis de problemas, y se nos dieron ejemplos de sus soluciones en forma de algoritmos, con los algoritmos siendo descritos como una serie de instrucciones con la finalidad de resolver el problema que se nos proporcionaba o para realizar una serie de acciones para cumplir con un objetivo.



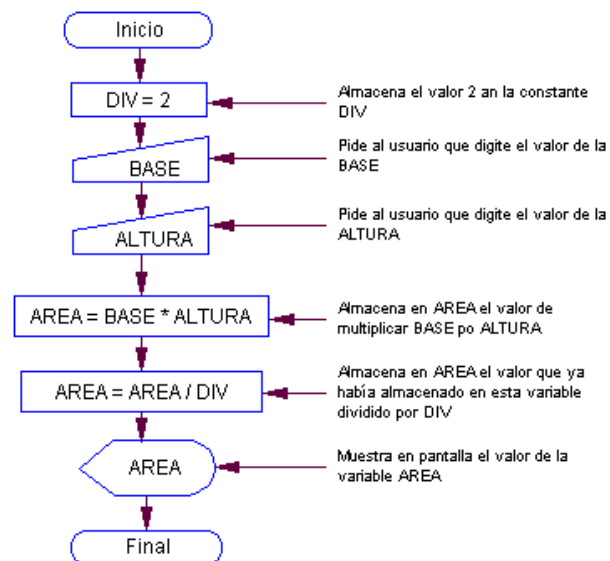
- **Diagramas de flujo**

Se crearon una serie de gráficos para representar de forma visual como es la función de un algoritmo, con el fin de cumplirse la tarea dada.

En el tema se muestran diferentes elementos que representan diferentes acciones del lenguaje gráfico, tal como el inicio y fin   o datos de entrada . También se dan a conocer varias reglas para poder construir un diagrama de flujo, tales como:

El diagrama debe ser construido de arriba hacia abajo y de izquierda a derecha, a cada gráfico solo le puede llegar una línea de dirección de flujo.

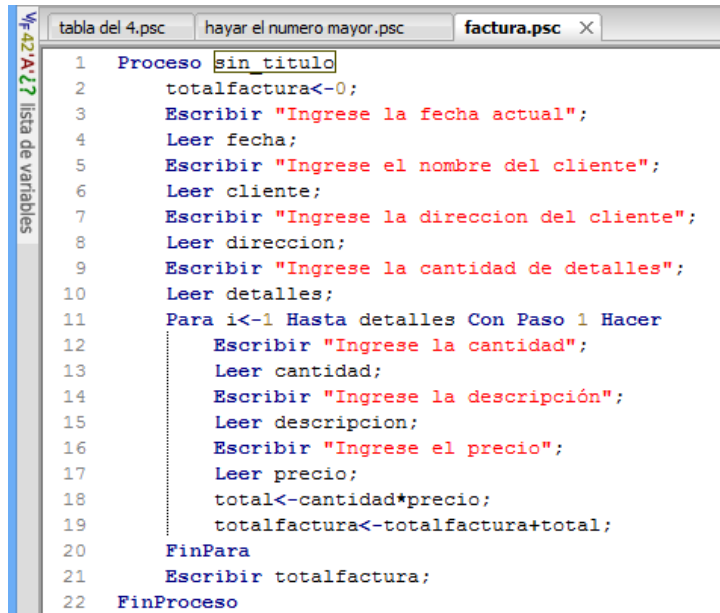
Existen diferentes estructuras con la función de permitir la ejecución condicional y la repetición de instrucciones con las funciones ayudando a dividir las soluciones de problemas en varios subprocesos.



- **Pseudocódigo**

El pseudocódigo se crea a partir del algoritmo, con este siendo su representación escrita con sus características siendo de siempre indicar el inicio y fin del programa, se debe anotar claramente las instrucciones con el uso de sangría o tabulación y se debe indicar donde se leen o escriben los datos.

