



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: Marco Antonio Martínez Quintana

Asignatura: Fundamentos de Programación

Grupo: 3

No de Práctica(s): 4

Integrante(s): Ramírez Pérez Daniela Itzel

*No. de Equipo de cómputo
empleado:* No aplica

No. de Lista o Brigada: 36

Semestre: 1er

Fecha de entrega: Viernes 30 de octubre

Observaciones:

CALIFICACIÓN: _____

Objetivo:

Elaborar diagramas de flujo que representen soluciones algorítmicas vistas como una serie de acciones que comprendan un proceso.

Introducción

Un diagrama de flujo es una forma esquemática de representar ideas y conceptos en relación.











Relacionando este concepto con las ciencias de la computación, el diagrama es considerado la representación grafica de su algoritmo, que es lo que permite que nuestro código funcione.

Nuestro diagrama debe de estar comprimida lo mas que se pueda, debe de estar simbolizada correctamente y debe de ser comprensible para el lector.

Para su creación y comprensión usamos la siguiente simbología para nuestros diagramas de flujo:

\

Símbolos

Representa el inicio o el fin del diagrama de flujo.	
Expresión de Datos de entrada	
En el Proceso se indican asignaciones u operaciones.	
Toma de decisión	
Escritura. Impresión del o los resultado(s).	
Dirección	
Conexión dentro de la misma página	
Conexión entre diferentes páginas.	
Módulo de un problema. Llamada a otros módulos o funciones.	
Decisión múltiple. Almacena un selector que determina la rama por la que sigue el flujo.	

Estructuras de control de flujo

Una de las estructuras usadas para nuestros diagramas de flujo, son las estructuras de control de flujo cuyo objetivo es ejecutar o repetir un algoritmo bajo ciertas condiciones.

Las 3 estructuras de control de flujo son:

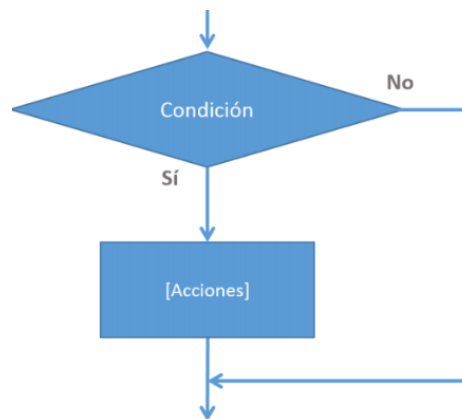
- Estructura de control secuencia
- Estructuras de control condicionales
- Estructuras de control iterativas o repetitivas

- **Estructura de control secuencial**

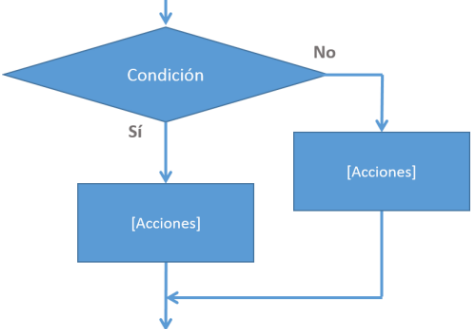

Son las sentencias o declaraciones que se realizan en el orden en el que están escritas.

```
x: REAL
x ← 5.8
x ← x*2
```

Es cuando se evalúa un algoritmo para determinar si es verdadero o falso, para que así realice un cierto flujo de instrucciones



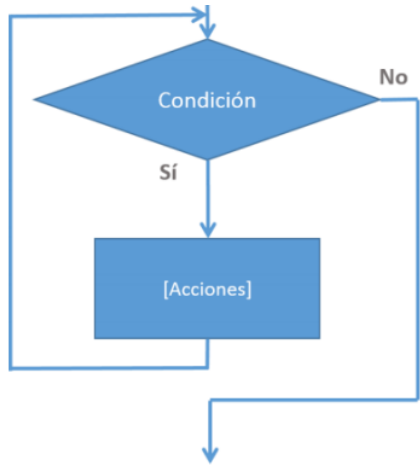
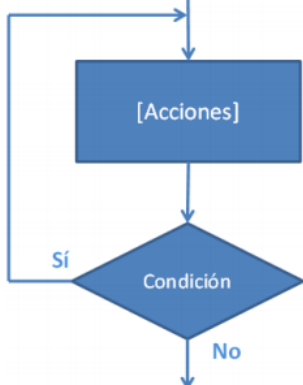
Este tipo de estructura se divide en 2 tipos de casos:

