

I.E.S. EL MAJUELO	
<i>Introducción a la programación</i>	<i>CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática</i>

Objetivos

Mapa conceptual

Glosario

1. Introducción
2. Orígenes de la programación
3. Paradigmas de programación
 - 3.1. Programación estructurada
 - 3.2. Programación modular
4. Pseudocódigo
 - 4.1. Operadores , palabras reservadas y tipos de datos
 - 4.2. Estructuras de control
5. Diagramas de flujo
 - 5.1. Simbología
 - 5.2. Estructuras de control

Resumen

Ejercicios

Supuestos

I.E.S. EL MAJUELO	
<i>Introducción a la programación</i>	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

1.- Introducción

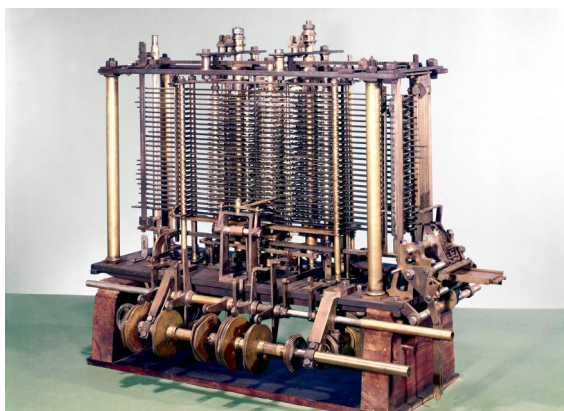
En este tema haremos una introducción a la programación desde sus orígenes hasta los tiempos actuales, presentando conceptos generales de programación y se abordarán los diferentes paradigmas de programación (programación modular y programación estructurada). Finalmente expondremos los diagramas de flujo y el pseudocódigo como técnicas de introducción a la programación.

2.- Orígenes de la programación

El término programación hace referencia al proceso mediante el cual se diseña y codifica un conjunto de instrucciones que implementan un determinado algoritmo.

La programación surgió ante la necesidad de automatizar determinadas tareas que se realizaban de forma manual. La programación tiene sus orígenes en las primeras calculadoras de operaciones matemáticas elementales, las cuales llevaban los algoritmos implementados en su estructura física, y en los telares automáticos , que recibían los datos sobre el trabajo a realizar a través de tarjetas perforadas.

No se puede hablar de programación como tal , hasta que apareció el diseño de la maquina analítica de Charles Babbage (esta máquina se considera como el primer ordenador de la historia) y la aportación posterior de Augusta Ada Byron (La primera programadora de la historia).



En las primeras computadoras las instrucciones de los programas eran escritas en código binario. Dada la complicación asociada a esta técnica de programación , los científicos que estudiaban esta nueva ciencia decidieron sustituir determinadas secuencias binarias por palabras que permitieron representar más fácilmente; este proceso dio lugar al conocido lenguaje ensamblador.

Conforme las necesidades fueron creciendo y se fue depurando la técnica aparecieron los lenguajes de alto nivel , que añadían una nueva capa de abstracción al proceso de programación y que permitían el desarrollo de programas cada vez más sofisticados y más fáciles de desarrollar.

I.E.S. EL MAJUELO	
Introducción a la programación	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

3.- Paradigmas de programación

Un paradigma de programación es un conjunto de reglas y costumbres que son aceptadas como modelo de referencia para generar un código de calidad.

Aunque cada paradigma de programación tiene unas características concretas, es cierto, que en la vida real , los programadores pueden basarse en varios de ellos al mismo tiempo para el desarrollo de sus programas. La inclusión de dos o más paradigmas de programación en el mismo programa se denomina “programación multiparadigma” y generalmente es la que mejores resultados obtiene.

Utilizar un paradigma de programación , aceptando y siguiendo sus reglas, derivará en un código fuente fácil de mantener , entender y corregir.

En este curso veremos el *paradigma de programación estructurada* , el *paradigma de programación modular* y el *paradigma de programación orientada a objetos*.

3.1.- Programación estructurada.

La programación estructurada se basa en el teorema del programa estructurado propuesto por Corrado Böhm y Giuseppe Jacopini, según el cual *todo programa se puede escribir empleando únicamente tres tipos de estructuras de control*.

1. **Secuencial:** las instrucciones se ejecutan una detrás de otra siguiendo un orden (secuencialidad).
2. **Alternativa:** las expresiones son evaluadas y, dependiendo del resultado, se decide cuál será la siguiente instrucción a ejecutar.
3. **Iterativa:** se repetirá un conjunto de instrucciones hasta que una condición sea cierta, permitiéndose de esta manera el salto a otra instrucción.

Los lenguajes de programación también ofrecen instrucciones de salto incondicional (go to) , pero su uso está totalmente desaconsejado porque complica el seguimiento y trazabilidad de los programas , generando un código espagueti más difícil de modificar y corregir. El por lo que este tipo de estructuras deben evitarse en la medida de lo posible.

Hay que reseñar que la programación estructurada presenta grandes dificultades a la hora de abordar proyectos de gran tamaño , para los cuales es más eficiente el empleo de la programación orientada a objetos.

En la programación estructurada los datos y las funciones que los manejan se definen separados.

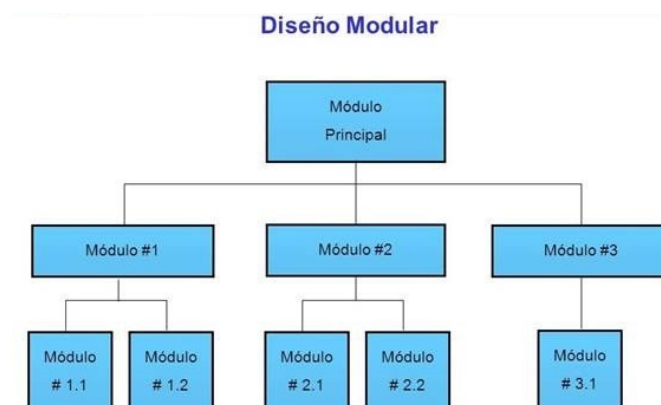
La programación estructurada no ofrece garantías para abordar proyectos de gran tamaño, en cuyo caso es más recomendable emplear programación orientada a objetos.

3.2.- Programación modular

La programación modular propone como solución la descomposición del problema en subproblemas de menor tamaño, los cuales serán descompuestos sucesivamente en otros de menor tamaño hasta que el resultado de dichas descomposiciones permita obtener problemas fáciles de resolver. A esta técnica de

I.E.S. EL MAJUELO	
Introducción a la programación	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

resolución de problemas se le denomina “divide y vencerás” y permite poner en practica un proceso de desarrollo software top-down (de arriba a abajo)



Cada bloque de código que se ocupará de resolver un problema de menor tamaño se denomina módulo o subprograma, y estará compuesto por un conjunto de sentencias físicamente unidas y delimitadas. Dicho módulo será referenciado por un nombre y podrá hacer uso de otros módulos. La comunicación entre módulos se realizará a través de interfaces de comunicación claramente definidas.

