|  |  |  |
| --- | --- | --- |
|  | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorio de docencia |

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| *Profesor:* | ***Ing. Karina García Morales*** |
| *Asignatura:* | ***Fundamentos de Programación (L) laboratorio*** |
| *Grupo:* | ***21*** |
| *No de Práctica(s):* | ***Practica 4*** |
| *Integrante(s):* | ***Carbajal Alvarez Gerardo*** |
| *No. de Equipo de cómputo empleado:* | 28 |
| *No. de Lista o Brigada:* |  |
| *Semestre:* | ***2020-2*** |
| *Fecha de entrega:* | ***Martes 3 de Marzo 2020*** |
| *Observaciones:* |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_

**Diagramas de Flujo.**

Objetivo

* Elaborar diagramas de flujo que representen soluciones algorítmicas vistas como una serie de acciones que comprenden un proceso

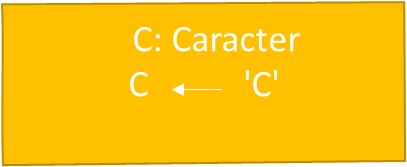
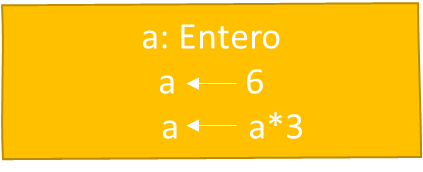
Desarrollo de la Practica

Diagrama de Flujo

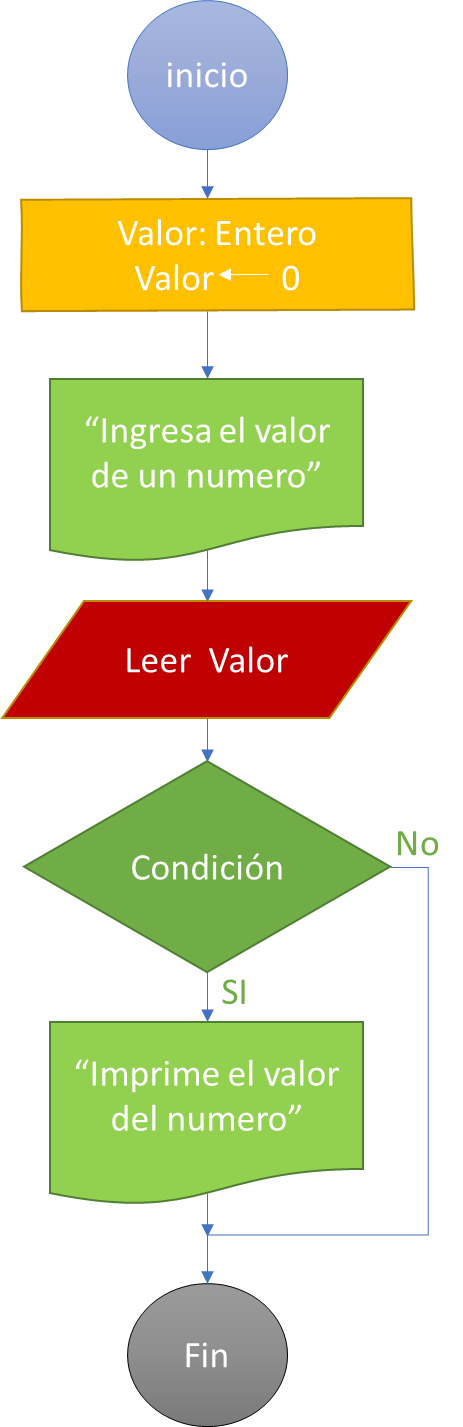
* Un diagrama de flujo es la representación gráfica de un Algoritmo, ilustra gráficamente las acciones secuenciales del algoritmo, tomas de decisiones y procesos repetitivos que cumplen con la ejecución de una tarea.
* La correcta elaboración del diagrama de flujo es fundamental para la etapa de codificación en cualquier lenguaje de codificación.
* A continuación, se describen las figuras básicas para la elaboración de un diagrama de flujo

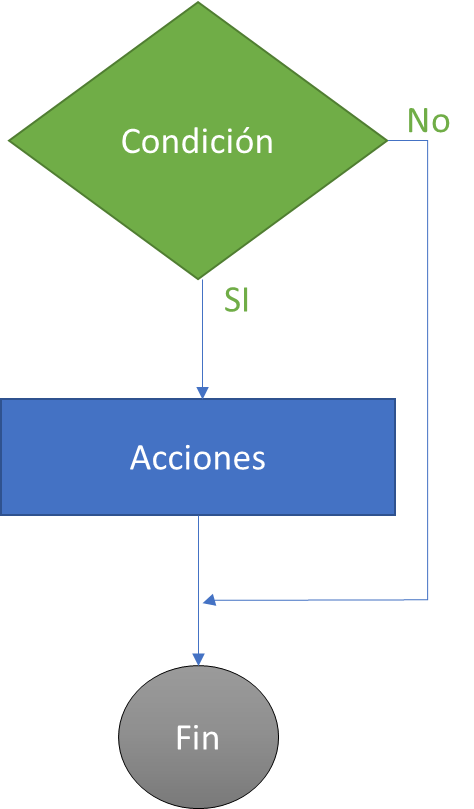
Estructura de Control Secuencial.