<p>○ SI-DE LO CONTRARIO (IF-ELSE):</p> <p>○</p> <p>Cuando una condición es verdadera se ejecutan las instrucciones del bloque Sí, y del mismo modo al no cumplirse se ejecuta las del bloque No.</p>	 <pre> graph TD Start(()) --> Condicion{Condición} Condicion -- Sí --> Acciones1[Acciones] Condicion -- No --> Acciones2[Acciones] Acciones1 --> Join(()) Acciones2 --> Join Join --> Exit(()) </pre>
<p>1. SELECCIONAR-CASO</p> <p>Se comprueba si el valor dado al diagrama está definido en cualquiera de los casos escritos, si no se encuentra definido por defecto se va por el caso marcado con [*]</p>	 <pre> graph TD Start(()) --> a{{a}} a -- valor1 --> Acciones1[Acciones] a -- valor2 --> Acciones2[Acciones] a -- valor3 --> Acciones3[Acciones] a -- "..." --> Acciones4[Acciones] a -- "*" --> Acciones4 Acciones1 --> Join(()) Acciones2 --> Join Acciones3 --> Join Acciones4 --> Join Join --> Fin([Fin]) </pre>

- **Estructuras de control iterativas o repetitivas**

Estas estructuras se les permiten ejecutar un algoritmo mientras se cumpla la expresión lógica.

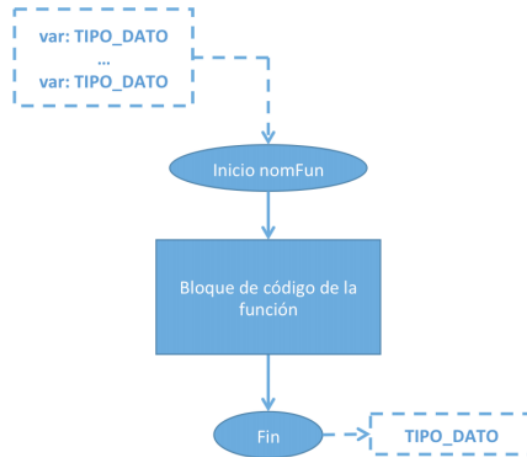
En esta estructura se encuentran dos tipos:

<p>2. MIENTRAS</p> <p>La estructura valida la condición y si ésta es verdadera procede a ejecutar el bloque de instrucciones de la estructura, de lo contrario rompe el ciclo y continúa el flujo normal del programa.</p>	 <pre> graph TD Entry(()) --> Condicion{Condición} Condicion -- Sí --> Acciones[Acciones] Acciones --> Condicion Condicion -- No --> Exit(()) </pre>
<p>3. HACER-MIENTRAS</p> <p>Primero ejecuta las instrucciones descritas en la estructura y al final valida la expresión lógica. Esta estructura es casi idéntica a la de MIENTRAS con la excepción de que esta estructura se cumple por lo menos una vez [Acciones] ya que HACER-MIENTRAS pide la condición después.</p>	 <pre> graph TD Entry(()) --> Acciones[Acciones] Acciones --> Condicion{Condición} Condicion -- Sí --> Acciones Condicion -- No --> Exit(()) </pre>

Funciones

Al ser la solución de algún problema demasiado complejo se ocupa un diseño descendiente, el cual divide al problema en subprocesos llamados módulos o *funciones*.