Los elementos empleados para llevar a cabo la **modularización** puede ser muy distintos dependiendo del nivel de abstracción:

1. *A un alto nivel serán las unidades, en diseños estructurados, y los paquetes y librerías, en diseños orientados a objetos.*
2. *A más bajo nivel serán los procedimientos y las funciones , en diseños estructurados, y las clases con sus correspondientes métodos, en diseños orientados a objetos.*

Para llevar a cabo una correcta modularización del programa, es necesario realizar un diseño de calidad que tenga en cuenta los siguientes criterios:

- El módulo debe de tener un único punto de entrada y un único punto de salida.
- Cada módulo debe realizar una única función bien definida.
- Cada módulo debe comportarse como una caja negra, de manera que dependa únicamente de las entradas.
- Módulos trazables en una pantalla (tamaño ideal 20-50 líneas de código por módulo)
- Máxima cohesión y mínimo acoplamiento. Ambos criterios están relacionados entre si de manera que, a mayor cohesión en los módulos , menor será el acoplamiento entre ellos.

I.E.S. EL MAJUELO	
Introducción a la programación	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

A) Cohesión

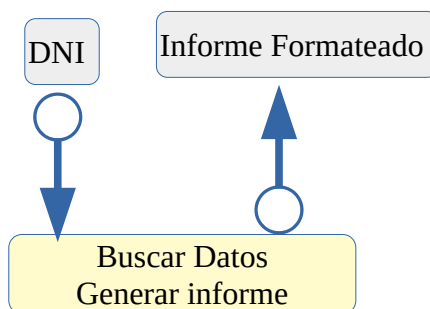
Se denomina *cohesión* de un módulo al nivel de relación existente entre los elementos software contenidos en él, entendiéndose como elementos software las instrucciones, definiciones de datos o las llamadas a otros módulos.

Los diferentes tipos de cohesión son:

- I. **Cohesión funcional:** los elementos del módulo contribuyen a la realización de una única tarea. Es decir, cada elemento es una parte integral de la estructura del módulo. El módulo se representa mediante un identificador simple. Por ejemplo: módulos matemáticos suma, seno, potencia,
- II. **Cohesión secuencial:** se da cuando el módulo realiza varias tareas según una secuencia que establece que la salida de una actividad es la entrada necesaria de la siguiente.

Ejemplo:

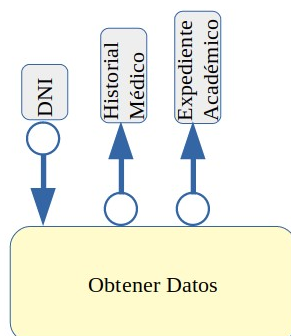
Progresivamente el módulo se ocupará de buscar información a partir de un dato de entrada y, con ella, generar un informe formateado que devolverá como dato de salida. Podrían encadenarse más acciones.



- III. **Cohesión comunicacional:** contiene actividades paralelas (sin orden) que comparten los mismos datos (de entrada o salida). Se recomienda su descomposición en módulos independientes de cohesión funcional.

Ejemplo:

El módulo se ocupará de buscar información a partir de un dato de entrada y devolverá como salida información de diverso tipo. El orden de las actividades no es relevante.

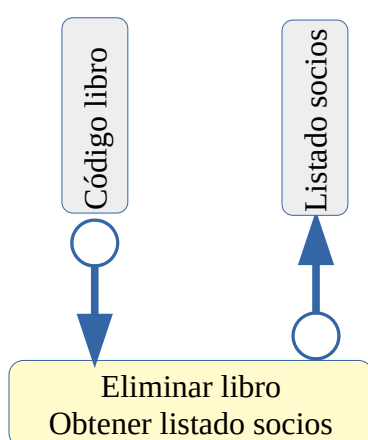


I.E.S. EL MAJUELO	
Introducción a la programación	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

IV. Cohesión procedural: los elementos realizan diferentes actividades posiblemente no relacionadas entre si. También es posible que no exista relación alguna entre los datos de entrada y de la salida de los módulos. El control fluye de una actividad a la siguiente.

Ejemplo

El módulo eliminará del sistema los datos relativos al libro cuyo código recibe como dato de entrada; posteriormente retornará como datos de salida un listado de los socios de la biblioteca



V. Cohesión temporal: los elementos están implicados en la realización de actividades relacionadas por el momento en el cual se llevan a cabo.

Ejemplo

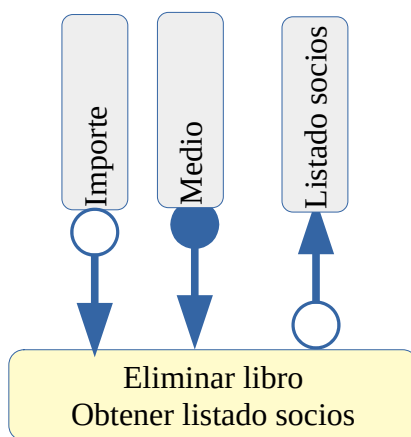
Un ejemplo típico son los módulos de inicialización y finalización de sistema (arranca sistema, apagar sistema, resetear sistema, inicializar sistema,). Es posible que no manejen ningún dato de entrada o salida.

VI. Cohesión lógica: los elementos están destinados a que realicen actividades de una misma categoría general, pero la selección de la actividad concreta tiene lugar desde fuera del módulo

Ejemplo

El módulo se encargará de efectuar un pago que podrá realizarse en efectivo o por transferencia (el medio de pago se seleccionará desde fuera del módulo)

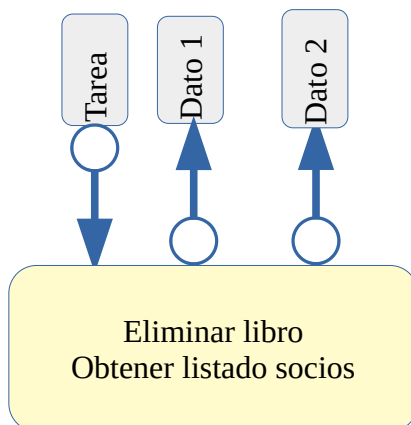
I.E.S. EL MAJUELO	
<i>Introducción a la programación</i>	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática



VII. **Cohesión casual:** los elementos no guardan ninguna relación observable y son fruto de una organización caótica.

Ejemplo

Dentro del módulo se realizarán tareas de diverso tipo pero sin relación alguna. La selección de la tarea en cuestión se realiza mediante un flag creado para tal efecto. Algunas tareas pueden devolver datos mientras que otras no.

