

BUAP[®]



VIRTUAL
BUAP®

Metodología de la Programación

UNIDAD 1. Diseño estructurado usando diagramas de flujo y pseudocódigo

Variables computacionales

- **Variable:** Área de **almacenamiento**, de **tamaño fijo** que sirve para guardar un tipo de dato específico y puede cambiar su valor.
- Tienen dos elementos: El **Nombre** de la variable y el **Contenido** de la variable.
- La variable puede almacenar un sólo tipo de dato como puede ser datos enteros, reales, booleanos, caracteres, etc.



Constante

- **Constante:** Área de **almacenamiento**, de **tamaño fijo** que sirve para guardar un tipo de dato específico y su valor no cambia.
- Puede tener dos elementos: El **Nombre** de la variable y el **valor**.
- La constante puede almacenar un sólo tipo de dato como puede ser datos enteros, reales, booleanos, caracteres, etc.

Operaciones Aritmético/Lógicas

- Las **operaciones** Aritméticas y Lógicas que una computadora puede hacer están en **función** del tamaño y del propósito de dicha computadora.



Aritmético-lógico básicas

Operador	Símbolo
Paréntesis	()
Incremento	++
Decremento	--
Suma	+
Resta	-
Multiplicación	* / MOD, DIV
AND	AND
OR	OR
Negación	NOT
Potencia	^

Prioridad	Asociatividad
1	Izq. A Der.
2	Der. A izq.
2	Der. A izq.
5	Izq. A der.
5	Izq. A Der.
4	Izq. A Der.
8	Izq. A der.
8	Izq. A Der.
3	Der. A izq.
2	Izq. A Der.

Lógicos

D1	D2	AND	OR	NOT D2
V	V	1	1	0
V	F	0	1	1
F	V	0	1	0
F	F	0	0	1

- - toma un operando numérico y devuelve 1 para un operando cero y 0 en caso contrario.
- O devuelve 1 si cualquiera de los dos operandos es no cero, y 0 en otro caso
- Y devuelve 0 si cualquiera de los 2 es 0 y 1 en caso contrario

número diferente de 0 es V y 0 es F

Ejemplos

- $8 \text{ AND } 4 = 1$
- $0 \text{ AND } 3 = 0$
- $0 \text{ OR } 0 = 0$
- $0 \text{ OR } 3 = 1$
- $8 \text{ MOD } 4 = 0$
- $7 \text{ MOD } 3 = 1$

Relacionales

Operador	Símbolos	Prioridad	Asociatividad
Igual que	=	7	Izq. A Der.
Menor que	<	6	Izq. A Der.
Mayor que	>	6	Izq. A Der.
Menor o igual que	<=	6	Izq. A Der.
Mayor o igual que	>=	6	Izq. A Der.
Distinto que	<>	7	Izq. A Der.

Expresiones

- Es una combinación de operandos (variables, constantes, llamado a módulos y expresiones) y operadores.
- Las expresiones se pueden dividir en expresiones aritméticas o lógicas.

Ejemplos de expresiones

- Aritméticas:

- $5^2 * 3$
- $(50 * 2) + 10$
- $\text{Salario} * 0.15$

- Lógicas

- $10 < 5$ AND $8 < -3$ $8 <> 5$

Simples: solo 1 operador relacional o lógico

Compuestas: 2 o mas operadores relacionales o lógicos

Orden de ejecución

- Operadores unarios de derecha a izquierda
- Operadores de multiplicación, división y modulo de izquierda a derecha
- Operadores de suma y resta de izquierda a derecha
- Operadores de relación de izquierda a derecha
- Operadores de asignación

Nota. Los paréntesis son para fijar un orden o hacer más clara la expresión

Sentencias

- Son estructuras formadas por una variable, seguida de un símbolo de asignación ← (también llamado operador de asignación) y finalmente una expresión.

