|  |
| --- |
| Laboratorio de Computación  Salas A y B |
|  |

|  |  |
| --- | --- |
| Profesor(a): | García Morales Karina |
| Asignatura: | Fundamentos de programación |
| Grupo: |  |
| No de Práctica(s): | 4 |
| Integrante(s): | Viveros Fernández Salvador Juan |
|  |  |
|  |  |
|  |  |
| No. de lista o brigada: | 54 |
| Semestre: | Primero |
| Fecha de entrega: | 11/9/2024 |
| Observaciones: |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

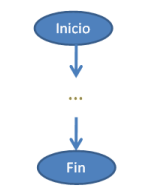
**Diagramas de flujo.**

**Objetivo:**

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

**Desarrollo:**

**Formas de los diagramas de flujo:**

Al un diagrama de flujo ser la representación gráfica de un algoritmo, estos contienen diversas formas para representar el comando que el paso en el que el programa se encuentre requiera, siendo estas las siguientes:

Todo programa al necesitar un inicio y un fin, se le selecciona el óvalo, además de las palabras Inicio y Fin para constatar donde se empieza y donde finaliza el programa creado.



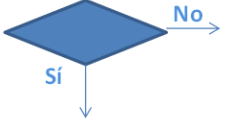
Al tener un orden de ejecución, se utilizan flechas para poder marcar a este mismo.



Para poder ingresar datos a un programa y que el sistema pueda leer datos, se utiliza el romboide.



Para representar operaciones y procesos, se utiliza el rectángulo.



Para decisiones simples de sí y no, llevando a diferentes caminos dependiendo de la respuesta, el rombo se es usado el rombo.



El poder hacer impresiones de pantalla también es necesario, por lo que se ocupa a:



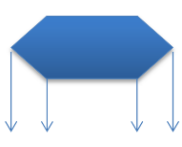
En ocasiones, el espacio no es necesario para almacenar todo un diagrama de flujo, por lo que se utiliza al círculo.



En ocasiones, se debe de dar saltos de página entre las instrucciones de un diagrama de flujo, por lo que se es utilizado el:



Para poder llamar a otros módulos y funciones se es utilizado:



Para la decisión múltiple se requiere a:

**Leyes de los diagramas de flujo:**

Para realizar un diagrama de flujo de calidad, se es requerido que se cumplan diversas reglas, las cuales son

-Todo diagrama de flujo debe de contar con un inicio y un fin.

-La dirección de las flechas que marcan el orden del diagrama, deben de ir estrictamente de arriba hacia abajo o, en su defecto de izquierda a derecha.

-Todas las flechas deben de estar asociadas con alguna figura, por lo que, no es permitido que alguna quede sin conexión.

-La notación del diagrama, debe ser independiente del lenguaje de programación que se vaya a utilizar.

-Si el diagrama de flujo ocupa más espacio que una hoja, se es requerido utilizar las notaciones correspondientes.

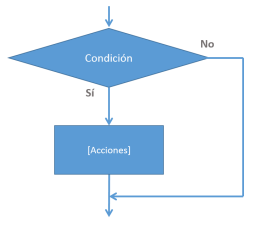
-Para nombrar variables y nombres de funciones, se debe de utilizar la notación de camello.

**Notación de camello:**

Apara poder nombrar una variable o función, se debe de utilizar la notación de camello, dicha notación consistes en empezar por una letra minúscula, y, si esta lleva más de una palabra, en lugar de utilizar el espacio, se utilizan mayúsculas para poder hacerlas legibles.

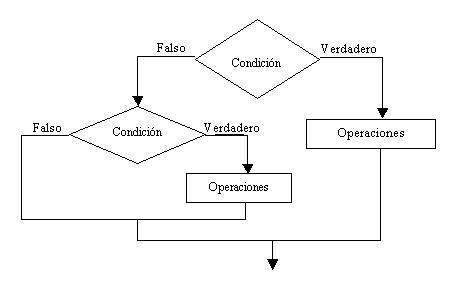
**Condicional simple:**

Al poder tener opciones dentro de un diagrama de flujo, se es necesario utilizar las condiciones, en este caso, de cumplirse se llevarán a cabo acciones, de cumplirse se llevarán a cabo diversas acciones, de no hacerlo, el diagrama continuará su flujo.



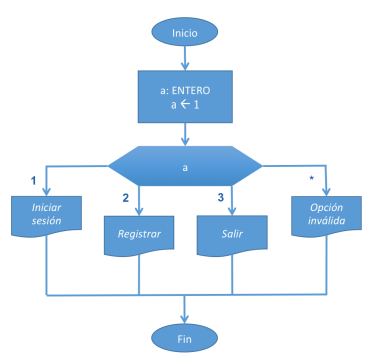
**Condicional anidada:**

En esta versión, se parte desde una condición, y dependiendo del resultado, alguno de ellos debe tener una nueva condicional.