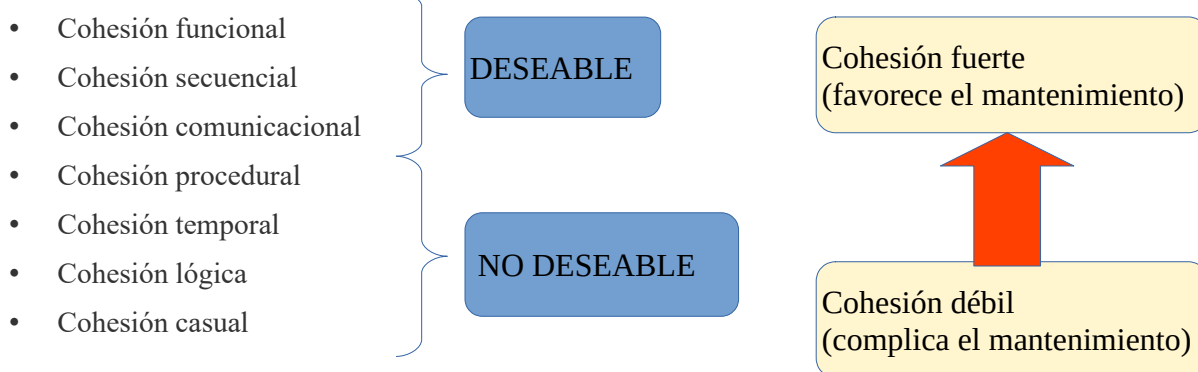


Se tiende a pensar que , a mayor cohesión , mejor será el diseño de un programa, hay que tener en cuenta que no todos los tipos de cohesión son deseables.

Esquema de los diferentes tipos de cohesión y cómo estos afectan al mantenimiento.

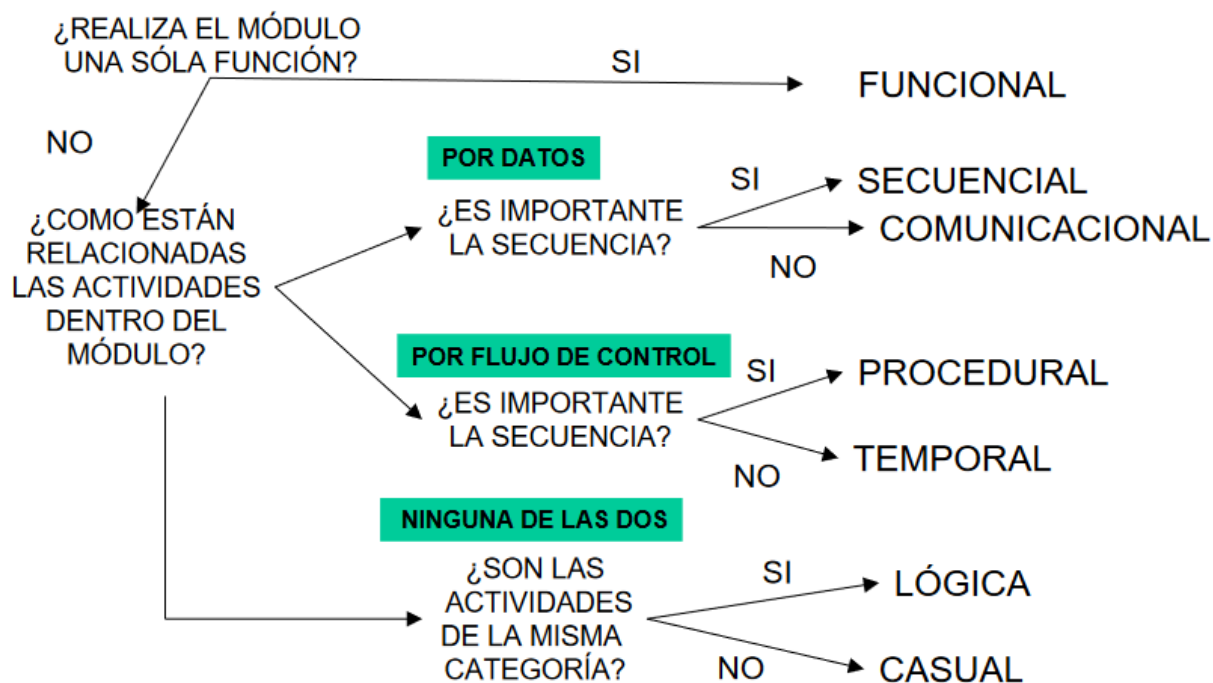
I.E.S. EL MAJUELO	
Introducción a la programación	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

Escala de cohesión



Mediante el siguiente árbol de cohesión se podrá identificar el tipo de cohesión ante el que un programador puede encontrarse.

A partir de una serie de preguntas situadas en los nodos se determina la cohesión dominante del módulo.



I.E.S. EL MAJUELO	
Introducción a la programación	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

B) Acoplamiento

El término acoplamiento hace referencia al grado y forma de dependencia entre los módulos. A menor cantidad de información compartida entre diferentes módulos , menor acoplamiento y por tanto mejor será el diseño.

El acoplamiento es una medida del grado de interdependencia entre los módulos de un sistema. Lo deseable es tener módulos con poco acoplamiento (**o independiente entre si**), para ser capaz de realizar el mantenimiento de un módulo sin tener que cambiar otros módulos

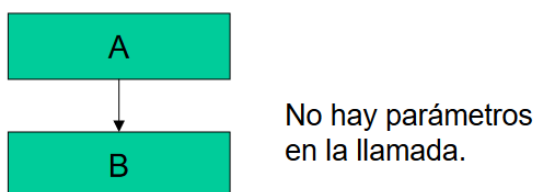
TIPOS DE ACOPLAMIENTO

- NORMAL
 - POR DATOS
 - POR ESTAMPADO
 - POR CONTROL
 - COMÚN
 - POR CONTENIDO
- Menos acoplamiento
↓
Más acoplamiento

1. **Acoplamiento normal** : se da en aquellos casos en los que un módulo (A) invoca a otro módulo (B)

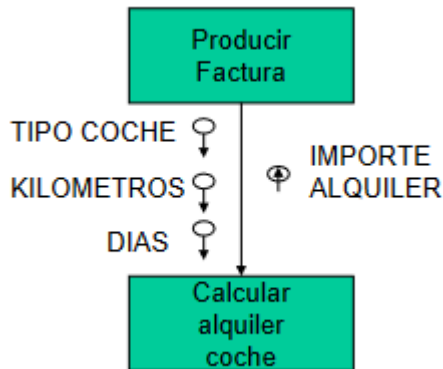
Ejemplo:

El módulo A invoca al módulo B, el cual tras realizar su función , retorna el control al módulo A



Toda la información que comparten la realizan a través de los parámetros presentes en la llamada. En este caso, y atendiendo al tipo de información , se definen tres subtipos:

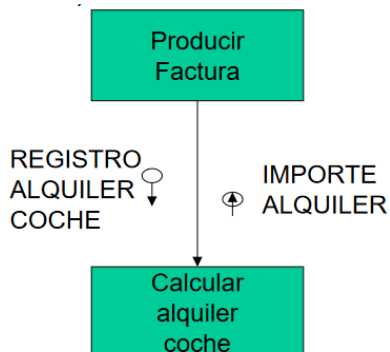
- a) **Acoplamiento normal por datos:** si los parámetros intercambiados son datos elementales sin estructura interna (tipo básicos)



El módulo “**Producir Factura**” pide unos datos simples al usuario (Cadena y número en este caso) y los almacena. Posteriormente invoca al módulo “**Calcular alquiler coche**” pasándole como parámetros los datos solicitados.