* La estructura de control secuencial son la declaración de variables y asignación de tipo de datos a cada una de ellas una tras otra en el orden secuencial escrito.
* La Siguiente Actividad permite declarar una variable tipo entero y asignarle el valor numérico de 6, posteriormente esa variable se multiplica por 3, el siguiente seria como declarar una variable tipo Carácter, los caracteres son letras basadas en el código ASCII la variable tipo carácter se le asigna la letra C.



Estructuras de Control de Selección (Condicionales).

* Las estructuras de Selección permiten evaluar una condición lógica y dependiendo si es verdadera o falsa se toma la decisión de (**si)** ir por un camino y realizar una acción o (**No)** ir por otro camino y realizar otra acción.
* **La Condicional simple** se conoce como **IF.**
* Ejemplo 1 Pedir al Usuario un valor si el valor es mayor a 100 imprimirlo.



**La Condicional simple** se evalúa

Una condición lógica, si la condición se

cumple, se realizan las acciones, si no

se cumple la condición, continua con

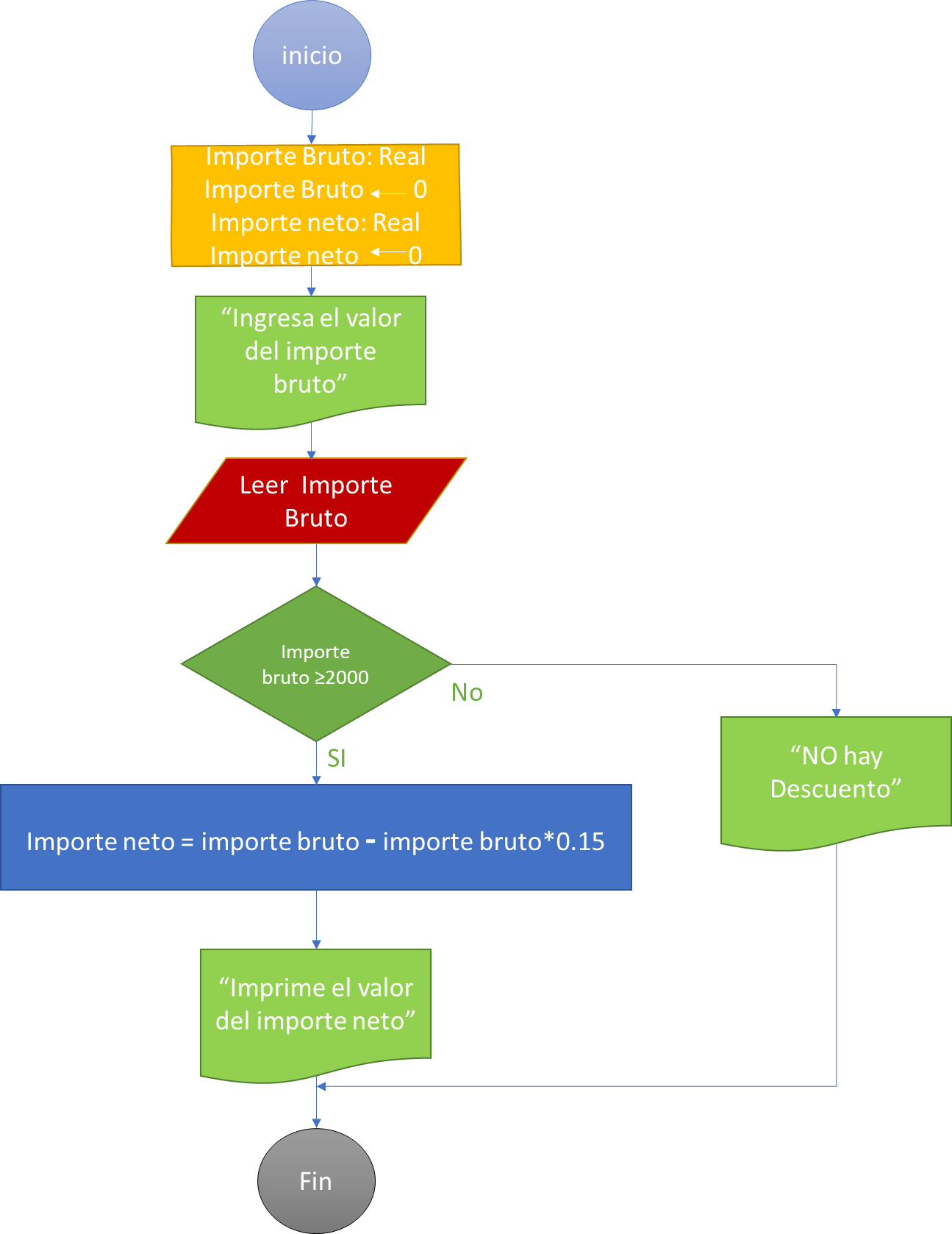
la ejecución normal del programa

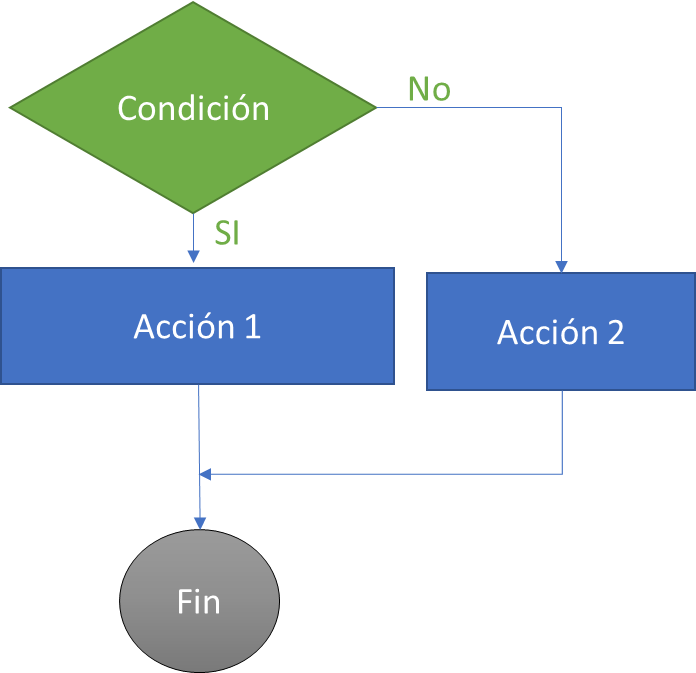


**Figura 1** Condicional Simple **IF**

**Figura 2** Solución Ejemplo1

* **La Condicional Compuesta** se conoce como **IF ELSE** evalúa la condición lógica si esta es **verdadera** se ejecutan las acciones del camino **SI**, de lo contrario si no se cumple la condición, es decir es **Falsa** se ejecutan las acciones del camino **NO.**
* Ejemplo 2 Solicitar al Usuario el importe bruto de una factura y determinar el importe neto según el criterio importe bruto menor de 20000 sin descuento, importe bruto mayor o igual a 20000 se aplica el 15% de descuento, imprime el descuento y el importe neto.