Ejemplos

Base $\leftarrow (500 * 2) + 100$

Salario \leftarrow DiasTrabajados * SalarioDiario

Impuesto \leftarrow Salario * 0.15

a \rightarrow 2 b \rightarrow 1 c \rightarrow 3

Mayor \leftarrow a > b

Menor \leftarrow a < b AND b < c

(a < b) AND (b < c)

(2 < 1) AND (1 < 3)

(0) AND (1)

Menor \leftarrow 0

**Evaluando
la expresión**

Ejercicios

Encontrar y evaluar la expresión equivalente

$$a \leftarrow d \leftarrow e \leftarrow 2 \quad b \leftarrow 2 \quad c \leftarrow y \leftarrow 4 \quad f \leftarrow 3$$

$$1. \quad \underline{\underline{((2 / b) - ((c * a) * y)) + (18 / c) - (3^6 * d) = -33}}$$

$$2. \quad a * b + c + c * d + e$$

$$3. \quad a * b + c - 2 * a + 4 * d - f$$

$$4. \quad a \text{ MOD } b + c \text{ DIV } d - 6$$

$$5. \quad a + 2 * 3 + b$$

$$6. \quad 3 * a \text{ MOD } b / c + 5$$

$$7. \quad 6 * 5 ^ 10 * 2 + 10$$

Encontrar y evaluar la expresión equivalente

8. $6 * 5 \text{ DIV } 10 * 2 + 10 ^ \wedge 3 / - 3$

9. $6 / 5 - 10 ^ \wedge 2 + 10 * \text{NOT } 3$

10. $6 * 5 \text{ MOD } 10 * 2 + 10 \quad (((6*5)\text{MOD } 10)*2)+10= 10$

11. $a * b / c * d$

30

10

0

2

12. $27 \text{ MOD } 4 + 15 / 4$

0

13. $37 / 4 + 4 + \text{NOT } 7$

14. $9 + 2 / 3 * 25 * 3$

15. $7 * 3 - 4 - 45$

Encontrar y evaluar la expresión equivalente

16. $27 \text{ MOD } 4 + 15 / 4 \leq 4 + \text{NOT } 7$

17. $9 + 2 / 3 > 3 * 25 * \text{NOT } 3$

18. $7 * 3 \text{ MOD } 45 - 45 \langle \rangle 2 / 3$

19. $'a' < 'z'$ véase el código *ascii*

20. $'J' = 'j'$ véase el código *ascii*

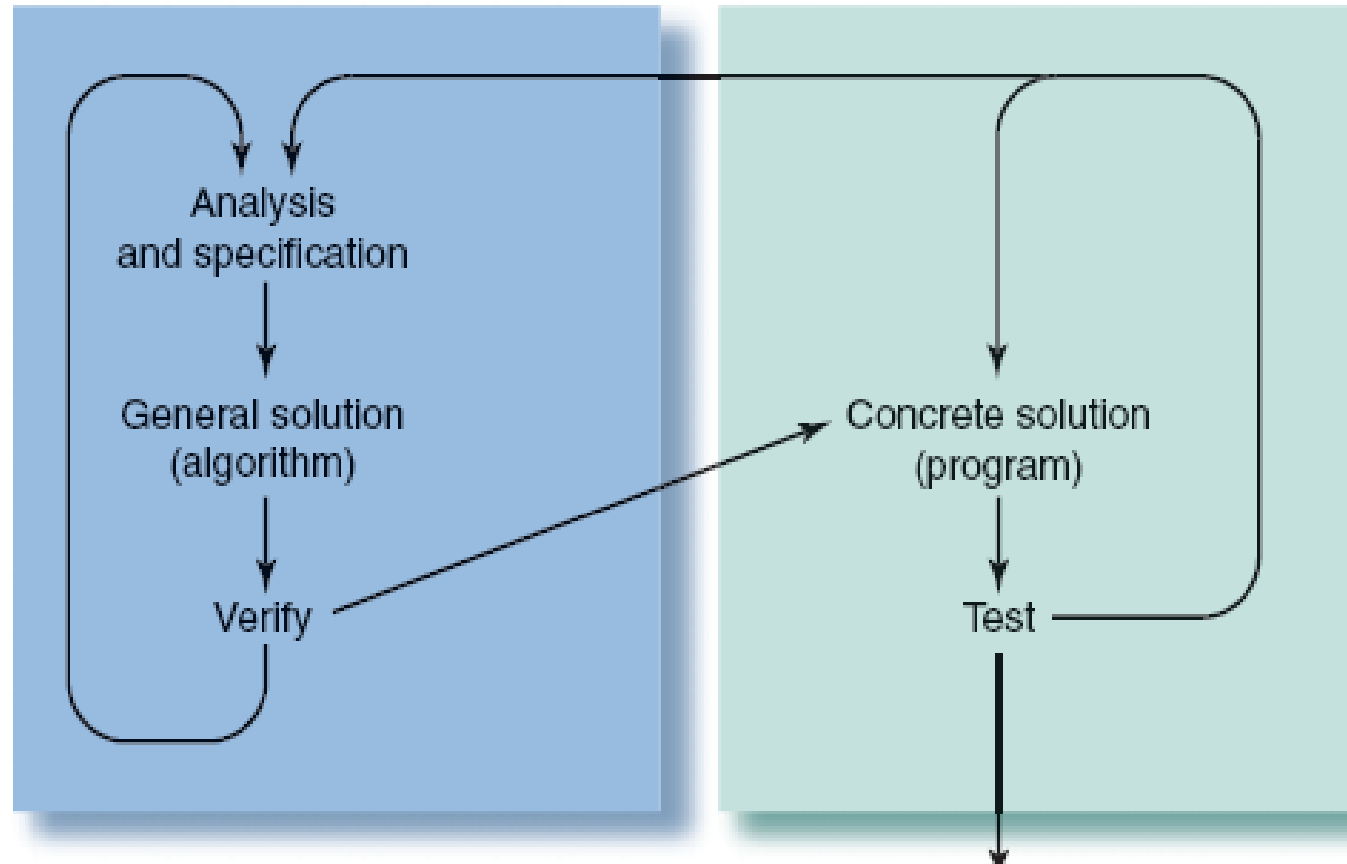
21. $24 < 5 \text{ AND } 10 \geq 10 \text{ OR } 10 = 5$