El módulo “**Calcular alquiler coche**” hace la operación indicada y devuelve el resultado

- b) **Acoplamiento normal por marca o estampado:** si los parámetros intercambiados son un dato compuesto (registro) de datos de tipo básicos.

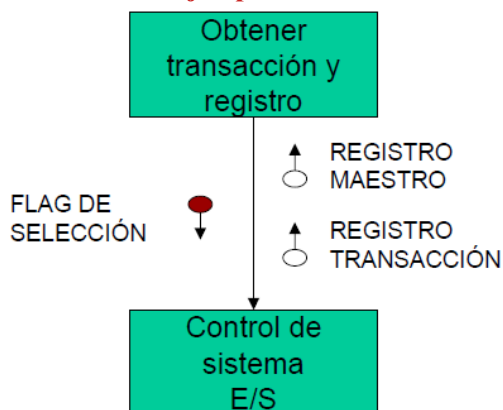


El módulo “**Producir Factura**” solicita datos simples al usuario y los almacena en forma de datos compuesto (registro). Posteriormente invoca al módulo “**Calcular alquiler coche**” pasándole como parámetro el dato compuesto.

El módulo “**Calcular alquiler coche**” calcula el valor para cada uno de los registros y devuelve el valor del importe.

- c) **Acoplamiento normal de control:** cuando un módulo le pasa al otro un parámetro con la intención de controlar su lógica de funcionamiento interna.

Ejemplo:



El módulo jefe (“**Obtener transacción y registro**”) controla, a través del dato “**Flag de selección**” , la lógica del módulo subordinado. Si fuera al revés se dice que hay una inversión de autoridad.

Valores del Flag de selección:

1. Obtiene registro maestro
2. Obtiene registro transacción
3. Obtiene ambos
4. Imprime cabecera
5. Etc.....

I.E.S. EL MAJUELO	
Introducción a la programación	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

2. **Acoplamiento externo:** se da cuando dos o más módulos utilizan las mismas fuentes externas de datos (interfaces de dispositivos o de programas externos).
3. **Acoplamiento global :** presente cuando los módulos utilizan los mismos datos globales (variables globales, memoria compartida, ficheros o bases de datos).
4. **Acoplamiento patológico o por contenido:** un módulo lee o modifica los datos internos de otro módulo o bien salta al interior de su código.

La programación modular intenta solventar el crecimiento desmesurado que se plantea con la programación estructurada mediante la descomposición en módulos o subprogramas de menor tamaño.

Cuanto mayor sea la relación existente entre los elementos contenidos en un módulo , mayor ser la cohesión y, por tanto, más exitosa la modularización llevada a cabo.
 Cuando menor sea la relación existente entre los elementos de un módulo con los elementos de otro módulo, menor será el acoplamiento y , por tanto, más exitosa la modularización realizada.

4.- Pseudocódigo

El pseudocódigo es un lenguaje cercano a un lenguaje de programación cuyo objetivo es el desarrollo de algoritmos fácilmente interpretables por un programador, independientemente del lenguaje de programación del que provenga. En si mismo no se trata de un lenguaje de programación pero si que utiliza un conjunto limitado de expresiones que permiten representar las estructuras de control y los módulos descritos en los paradigmas de programación estructurada y modular.

Mediante pseudocódigo se puede escribir aquellos algoritmos que tengan solución finita y que comiencen desde un único punto de partida. La escritura de un algoritmo o programa en pseudocódigo debería favorecer la posterior traducción al lenguaje de programación elegido.

Dado que no existe una sintaxis estandarizada para la escritura de pseudocódigo , ese posible encontrar diferencias sustanciales en los pseudocódigos escritos por diferentes programadores

4.1.- Operadores , palabras reservadas y tipos de datos.

A pesar de que no existe una norma rígida que establezca cómo realizar la escritura de programas en pseudocódigo, es recomendable seguir una serie de recomendaciones que permitan transcribir el programa al lenguaje de programación que va a usarse con la mayor facilidad.

I.E.S. EL MAJUELO	
Introducción a la programación	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

Operadores Aritméticos

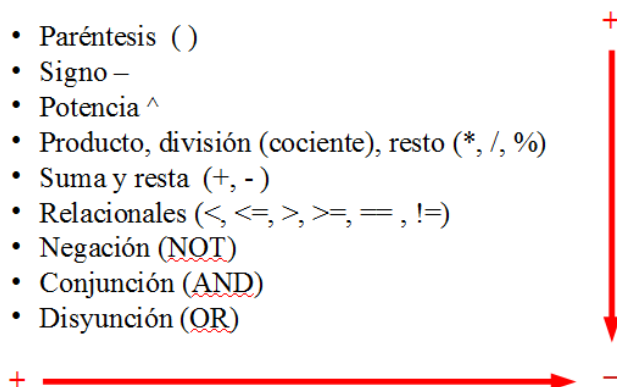
Aritméticos	
+	Suma
-	Resta
*	Multiplicación
/	División real
div	División entera
% o mod	Resto de la división
^	Potencia

Operadores Relacionales			
Operador	Operación	Ejemplo	Resultado
=	Igual que	'hola' = 'lola'	FALSO
<>	Diferente a	'a' <> 'b'	VERDADERO
<	Menor que	7 < 15	VERDADERO
>	Mayor que	22 > 11	VERDADERO
<=	Menor o igual que	15 <= 22	VERDADERO
>=	Mayor o igual que	35 > = 20	VERDADERO

Lógicos	
AND (&&)	Y lógico
OR ()	O lógico
NOT (!)	Negación
Especiales	
←	Asignación
//	Comentario de una sola línea
/**/	Comentario de varias líneas

I.E.S. EL MAJUELO	
Introducción a la programación	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

Prioridad de operadores

- Paréntesis ()
 - Signo –
 - Potencia ^
 - Producto, división (cociente), resto (*, /, %)
 - Suma y resta (+, -)
 - Relacionales (<, <=, >, >=, ==, !=)
 - Negación (NOT)
 - Conjunción (AND)
 - Disyunción (OR)
- 

Palabras reservadas

Inicio	Si no	Otro	Para	En
Fin	Según	Mientras	Hasta	Procedimiento
Si	Hacer	Repetir	Incremento	Función
Entonces	Caso	Hasta que	Cada	Imprimir
Leer	Retornar			
Carácter	Cadena	Entero	Real	Booleano

4.2.- Estructuras de control

El pseudocódigo utiliza las estructuras de control propias de la programación estructurada. Por este motivo se emplearán secuencias que representen las estructuras de control secuencial, alternativa e iterativa.