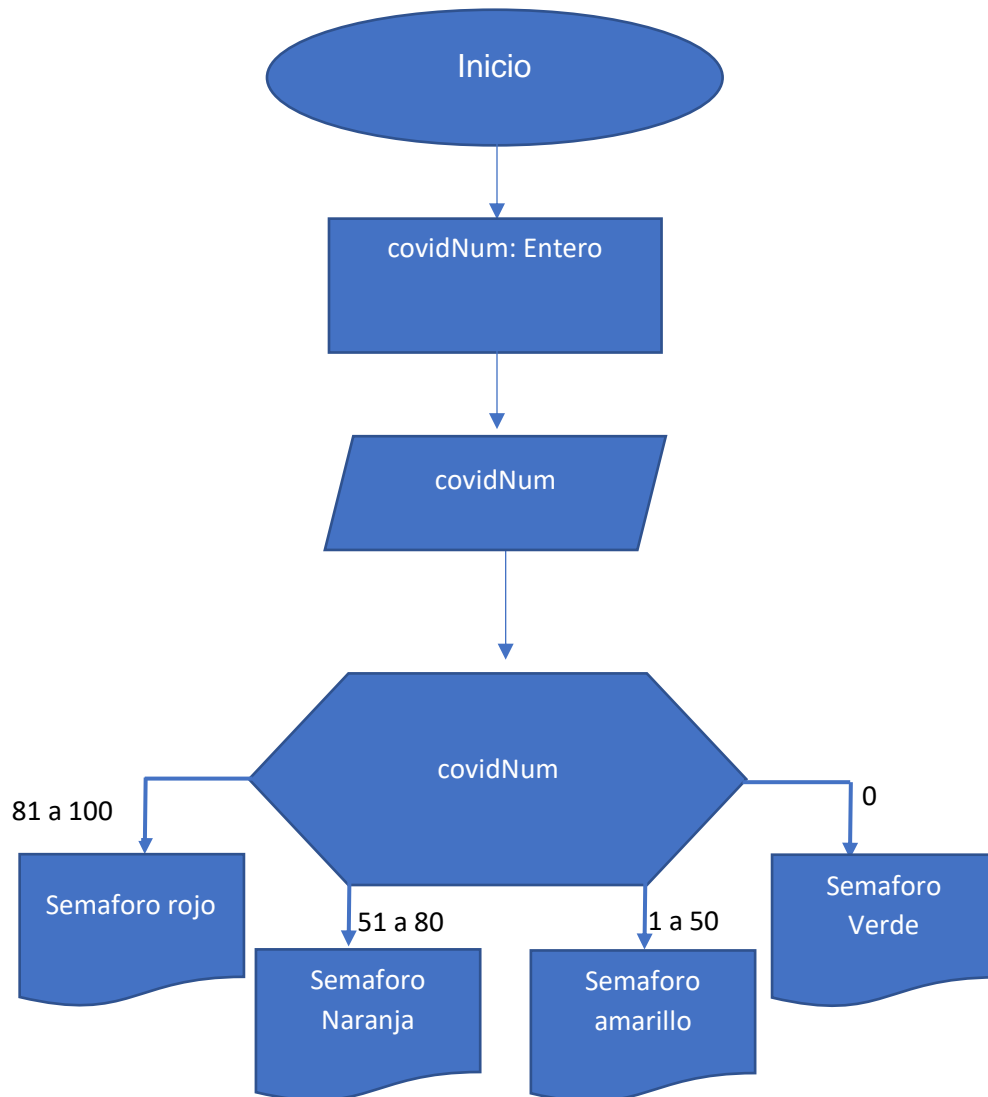
En el siguiente ejemplo se mostrara por lo que está constituida la función:



<ul style="list-style-type: none">Nombre	
<ul style="list-style-type: none">Parámetros de entrada (tipo de dato)	
<ul style="list-style-type: none">Conjunto de salida (tipo de dato).	
<ul style="list-style-type: none">Valor de retorno	

Actividades

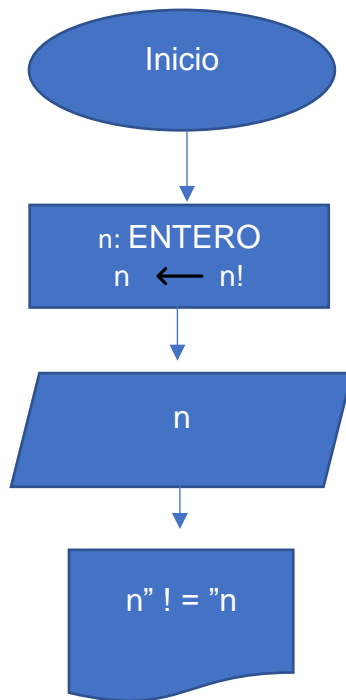
- Realizar un diagrama de flujo que determine el color del semáforo COVID en base a una muestra de 100 individuos:
 - Si hay más de 80 individuos con COVID el color del semáforo es rojo
 - Si hay de 51 a 80 individuos con COVID el color del semáforo es naranja
 - Si hay de 1 a 50 individuos con COVID el color del semáforo es amarillo
 - Si no hay individuos con COVID el color del semáforo es verde



- Realizar un diagrama de flujo que calcule dado un número el cálculo de su factorial:

Ejemplo:

- $1! = 1$
- $2! = 2$
- $3! = 6$
- $4! = 24$



Conclusión

Con estos ejercicios de creación de diagramas de flujo podemos entender de mejor manera los algoritmos con ayuda grafica y tener una mejor idea de como funciona el proceso lógico que se lleva a cabo.