22. $-6 / 3 > 3 \text{ OR } 7 < 7 \text{ AND } 3 \geq 9 / 2 \text{ OR } 2 + 3 \leq 7$

PROCESO DE PROGRAMAR

Fase de RESOLUCIÓN de PROBLEMAS

Fase de IMPLEMENTACIÓN



Tomado de: Programming and Problem Solving with Java. Nell Dale Chap 1

Fase de MANTENIMIENTO

ALGORITMO

¿Qué es?

¿Qué características tiene?

¿Para que sirven?

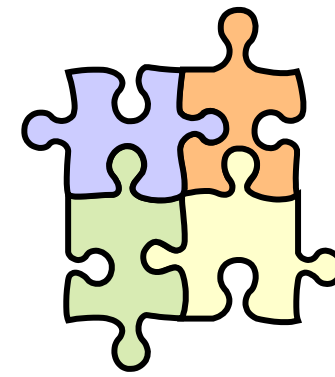
¿ QUE ES UN ALGORITMO?

- ◉ **Describe** el método mediante el cual se realiza una tarea.
- ◉ Conjunto finito de reglas que dan lugar a una secuencia de operaciones para resolver un tipo específico de problema.

CARACTERÍSTICAS

- ◉ **Finito**: Un algoritmo tiene que acabar siempre tras un **número finito** de pasos.
- ◉ **Definido**: Cada paso de un algoritmo debe definirse de modo preciso; las acciones a realizar han de estar especificadas **sin ambigüedad**
- ◉ **Preciso**: Orden de realización de cada uno de los pasos.

COMPONENTES



- ◉ **Entradas:** Debe existir un conjunto especificado de datos, cada uno de los cuales constituyen los datos iniciales.
- ◉ **Proceso:** Operaciones o cálculos (sentencias) necesarios para encontrar la solución del problema.
- ◉ **Salida:** salida o respuesta que debe obtener el algoritmo para los diferentes casos particulares del problema.

EJEMPLO 1

- ◉ Supongamos que desea desarrollar un algoritmo que calcule la superficie de un rectángulo proporcionándole su base y altura.

entrada

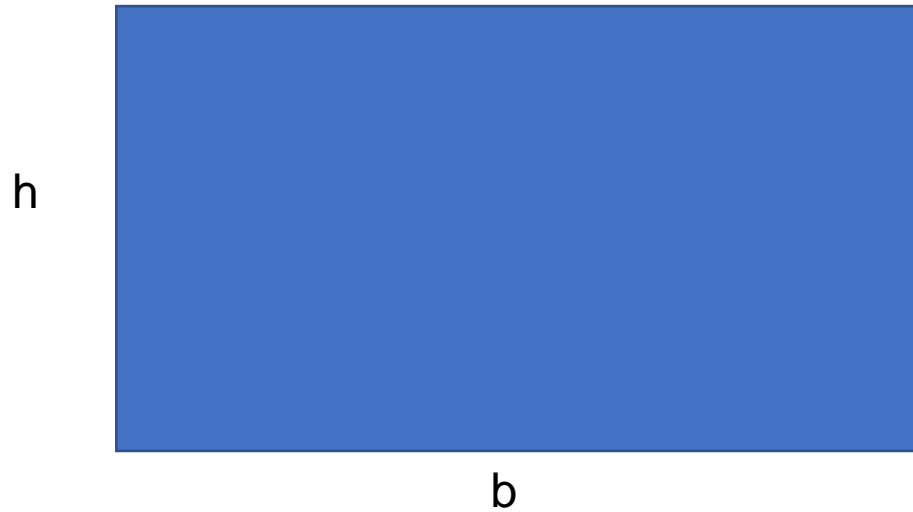
- ◉ Especificaciones de entrada
 - ¿Que datos son de entrada?