A) Estructura de control secuencial

Describen bloques de instrucciones que son ejecutadas en orden de aparición (secuencialmente). Los bloques pueden estar delimitados por las expresiones Inicio-Fin o bien estar contenidos en otras estructuras.

```

Inicio
    <instrucción 1>
    .....
    <instrucción n>
Fin
  
```

I.E.S. EL MAJUELO	
<i>Introducción a la programación</i>	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

B) Estructuras de control alternativa

La estructura de control alternativa o selectiva encauza el flujo de ejecución hacia un bloque de instrucciones u otro bloque en función de la evaluación que se realiza sobre una condición determinada.

Hay diferentes subtipos de este tipo de estructura de control:

1. **Alternativa simple:** establece un conjunto de instrucciones que se ejecutarán si se cumple una condición que retornará un valor booleano.

```

Si <condición> entonces
    <instrucción 1>
    .....
    <instrucción n>
Fin si

```

2. **Alternativa doble:** añade otro bloque de instrucciones que se ejecuta en caso de que no se cumpla la condición

```

Si <condición> entonces
    <instrucciones 1>
Si no
    <instrucciones 2>
Fin si

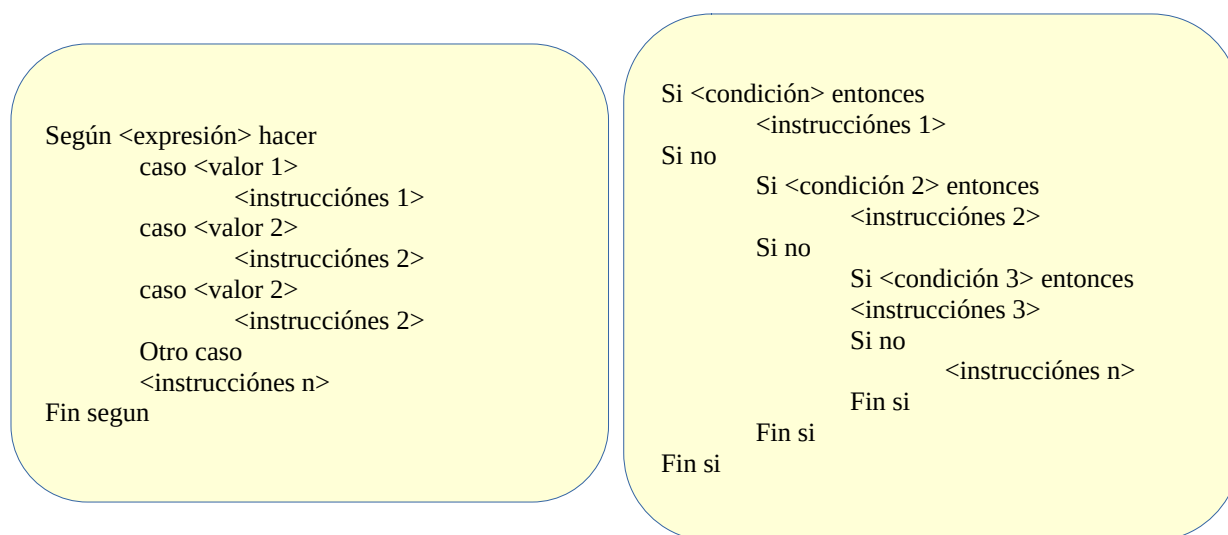
```

3. **Alternativa múltiple:** permite ejecutar diferentes bloques de instrucciones según el valor que tome una expresión que es comparada con los valores de cada caso o bien mediante el anidamiento de diferentes estructuras de alternativa doble cuyas condiciones son excluyentes.

I.E.S. EL MAJUELO	
Introducción a la programación	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

Ejemplo de dos bloques equivalente

BLOQUES EQUIVALENTES

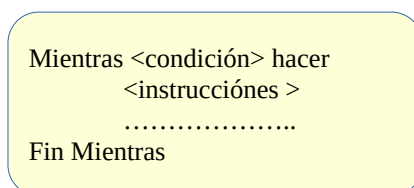


C) Estructuras de control iterativa

La estructura de control iterativa permite que un bloque de instrucciones sea ejecutado mientras se cumpla una condición.

Hay diferentes subtipos de este tipo de estructura de control:

1. **Iteración con salida al principio (while):** primeramente , evalúa la condición y en caso de cumplirse ejecuta el bloque de instrucciones. *La condición deberá cambiar de valor según las instrucciones contenidas para evitar bucles infinitos.* También es posible que nunca llegue a ejecutarse el bloque de instrucciones .



2. **Iteración con salida al final (repeat y do while):** primeramente , ejecuta el bloque de instrucciones y posteriormente evalúa la condición. *La condición deberá cambiar de valor según las instrucciones contenidas para evitar bucles infinitos.*

El bloque de instrucciones se ejecutará, como mínimo, una vez. La versión que hace uso de **repeat** ejecutará las instrucciones hasta que se cumpla la condición mientras que la variante **do while** ejecutará las instrucciones mientras se cumpla la condición.

I.E.S. EL MAJUELO	
Introducción a la programación	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

Repetir
 <instrucciones >

Hasta Que <condición>

Hacer
 <instrucciones >

Mientras <condición>

3. **Iteración con contador (for)** : ejecuta el bloque de instrucciones un número determinado de iteraciones. Hace uso de una variable que irá incrementando o decrementando por cada iteración hasta que se cumpla la condición de salida. La condición de control , que se traducirá a $1 \neq N$ o bien $i = 1$ según sea el incremento , deberá ser falsa para que sigan ejecutándose las instrucciones.

Ejemplo de for incremental y decremental

entero i
Para i ← 1 hasta N incremento 1 hacer
 <instrucciones >
Fin Para

Incremento positivo

entero i
Para i ← N hasta 1 incremento -1 hacer
 <instrucciones >
Fin Para

Incremento negativo

4. **Iteración para cada (for each):** ejecutará el bloque de instrucciones para cada elemento contenido en un conjunto .

entero i
Para Cada elemento en conjunto hacer
 <instrucciones >
Fin Para Cada

D) Estructuras modulares

También es posible describir mediante pseudocódigo la descomposición modular de los programas representando los procedimientos y las funciones encargados de realizar las tareas.