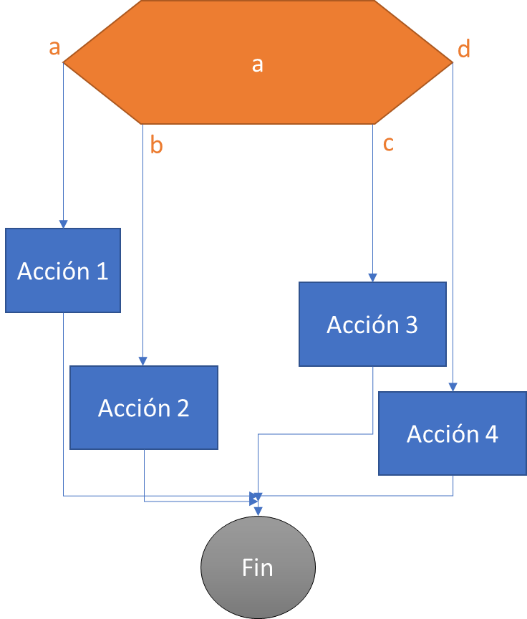




**Figura 3** Condicional Compuesta **IF ELSE**

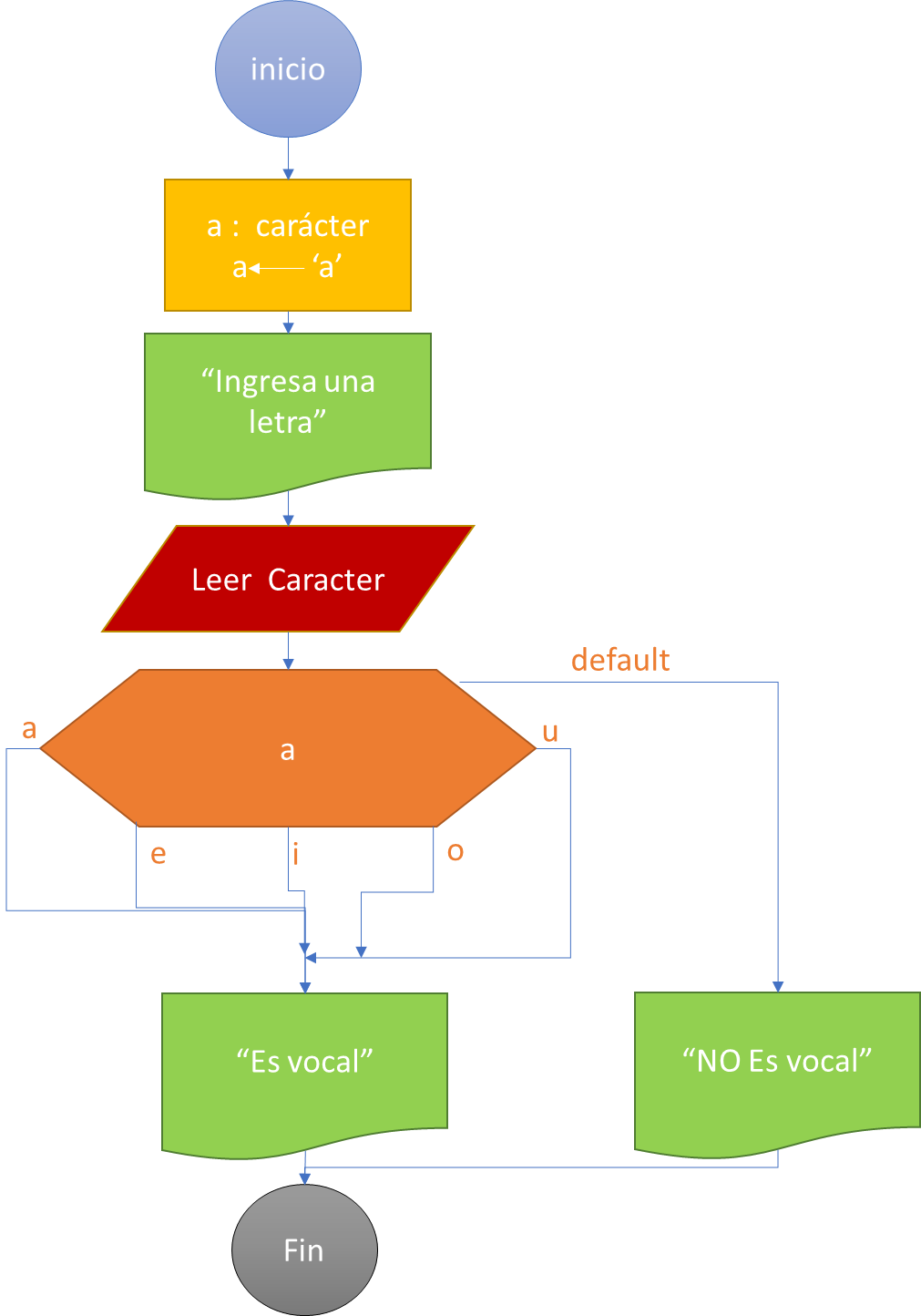
**Figura 4** Solución Ejemplo 2

* **La Condicional Múltiple** se conoce como **CASE;** verifica el valor de la variable dentro del hexágono y comprueba si es igual al valor definido para cada caso, toma un camino diferente dependiendo si el valor ingresado es el mismo que el valor de alguno de los caminos, si la variable no es ningún valor de los definidos se va al valor por defecto



**Figura 5** Condicional múltiple **CASE**

* Ejemplo 3 Pedir al Usuario un carácter e indicar si es o no una vocal.