a y b



PROCESO

Conjunto de sentencias



area = base X altura

area=b X h

area \leftarrow b * h

SALIDA

- Especificaciones de salida
 - ¿Cuáles son los datos de salida?
 - ¿Qué formato y precisión tendrán los resultados?



ALGORITMO

Paso 1. Entrada de base y altura.

Paso 2. Cálculo de superficie, multiplicando la base por la altura.

Paso 3. Salida de base, altura y superficie.

- El lenguaje algorítmico debe ser independiente de cualquier lenguaje de programación particular.

- Métodos** normalizados para la representación de algoritmos: Pseudocódigo

- Diagrama de flujo



Algoritmo: Srie_del_1_al_100

Entrada: x

Salida: x

Inicio

$x \leftarrow 1$

mientras $(x < 101)$

escribir (x)

$x \leftarrow x + 1$

fin_mientras

finSrie_del_1_al_100

PRUEBA DEL ALGORITMO

- ◉ Comprobar que realiza las tareas para las que se ha diseñado y produce el resultado correcto y esperado.
- ◉ Mediante su ejecución manual, usando datos significativos que abarquen todo el posible rango de valores.



EJEMPLO 2

Algoritmo para determinar si un número es mayor, menor o igual a cero (descripción narrada)

Entrada: Un número de tipo entero

Proceso:

Inicio

Leer el número

Comparar el número con el cero

Si el número es > 0 escribir 'es positivo'

Si el número es < 0 escribir 'es negativo'

Si es $= 0$ escribir 'es cero'

Fin

Salida: 'Es positivo', 'Es negativo' o 'Es cero'

Ejercicios



En equipo de 4 resuelve

- ◉ Descripción del problema:
- ◉ Queremos hallar el MCD de dos números enteros positivos dados.
- ◉ Actividad:
 - Analiza el problema
 - Escribe los componentes del algoritmo y también el algoritmo en descripción narrada.

Ejemplo: 60 y 48 $60/48=1$ res=12 $48/12=4$ res=0



○ **Descripción:**

Se requiere escribir un conjunto de instrucciones que puedan emplearse para determinar si un año es bisiesto. Las instrucciones deben ser muy claras porque las utilizarán alumnos de cuarto grado que han aprendido la multiplicación y la división. Se planea usar las instrucciones como parte de una tarea para saber si alguno de sus parientes nació en un año bisiesto.

○ **Actividad:**

- Analicen el problema
- Escribe los componentes del algoritmo y también el algoritmo en descripción narrada.

Tarea2

Diseño estructurado

- Técnica para desarrollar algoritmos
 - Fáciles de escribir
 - Leer
 - Verificar
 - Modificar

Algoritmo estructurado

- Tiene un solo punto de E/S
- Toda acción del algoritmo es accesible
- No posee lazos infinitos

Puede ser escrito usando únicamente las estructuras secuencial, selectiva y repetitiva.

Representación de algoritmos

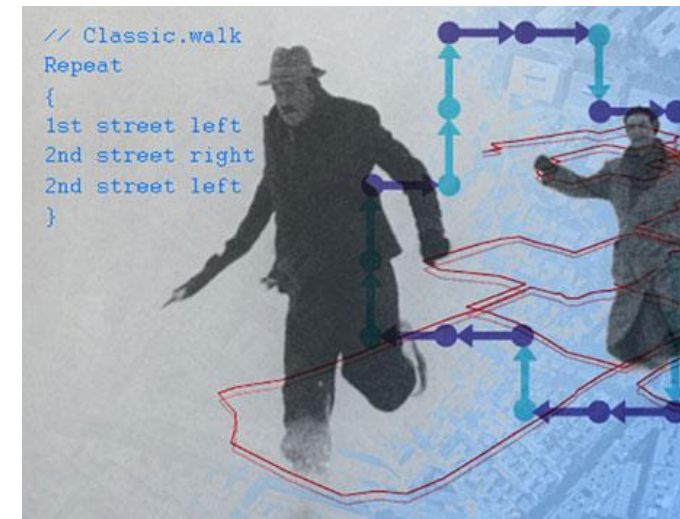

- Descripción narrada: Relato en lenguaje natural.
- Diagrama de flujo(D.F.): Representación gráfica.
- Pseudocodigo: Lenguaje que usa palabras reservadas e indentación.