1. Procedimientos

Cada bloque de código contenido en un procedimiento se ocupará de llevar a cabo un conjunto de instrucciones cuando este sea invocado. Podrá recibir argumentos con los que trabajar a la hora de ser llamado si se ha definido el parámetro correspondiente, pero *en ningún caso retornará un valor de salida* .

Ejemplo de pseudocódigo para un procedimiento

I.E.S. EL MAJUELO	
Introducción a la programación	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

```

Procedimiento nombre (tipo parámetro1, tipo parámetro 2, ..... )
    <instrucciones >
Fin Procedimiento

```

Ejemplos típicos de programación correspondientes a HolaMundo y a un programa Saludo:

```

Procedimiento HolaMundo()
    imprimir ("Hola Mundo")
    imprimir ("_____")
Fin Procedimiento

```

```

Inicio
    HolaMundo()
Fin

```

```

Procedimiento Saludo (cadena nombre)
    imprimir ("Hola " + nombre)
Fin Procedimiento

```

```

Inicio
    cadena nombre
    imprimir ("Introduzca su nombre: ")
    leer (nombre)
    Saludo (nombre)
Fin

```

2. Funciones

Al igual que los procedimientos, una función podrán recibir argumentos o parámetros de entrada con los que trabajar a la hora de ser llamada si se ha definido el parámetro correspondiente, pero , **a diferencia de los procedimientos, retornará un valor de salida.**

Ejemplo de pseudocódigo para una función

```

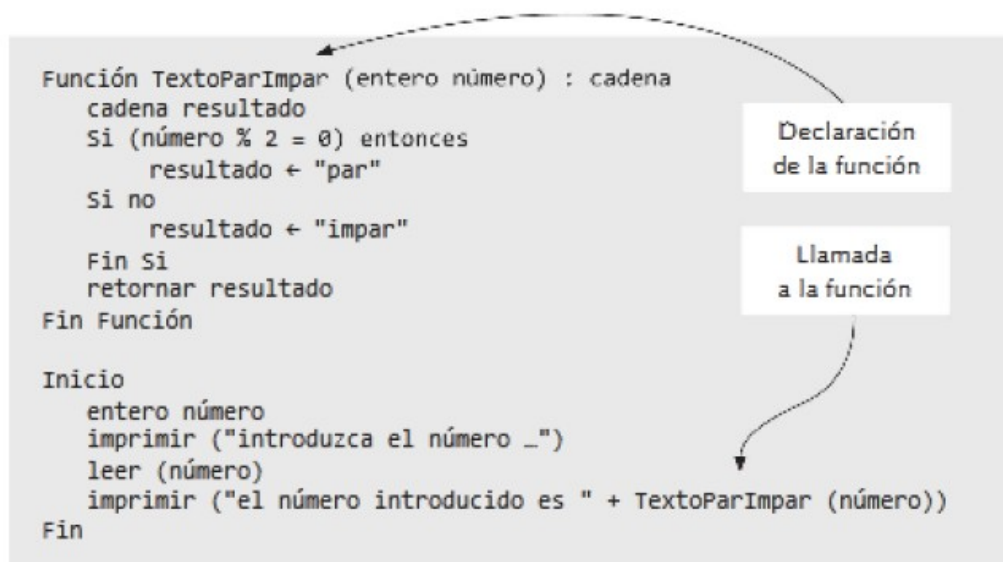
Función nombre (tipo parámetro1, tipo parámetro 2, ..... ) : tipo Retorno
    <instrucciones >
    .....
    retornar x
Fin Función

```

El pseudocódigo es un lenguaje que permite realizar una aproximación muy cercana a A la implementación de un problema determinado en un lenguaje real de programación (C, Pascal, Java,)
No se ajusta a ningún estándar y , por lo tanto , puede presentar diferencias dependiendo De qué programador lo escriba.

I.E.S. EL MAJUELO	
<i>Introducción a la programación</i>	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

Ejemplo típico de programación



Ejercicio

Empleando pseudocódigo desarrolla un programa que solicite la introducción de dos números por teclado (num1 y num2) y muestre por pantalla la suma, resta, multiplicación y división entera (siendo num2 el divisor). Ten en consideración que si num2 = 0 , el resultado será infinito

5.- Diagramas de flujo

Un diagrama de flujo , ordinograma o flujograma es una representación gráfica de un algoritmo o proceso. Son utilizados en informática , aunque también se emplean en otros ámbitos diferentes.

Facilita la comprensión del algoritmo gracias a la descripción visual que aporta sobre el flujo de ejecución de este.

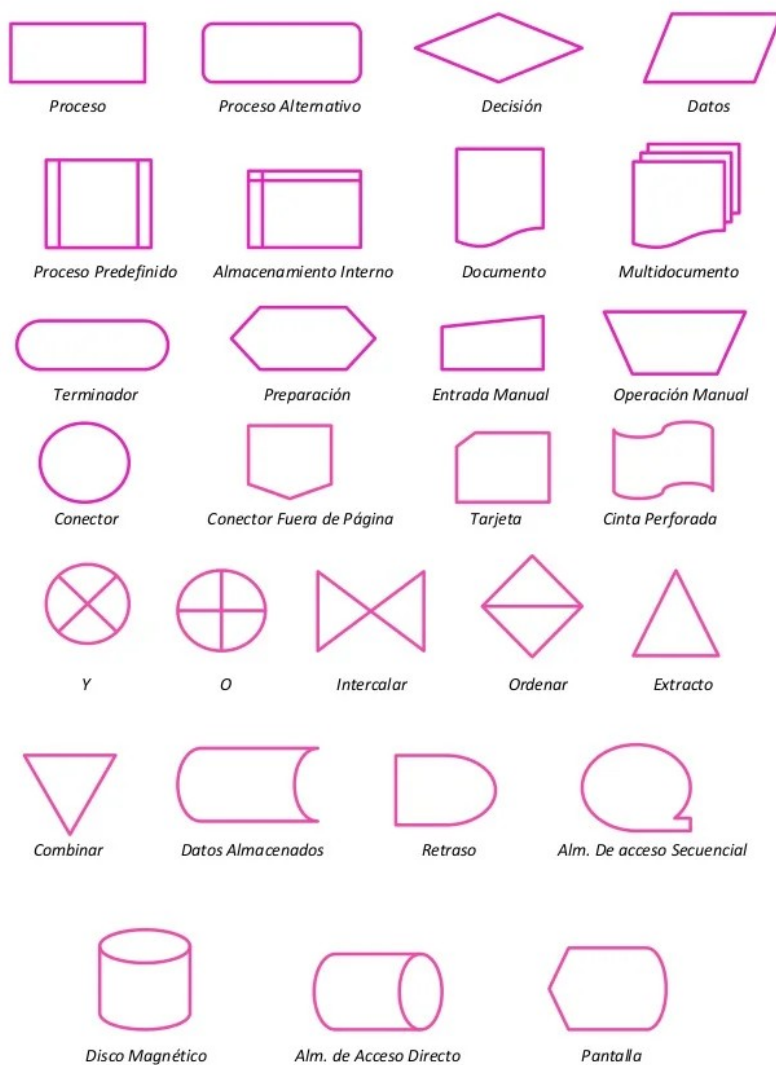
5.1.- Simbología

Los diagramas de flujo se construyen utilizando un conjunto de símbolos que se van enlazando entre sí para dar significado al proceso.