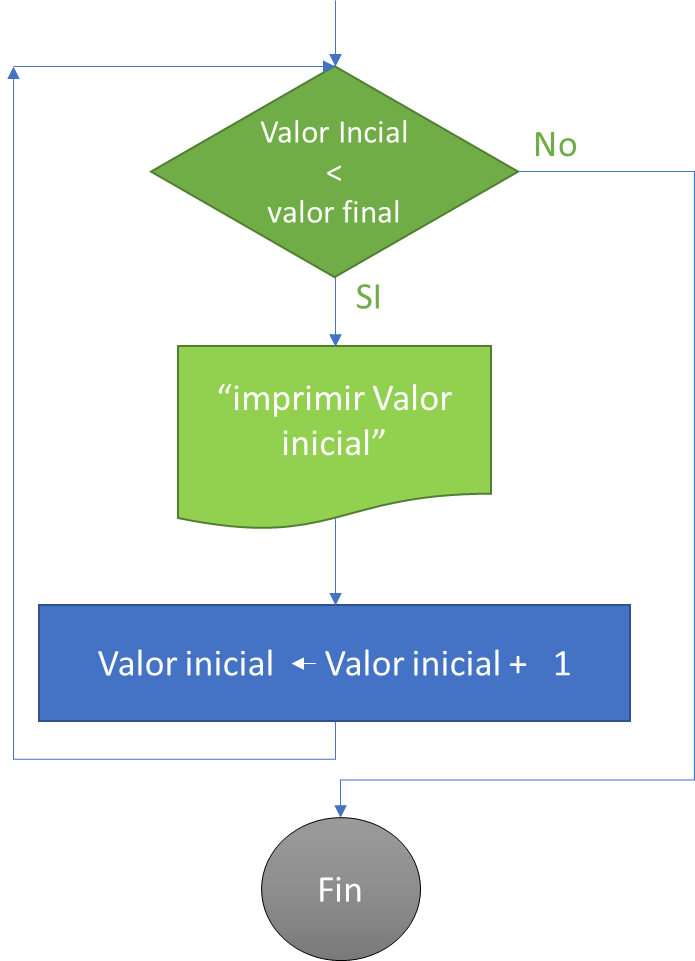


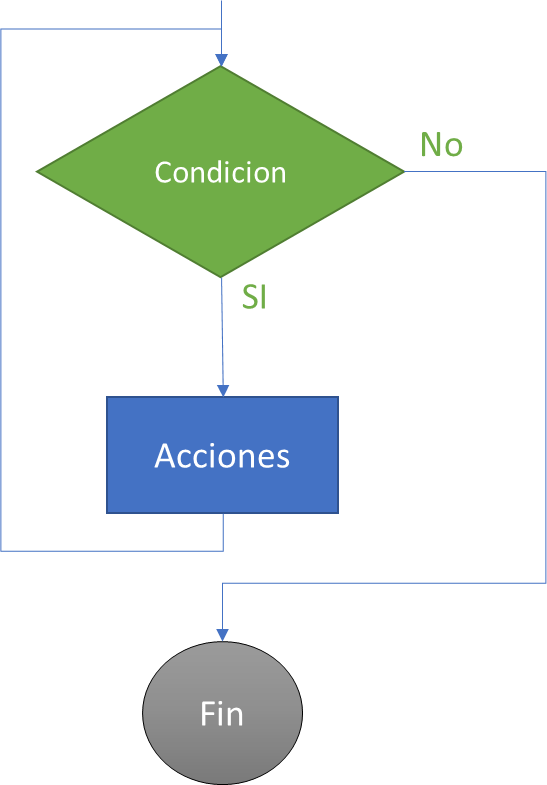
**Figura 6** Solución Ejemplo 3

Estructura de Control de Repetición (iterativas)

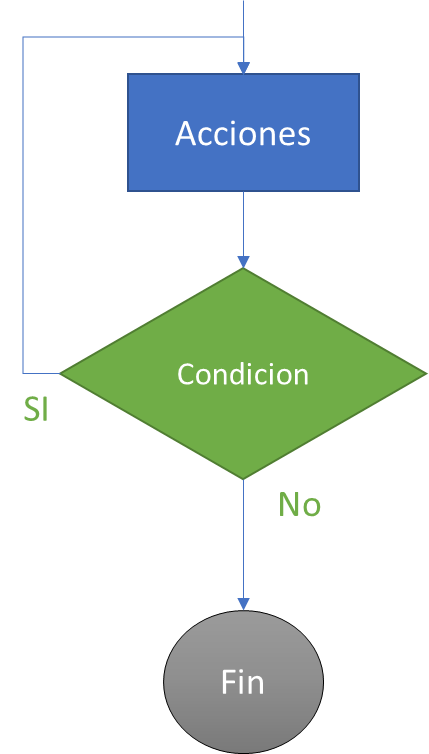
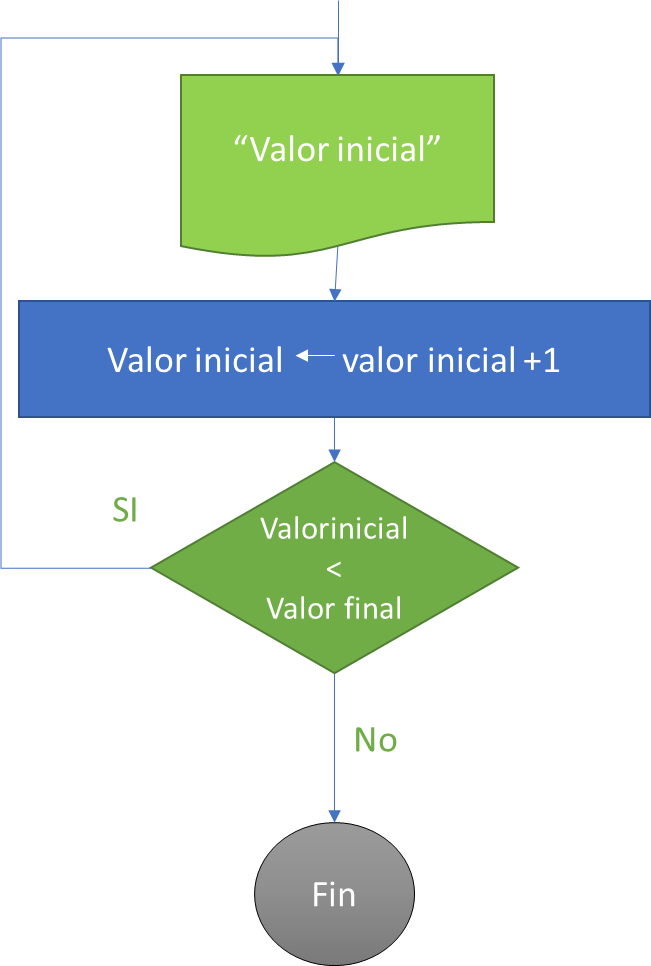
* La estructura de control repetitiva permite realizar instrucciones iterativas o cíclicas donde por cada vez que se ejecuta un ciclo o una iteración se incrementa una variable las iteraciones se repiten mientras que se cumpla una condición.

* **La Estructura Mientras** se conoce como **WHILE** primero valida la condición lógica si esta se cumple se ejecutan las acciones del ciclo, cuando ya no se cumpla la condición se sale del ciclo.