EJEMPLO DE ALGORITMO

Cambiarle la rueda pinchada de un automóvil

ALGORITMO

- Aflojar tornillos con la llave inglesa
- Ubicar el gato mecánico
- Levantar el gato hasta que la rueda quede libre
- Quitar tornillos poner rueda de repuesto
- bajar el gato hasta que se pueda liberar




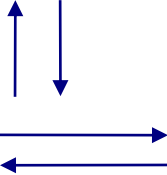






Diagramas de Flujo(DF)

Representación gráfica de un algoritmo

- Usa símbolos gráficos
- Forma sencilla y clara para el orden lógico de las acciones de un algoritmo.
- Símbolos normalizados por el Instituto Norteamericano de Normas (ANSI)

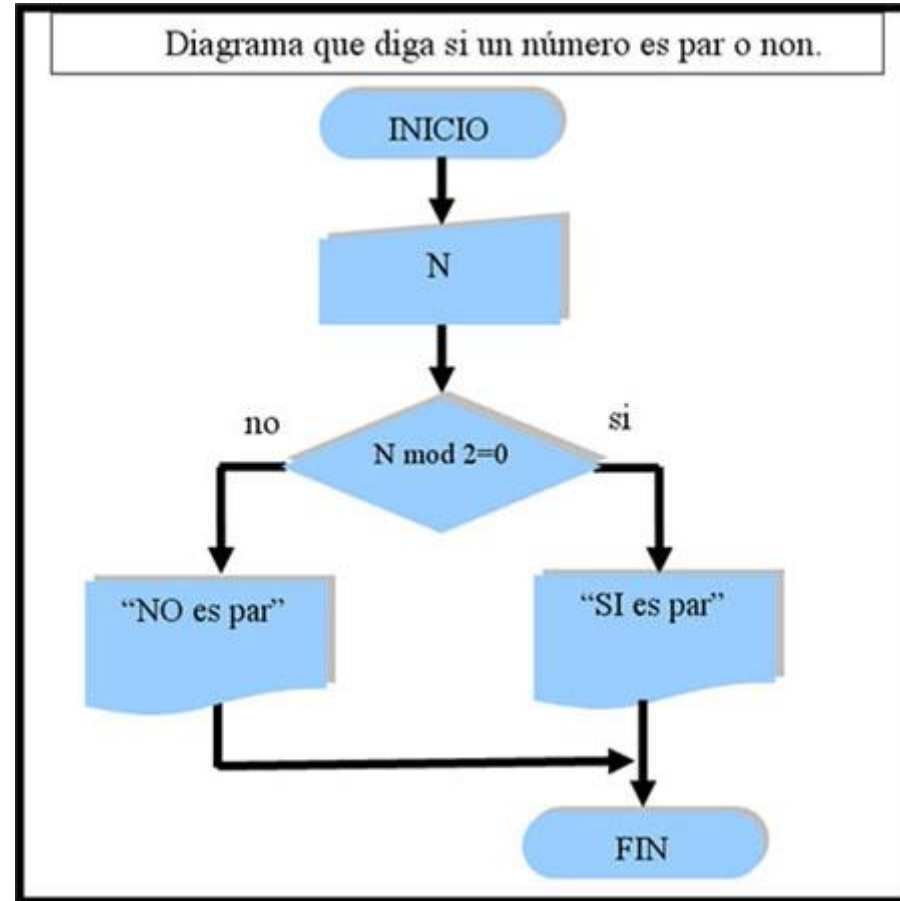
Notación

	Inicio / Fin	Indica el inicio o el fin del diagrama de flujo		módulo	Permite el llamado a otros diagramas de flujo
	Entrada	Permite la lectura o salida de datos (teclado o pantalla)		Flujo	Permite indicar la dirección del flujo del diagrama
	Salida				
	Decisión simple / doble repetición	Permite evaluar una condición y dependiendo del resultado se sigue por uno de los caminos alternativos		Conector	Conector para unir el flujo a otra parte del diagrama
	Decisión múltiple				

Reglas

- 1 Todo DF debe tener un *inicio* y un *fin*.
- 2 Las líneas del flujo del diagrama deben ser rectas, verticales y horizontales
- 3 Todas las líneas deben estar conectadas a alguno de los símbolos mostrados en la tabla.
- 4 El DF se construye de arriba hacia abajo y de izquierda a derecha.