- Todo diagrama de flujo comienza y finaliza con un terminal representado mediante un óvalo o elipse.
- El trapecio rectángulo representa la entrada de datos desde el teclado. Indica la detención del proceso a la espera de que el usuario teclee los datos.

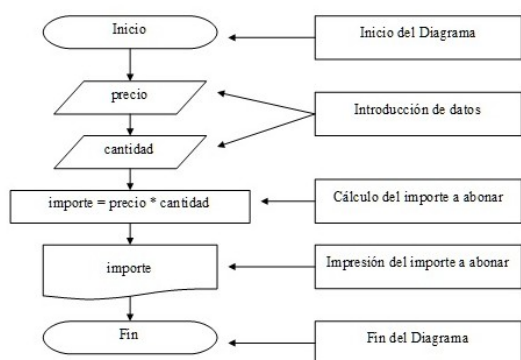
I.E.S. EL MAJUELO	
<i>Introducción a la programación</i>	<i>CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática</i>

- La comunicación con los periféricos para la entrada o salida de datos se representa mediante paralelogramo.
- En todo algoritmo existe un orden en el que se realizan sus operaciones. En los diagramas de flujo ese orden se indica mediante una flecha.
- Las decisiones se representan mediante un rombo del que salen tantas líneas de flujo como alternativas sean posibles.
- Si el algoritmo a representar es muy grande , puede ser complejo diseñar su diagrama en un único bloque y lo más conveniente será dividir el diagrama en bloques más pequeños. Para ello se pueden emplear **conectores** que representarán el punto al cual saltar una vez se llegue a ese punto. Los saltos se representarán mediante una circunferencia si son en la misma página, o con un pentágono irregular si son en otra.
- Los procesos que llevan a cabo las operaciones internas de cálculo se representan mediante un rectángulo.
- Es posible que haya determinadas tareas que, por su complejidad, no puedan ser representadas como un proceso. En ese caso serán descompuestas en subprocesos que estarán definidos en otro lugar y que están representados por un rectángulo con doble línea en cada lado.
- Las bases de datos con las que se intercambia información tienen su propio símbolo (cilindro).
- Los documentos impresos también tienen sus propios símbolos de representación.

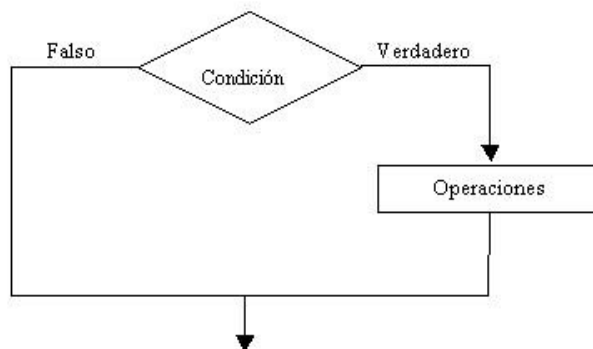


I.E.S. EL MAJUELO	
<i>Introducción a la programación</i>	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

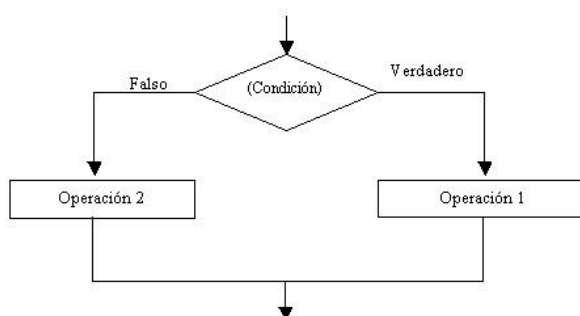
5.2.- Estructuras de control



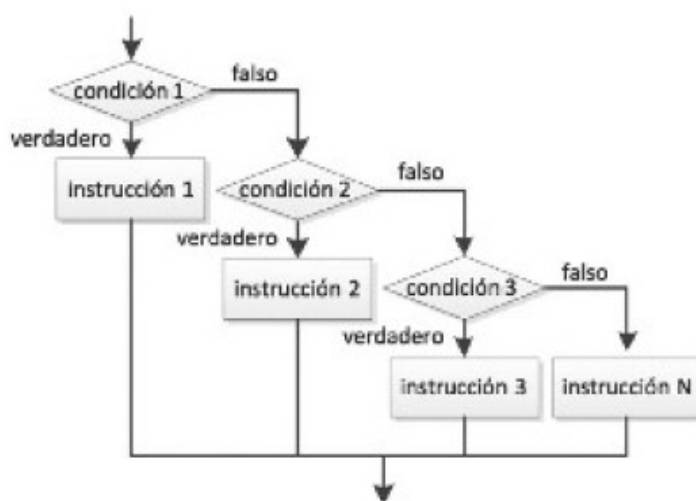
Secuencial



Alternativa Simple

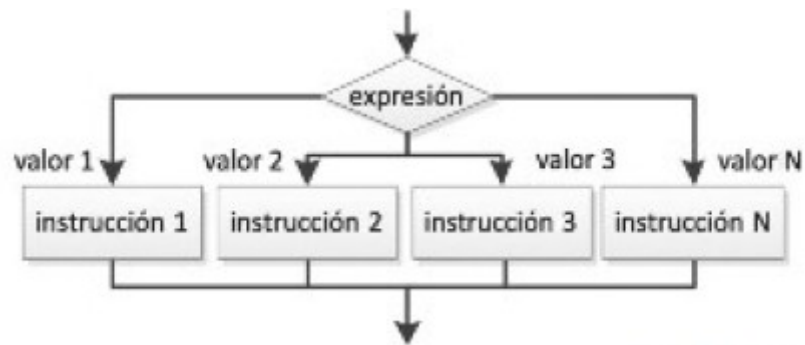


Alternativa doble



Alternativa múltiple con simples anidadas

I.E.S. EL MAJUELO	
Introducción a la programación	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática