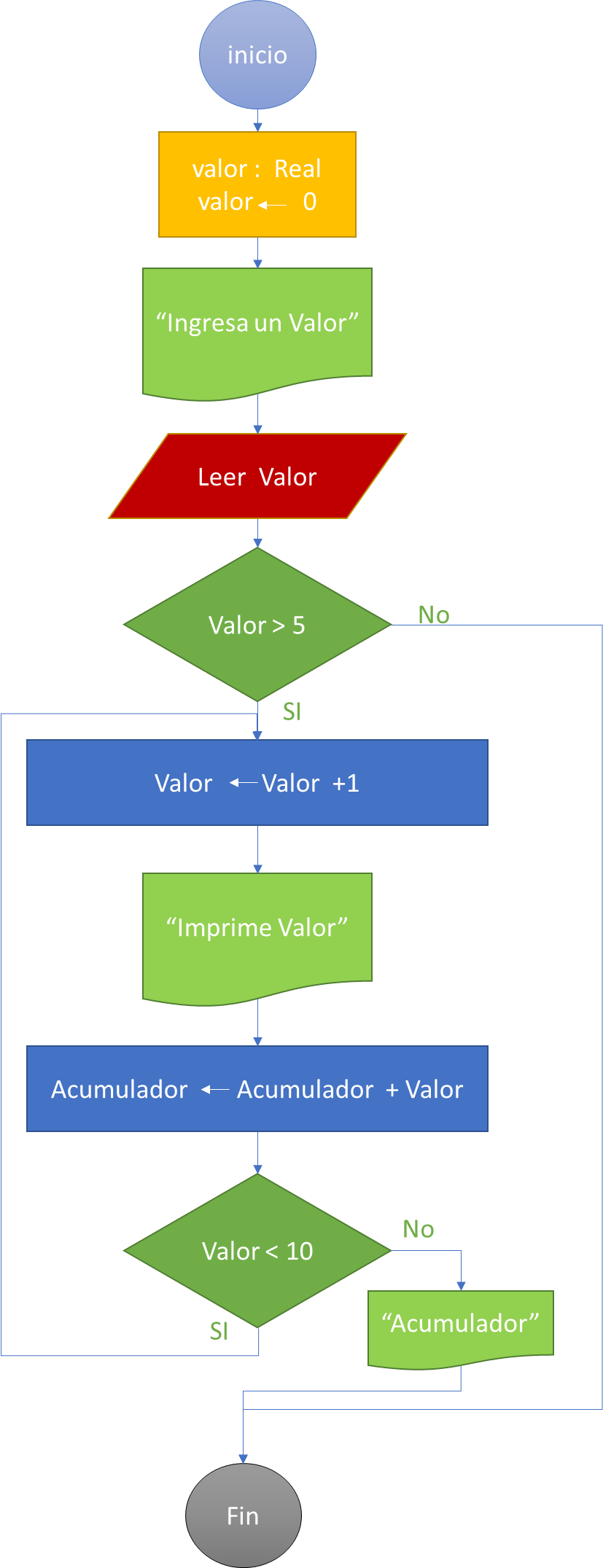




* **La Estructura Hacer Mientras** se conoce como **DO** **WHILE** primero ejecuta las acciones y después valida la condición lógica si esta se cumple vuelve a ejecutar las acciones del ciclo, cuando ya no se cumpla la condición se sale del ciclo.

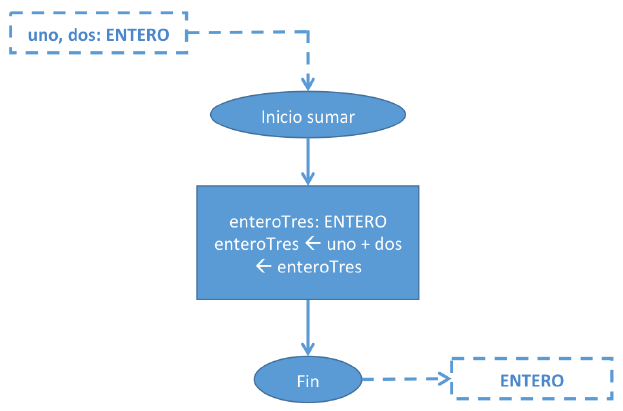


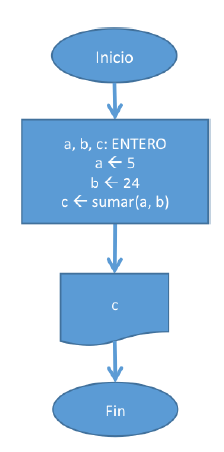
* Ejemplo 3 Solicitar un valor al usuario y validar si es mayor a 5 sumar 1 e imprimir la suma de los números hasta llegar al 10.