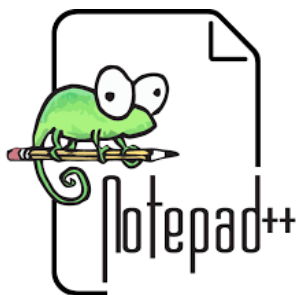
Como el tema es similar a los diagramas de flujo, al ambos ser representaciones del algoritmo, se hace referencia uno con el otro para crearse el pseudocódigo.



```
1 Proceso sin titulo
2 totalfactura<-0;
3 Escribir "Ingrese la fecha actual";
4 Leer fecha;
5 Escribir "Ingrese el nombre del cliente";
6 Leer cliente;
7 Escribir "Ingrese la direccion del cliente";
8 Leer direccion;
9 Escribir "Ingrese la cantidad de detalles";
10 Leer detalles;
11 Para i<-1 Hasta detalles Con Paso 1 Hacer
12     Escribir "Ingrese la cantidad";
13     Leer cantidad;
14     Escribir "Ingrese la descripción";
15     Leer descripcion;
16     Escribir "Ingrese el precio";
17     Leer precio;
18     total<-cantidad*precio;
19     totalfactura<-totalfactura+total;
20 FinPara
21 Escribir totalfactura;
22 FinProceso
```

- **Entorno de C (editores, compilación y ejecución)**

Con este tema se nos enseñó los diferentes programas que podemos utilizar para escribir, compilar y ejecutar nuestro código, en este caso utilizamos el lenguaje C y en este caso uno recomendado fue Notepad++, en el cual se podía escribir y se utilizaba GCC para compilar y ejecutar en el mismo programa de la computadora



- **Fundamentos de Lenguaje C**

En este tema empezamos ya codificar el algoritmo con el lenguaje tipo C, empezando con un editor para crear el código fuente, se crea el programa ejecutable y al final se ejecuta dicho programa con las instrucciones que han sido definidas.



También se incluye en el tema la creación de comentarios de nuestro código para describir lo que se realiza en cada bloque de código.

Por otra parte, está la declaración de variables donde se utilizan para identificar y manejar los valores que se les asigna, con diferentes tipos de datos siendo posibles para declarar dependiendo de que problema se necesita solucionar.

Otro aspecto son las expresiones lógicas para cuando estamos declarando condiciones donde los únicos valores son verdadero o falso, y dependiendo del resultado se puede modificar las variables con la ayuda de operadores que se nos dan en el lenguaje C.

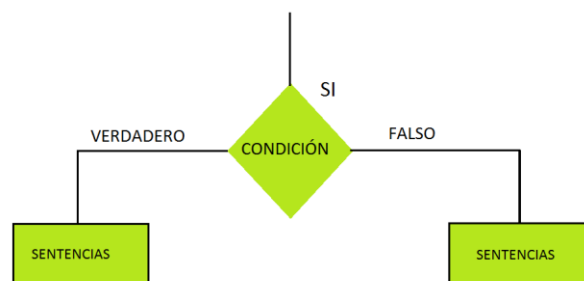
```
1  ' Globales -----
2  Var Variable0:Booleano
3  Var Variable1:Cadena
4  ' Fin Globales -----
5  Proc Procedimiento ' <- Procedimiento sin retorno.
6  Var Variable2:Entero ' Locales
7  Var Variable3:Real
8
9  Si Variable0 = Falso Entonces ' Condición "If"
10  Contar Variable2 = 0 a 9 ' Bucle "For"
11  Variable1 = Variable1 + "1"
12  Seguir ' "End For"
13  FinSi ' "End If"
14
15  Variable3 = 5.13
16  FinProc
```

## • Estructuras de selección

Con este tipo de estructuras se declara el orden como las expresiones lógicas dependerán de que se realicen diferentes acciones para que ejecute la estructura escrita.