Alternativas múltiples II



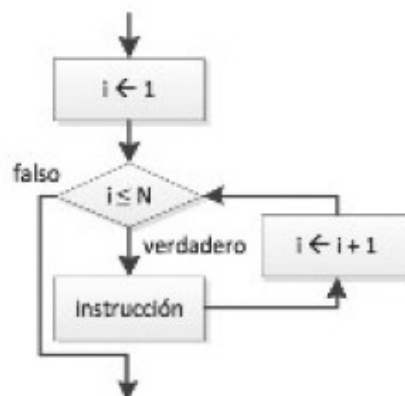
Iterativa while



Iterativa repeat

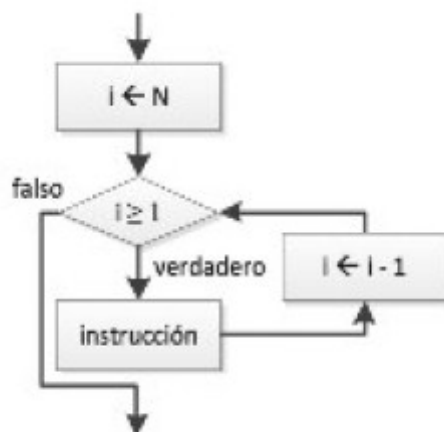


Iterativa do while



Iterativa for incremental

I.E.S. EL MAJUELO	
<i>Introducción a la programación</i>	CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática

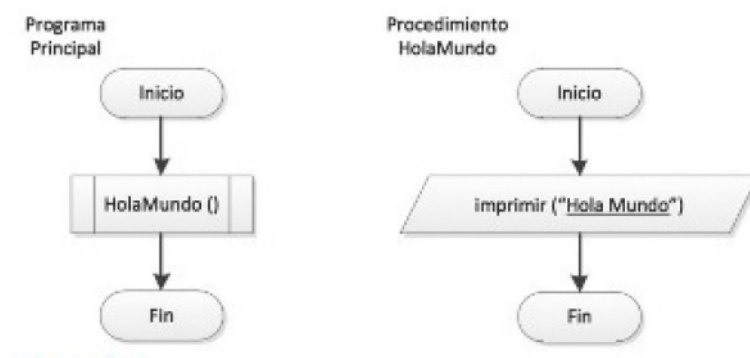


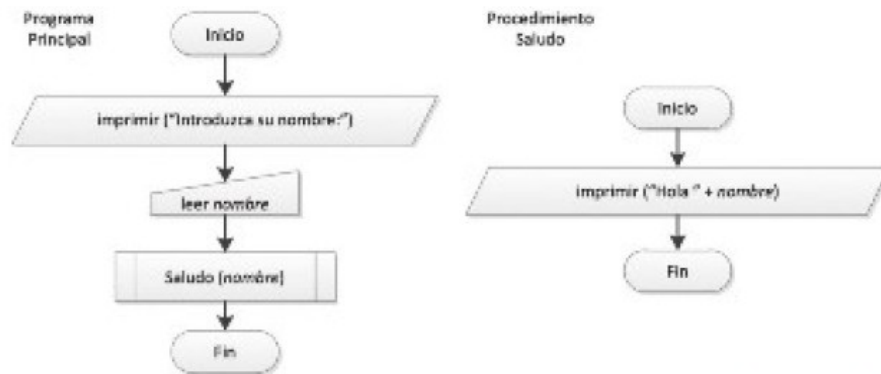
Iterativa for decremental

Cada función o procedimiento (*estructuras modulares*) quedará descrito por un diagrama independiente y la invocación a dicho subprograma se realizará desde el diagrama principal.

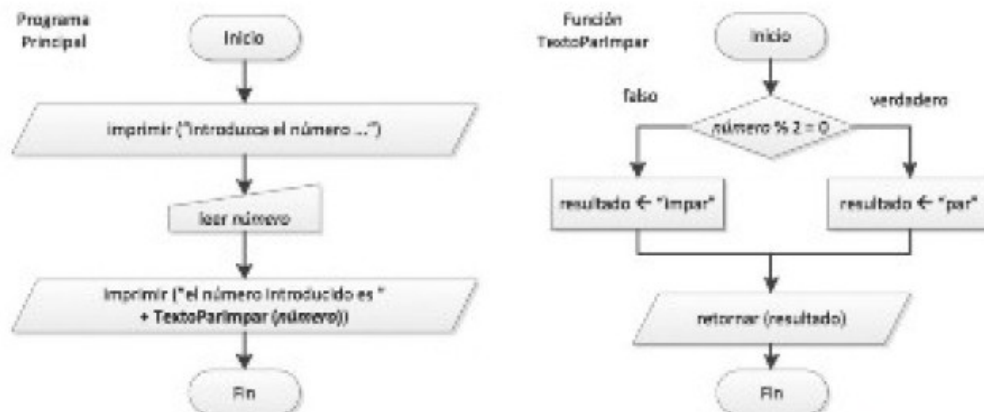
- En el caso de invocar un procedimiento , lo representaremos mediante una llamada al subproceso que queda identificado por el rectángulo de lados dobles. .

Diagramas correspondientes a los ejemplos realizados en pseudocódigo





Para invocar una función bastará con indicar la llamada en el proceso



Actividad del tema:

Realiza un documento de texto con las siguientes cuestiones:

1. Define el concepto de paradigma de programación
2. Describe las diferencias existentes entre una función y un procedimiento.
3. ¿Qué diferencia hay entre cohesión y acoplamiento?
4. Enumera y describe los tipos de estructuras de control que son empleados en programación estructurada.

I.E.S. EL MAJUELO	
<i>Introducción a la programación</i>	<i>CURSO ACADÉMICO 2022-2023 NIVEL C.F.G.S. D.A.M CURSO 1º (MÓDULO PROGRAMACIÓN) DEPARTAMENTO : Informática</i>

5. ¿En que consiste el concepto de divide y vencerás que permite poner en practica un procedo de desarrollo top-down?

Practicas

1.- Implementa mediante pseudocódigo un algoritmo que determine si un año es bisiesto.

NOTA: un año es bisiesto si es divisible entre 4 , a excepción de aquellos años que son divisibles entre 100 pero no entre 400

2.- Realiza la instalación de una aplicación que permita realizar diagramas de flujo. Una alternativa muy utilizada es Microsoft Visio, Libre Office Draw , Dia Diagram Editor, o bien alguna herramienta de diseño online como Lucidchart.

(Realiza un pequeño manual de instalación de dicha aplicación)

3.- Usando una de las herramientas indicadas en el ejercicio anterior, realiza el diagrama de un algoritmo que determine si un año es bisiesto.

4.- Completa los apartados 1 y 3 creando un programa principal que sea el encargado de solicitar al usuario el año del cual se pretende conocer si es o no bisiesto y que invoque el algoritmo representado en los apartados anteriormente mencionados para conocer el resultado.