Ejemplo



Pseudocódigo

Herramienta para escribir algoritmos

- Las instrucciones se escriben en palabras similares al inglés o español
- Facilitan tanto la escritura como la lectura de programas.
- Lenguaje de especificación de algoritmos

Palabras Reservadas

***Inicio, Fin, Leer, Escribir, Si,
Entonces, Sino, FinSi, Según,
FinSegun, Mientras, Hacer,
FinMientras, Repetir, HastaQue,
Para, FinPara, Desde, etc.***

Características

- Mantiene una indentación o sangría adecuada para facilitar la identificación de los elementos que lo componen.
- Permite la declaración de los datos (constantes y/o variables) manipulados por el algoritmo.
- Dispone de un conjunto pequeño de palabras reservadas (normalmente escritas con letra negrita).
- Supera las dos principales desventajas del diagrama de flujo: lento de crear y difícil de modificar.

Elementos de un Algoritmo en pseudocódigo

Cabecera: Algoritmo:nombre_algoritmo

Entrada: Lista de variables

Salida: Datos de salida

Cuerpo: Donde se realizan todas las acciones del algoritmo.

Estructura de un algoritmo

coment: Este es un comentario

Algoritmo: prueba

Entrada: SU 100

x,y

Salida: real

Inicio

2 Escribir (“Dame dos numero entero:“)

3 Leer(x,y)

4 $\text{real} \leftarrow x/y$

5 Escribir(“Resultado:“ ,real)

6 Finprueba

Tipos de datos estándar

- **Carácter:** Se declara con la palabra reservada **caracter**.
- **Real:** Se declara con la palabra reservada **real**.
- **Entero:** Se declara con la palabra reservada **entero**.

Ejemplo

Algoritmo: SumaProducto

Entrada: nota1, nota2

Salida: suma, producto

Inicio

Variables:

suma \leftarrow producto \leftarrow 0

Escribir ("Introduzca las notas")

Leer (nota1, nota2)

suma \leftarrow nota1 + nota2

producto \leftarrow nota1 * nota2

Escribir ("La suma de las dos notas es:", suma)

Escribir ("El producto de las dos notas es :", producto)

FinSumaProducto

Cabecera



Cuerpo



Modulo de salida (escribir)

Sintaxis:

Escribir(cadena de control, lista de argumentos)

Donde, cadena de control

Es una cadena delimitada por comillas (“ ”), formada por caracteres ordinarios.

lista de argumentos

representa el valor o valores a escribir en la pantalla.

Ejemplos

Ejemplo

Tiempo ← 8

Escribir("bienvenido a puebla son las ", tiempo)

modulo leer

Sintaxis:

Leer (*lista de argumentos*)

Donde, lista de argumentos

representa el valor o valores a escribir en la pantalla.

modulo de salida (leer)

Ejemplos:

Escribir("Dame un numero :")

Leer (n)

Escribir("El numero capturado es",n)

Bibliografía

1. Cairó O (2005). Metodología de la programación, Algoritmos, diagramas de flujo y programas (3ª ed). México: Alfaomega.
2. Joyanes, A. (2008). Fundamentos de programación, Algoritmos, Estructuras de datos y Objetos (4ª. ed). España: Mc Graw Hill.
3. Skiena, S (2008). The algorithm design Manual (2nd ed). USA: Springer.

Responsables del Curso

De la Rosa Flores Rafael

Moyao Martínez Yolanda

Sánchez Román Guillermina

Es responsabilidad exclusiva de los autores el respeto de los derechos de autor sobre los contenidos e imágenes en el presente documento, en consecuencia, la **BUAP** no se hace responsable por el uso no autorizado, errores, omisiones o manipulaciones de los derechos de autor y estos serán atribuidos directamente al **Responsable de Contenidos**, así como los efectos legales y éticos correspondientes.

gracias.

BUAP ©2020

Es responsabilidad exclusiva de los autores el respeto de los derechos de autor sobre los contenidos e imágenes en el presente documento, en consecuencia, la **BUAP** no se hace responsable por el uso no autorizado, errores, omisiones o manipulaciones de los derechos de autor y estos serán atribuidos directamente al **Responsable de Contenidos, así como los efectos legales y éticos correspondientes.**