Con las 3 estructuras siendo:

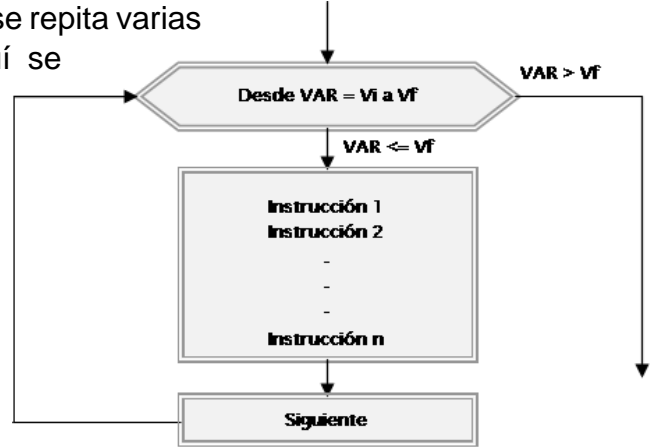
- **if-else:** Solo se ejecuta lo que se encuentran entre las llaves si la condición es verdadera, sino se continua normalmente el programa, con “else” se ejecuta al ser falsa y sigue el flujo normal del programa.
- **switch:** De esta estructura se compara los valores que se encuentra dentro de cada caso declarado y se ejecuta lo que coincida con el valor que se le da, si ninguno coincide se ejecuta “default” usualmente encontrada al final de la estructura.
- **Condicional:** Se da una condición que se debe cumplir y se dan dos posibles acciones siguientes dependiendo si la condición es verdadera o falsa



## • Estructuras de repetición

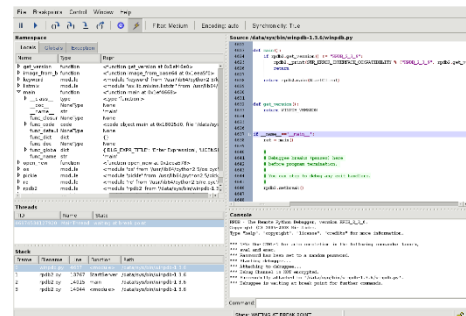
Este tipo de estructuras permite que un programa se repita varias cuando la expresión se declare verdadera, aquí se utilizan 3 tipos de estructuras siendo:

- **While:** esta estructura ejecuta el bloque solamente cuando la expresión lógica es verdadera.
- **do-while:** esta estructura ejecuta el bloque de código que se encuentre entre las llaves {} una vez y después checará si la expresión lógica en "while" es verdadera.
- **for:** esta es la estructura en donde se definen variables con sus respectivos valores si se requiere, se valida la expresión lógica y finalmente si es verdadera se ejecuta el bloque de código y se realiza la operación declarada al principio de la estructura cada vez que se repita



## • Depuración de programas

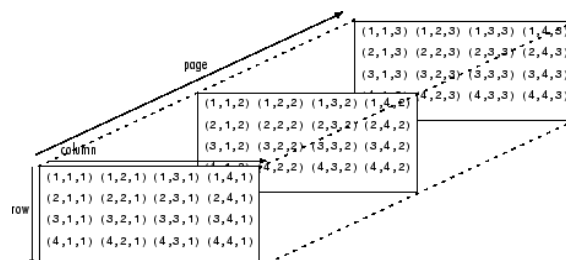
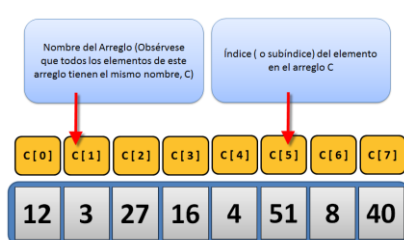
La depuración de programas es un proceso por cual se evalúa un programa en C para observar exactamente como va cambiando el valor que se dio para corregir errores que encuentren dentro programa.



el  
le  
del

## • Arreglos unidimensionales y multidimensionales

Los arreglos vistos en el tema sirven para acceder a una variable en específico por medio de índices con la ayuda de apuntadores, estos mismo siendo utilizados para crear una variable que contiene la dirección de la variable deseada, declarándolo con el carácter \*.



## • Variables

Lo más importante que se puede aprender son los tipos de datos que se pueden almacenar en la memoria de la computadora y para cuando estamos asignándole valores a estos mismos.

Para ejemplificar se tiene la siguiente tabla de los tipos de datos.