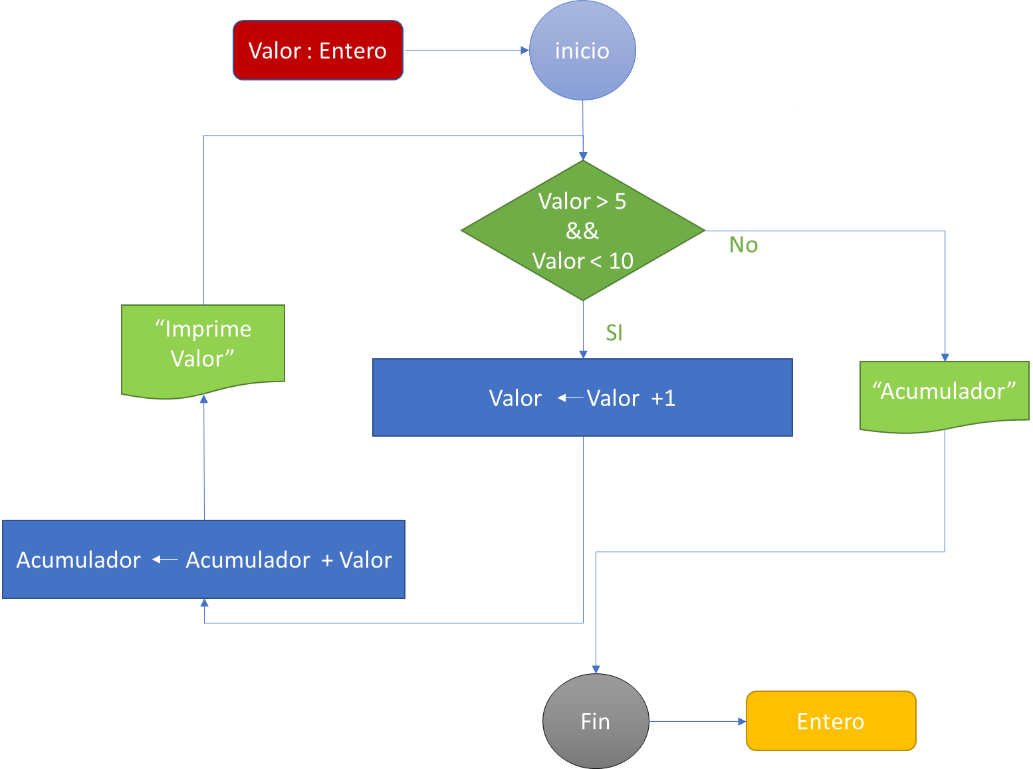
Funciones

* Cuando un programa es muy complejo de resolver, se suele dividir el algoritmo en varios subprocesos, funciones o subrutinas. *Divide y Vencerás.* Este diseño divide el programa en subrutinas más sencillas que juntas forman la solución completa*.*
* A estos subprocesos se les llaman módulos o funciones.



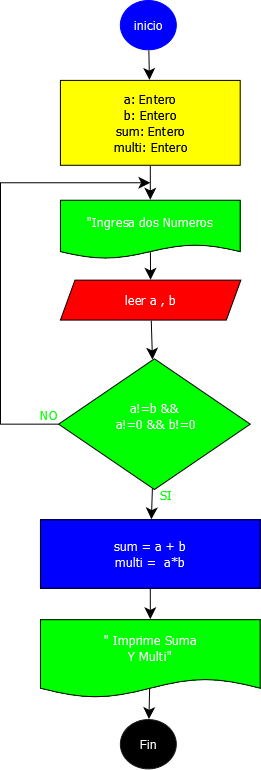


* En este ejemplo se ejecuta un programa principal donde se declaran dos números enteros, después se manda a llamar a la función sumar, los dos números declarados entran a la función donde se suman en ese subprograma cuando finaliza la función esta devuelve un resultado entero al programa principal