Tipos de Datos	Memoria que ocupa	Rango de valores
boolean	1 byte	0 o 1 (True o False)
byte / unsigned char	1 byte	0 – 255
char	1 byte	-128 – 127
int	2 bytes	-32.768 – 32.767
word / unsigned int	2 bytes	0 – 65.535
long	2 bytes	-2.147.483.648 – 2.147.483.647
unsigned long	4 bytes	0 – 4.294.967.295
float / double	4 bytes	-3,4028235E+38 – 3,4028235E+38
string	1 byte + x	Array de caracteres
array	1 byte + x	Colección de variables

## • ASCII Code

Una herramienta conveniente que se me hizo saber es la tabla de ASCII que contiene todos los códigos HEX para poner caracteres “especiales” que se deseen, debido a que ellos no son reconocidos automáticamente por el compilador que se utiliza.

### The ASCII code

American Standard Code for Information Interchange

ASCII control characters		ASCII printable characters			Extended ASCII characters				
DEC	HEX	Símbolo	ASCII	DEC	HEX	Símbolo	DEC	HEX	Símbolo
00	00h	NULL	(carácter nulo)	32	20h	espacio	64	40h	@
01	01h	SOH	(inicio encabezado)	33	21h	!	65	41h	A
02	02h	STX	(inicio texto)	34	22h	"	66	42h	B
03	03h	ETX	(fin de texto)	35	23h	#	67	43h	C
04	04h	EOF	(fin transmisión)	36	24h	\$	68	44h	D
05	05h	ENQ	(enquiry)	37	25h	%	69	45h	E
06	06h	ACK	(acknowledgement)	38	26h	&	70	46h	F
07	07h	BEL	(bél)	39	27h	'	71	47h	G
08	08h	BS	(retroceso)	40	28h	(	72	48h	H
09	09h	HT	(tab horizontal)	41	29h	)	73	49h	I
10	0Ah	LF	(salto de línea)	42	2Ah	*	74	4Ah	J
11	0Bh	VT	(tab vertical)	43	2Bh	+	75	4Bh	K
12	0Ch	FF	(form feed)	44	2Ch	,	76	4Ch	L
13	0Dh	CR	(retorno de carro)	45	2Dh	-	77	4Dh	M
14	0Eh	SO	(shift Out)	46	2Eh	.	78	4Eh	N
15	0Fh	SI	(shift In)	47	2Fh	/	79	4Fh	O
16	10h	DLE	(data link escape)	48	30h	0	80	50h	P
17	11h	DC1	(device control 1)	49	31h	1	81	51h	Q
18	12h	DC2	(device control 2)	50	32h	2	82	52h	R
19	13h	DC3	(device control 3)	51	33h	3	83	53h	S
20	14h	DC4	(device control 4)	52	34h	4	84	54h	T
21	15h	NAK	(negative acknowledge)	53	35h	5	85	55h	U
22	16h	SYN	(synchronous idle)	54	36h	6	86	56h	V
23	17h	ETB	(end of trans. block)	55	37h	7	87	57h	W
24	18h	CAN	(cancel)	56	38h	8	88	58h	X
25	19h	EM	(end of medium)	57	39h	9	89	59h	Y
26	1Ah	SUB	(substitute)	58	3Ah	:	90	5Ah	Z
27	1Bh	ESC	(escape)	59	3Bh	;	91	5Bh	[
28	1Ch	FS	(file separator)	60	3Ch	<	92	5Ch	\
29	1Dh	GS	(group separator)	61	3Dh	=	93	5Dh	]
30	1Eh	RS	(record separator)	62	3Eh	>	94	5Eh	^
31	1Fh	US	(unit separator)	63	3Fh	?	95	5Fh	_
127	7Fh	DEL	(delete)						

## Referencias

- Nakayama Cervantes, A., Castañeda Perdomo, M., Solano Gálvez, J. A., García Cano, E. E., Sandoval Montaña, L., & Arteaga Ricci, T. I. (2018, 6 abril). Manual de prácticas del laboratorio de Fundamentos de programación. Laboratorio de Computación Salas A y B. [http://odin.fib.unam.mx/salac/practicasp/MADO-17\\_FP.pdf](http://odin.fib.unam.mx/salac/practicasp/MADO-17_FP.pdf)