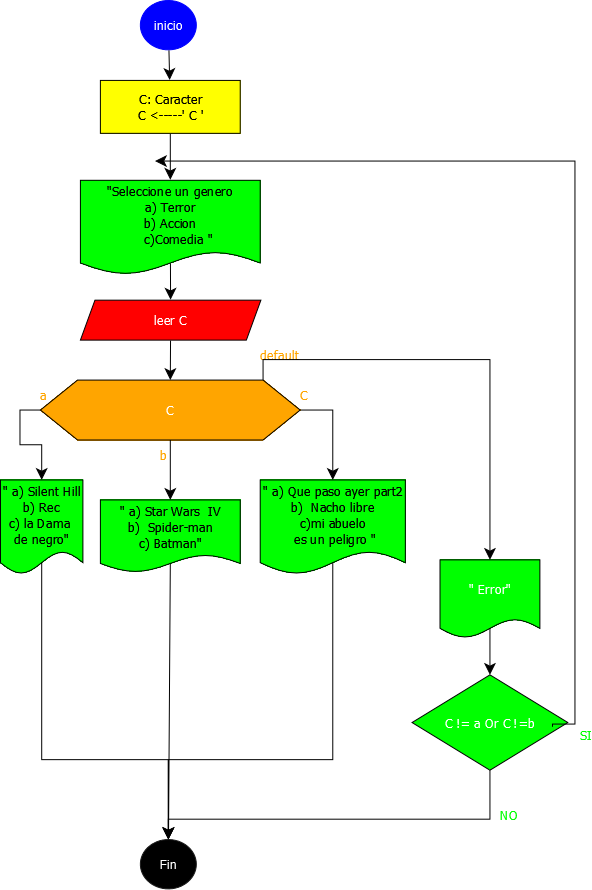


EJERCICIOS DE TAREA

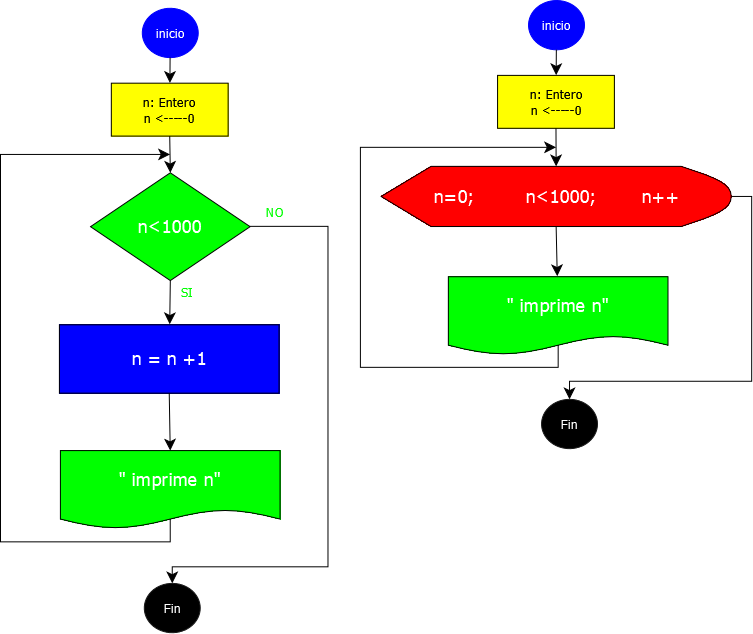
* Resuelve el programa que pida a un usuario dos valores enteros diferentes entre ellos y diferentes de cero; si ingresa los valores adecuados realizar la multiplicación y la suma de ellos, en caso contrario volverlos a pedir y realizar lo indicado anteriormente para que al final muestre ambos resultados.



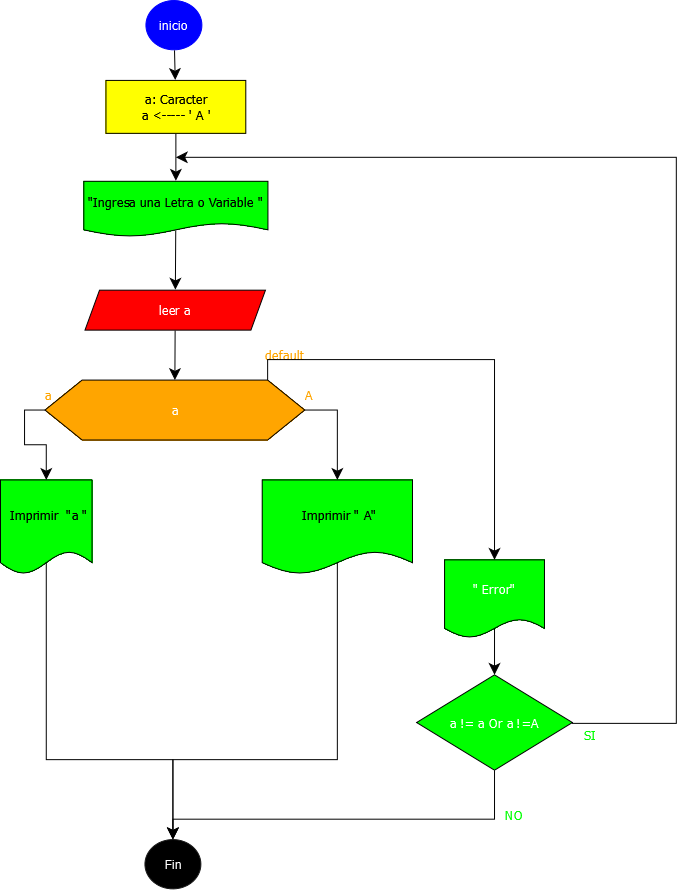
* Del ejercicio de clase (el menú de películas) agregar el ciclo hacer mientras, en donde valide, si el usuario no elige las opciones 1,2 ó 3(para carácter pueden ser 'a', 'b' o 'c') imprimir un letrero de "error"(es el default) y volver a mostrarle el menú.



* Realiza un diagrama que muestre la numeración del 1 al 1000, emplea la estructura MIENTRAS y realizarlo también con la estructura PARA



* Solicitar al usuario que ingrese la variable, si ingresa una variable diferente a la letra 'a' o 'A', volver a solicitarla, en caso de contrario, imprimir la letra 'a' o 'A' que ingresó el usuario.



Conclusiones

• Se ha comprendido como Diseñar Diagramas de flujo que representan Algoritmos Gráficos, se aprendió en la práctica como realizar los diagramas para estructuras Condicionales, Repetitivas y Además aprendimos como ejecutar un subproceso dentro de un programa mandando llamar a una función.

Bibliografía

Manual de Practicas de Fundamentos de Programación.

Metodología de la programación a través de pseudocódigo. Miguel Ángel Rodríguez Almeida, primera edición, McGraw Hill

Metodología de la programación. Osvaldo Cairó, tercera edición, México D.F., Alfaomega 2005.