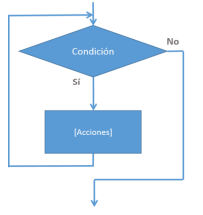
**Condicional múltiple:**

Al necesitarse más de dos variables, se es utilizada la condicional múltiple, pudiendo llevar a cada camino a procesos diferentes. Llevando además, la opción default, en caso de que ninguna opción sea seleccionada



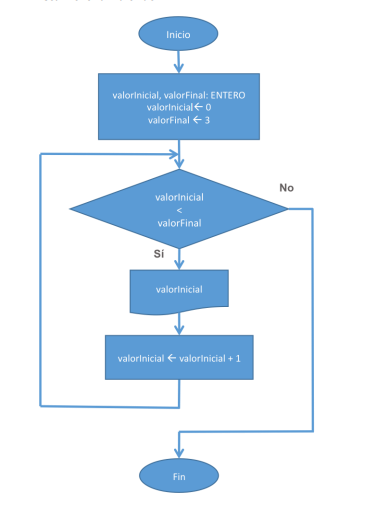
**Iterativa para:**

También, se es requerido que algún fragmento del código se repita mientras se cumple una condición, siendo en este caso, la Iterativa paran en el cual se da una condición, al ser cierta, se dan una serie de instrucciones y se regresa al paso anterior, de ser falso, el programa continúa más adelante.



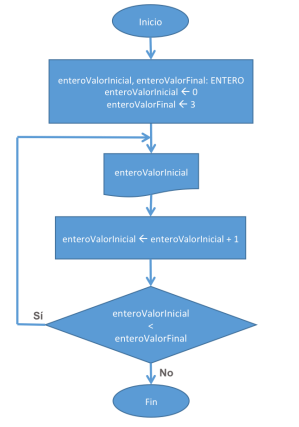
**Iterativa mientras:**

En este tipo de iterativa, una serie de instrucciones se llevarán a cabo mientras se cumpla una condición, hasta el momento en el que se cumpla, en el cual, el programa puede continuar con sus secuencias siguientes. Siendo el siguiente un ejemplo:



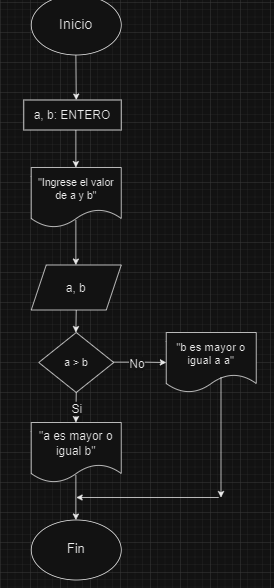
**Iterativa hacer mientras:**

En esta estructura, primero se realizan las instrucciones y luego se verifica la condición, de cumplirse, las secuencias de instrucciones se repetirán, de no pasar, se continuará con el resto del diagrama. Siendo un ejemplo:

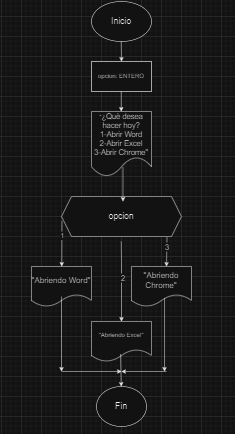


**Ejercicios:**

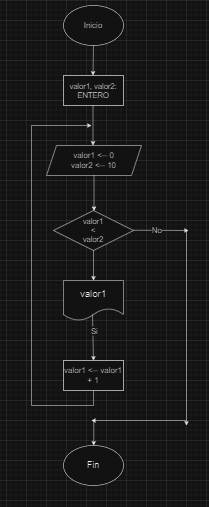
**Ejercicio 1: Determinar si un número es mayor a otro.**

****

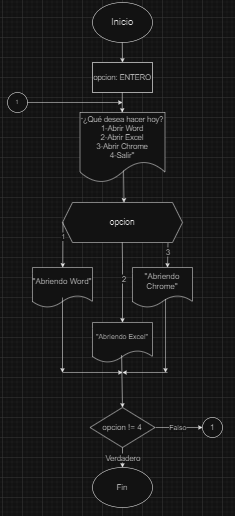
**Ejercicio 2: Hacer un menú con 3 opciones.**

****

**Ejercicio 3: Realizar un programa que cuente del 0 al 10.**

****

**Ejercicio 4: Añadir al menú del ejercicio 3 la opción de salir, de no seleccionarla, se repetirá en bucle**

****

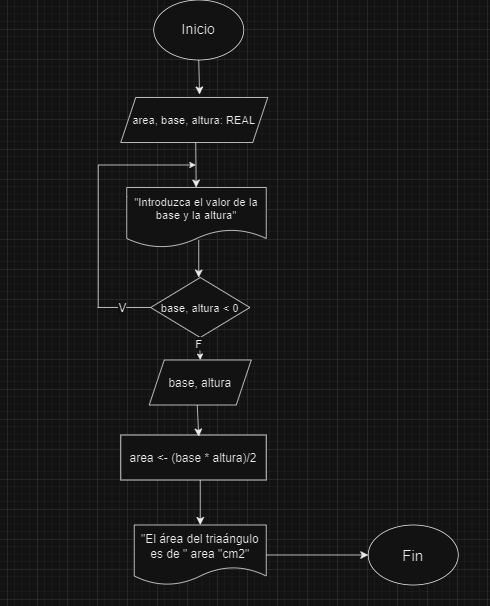
**Tarea:**

**1-Calcule el área de un triángulo.**

Datos de entrada: Base y altura del triángulo.

Datos de salida: Área del triángulo

Restricciones: Los números no pueden ser menores o iguales a 0



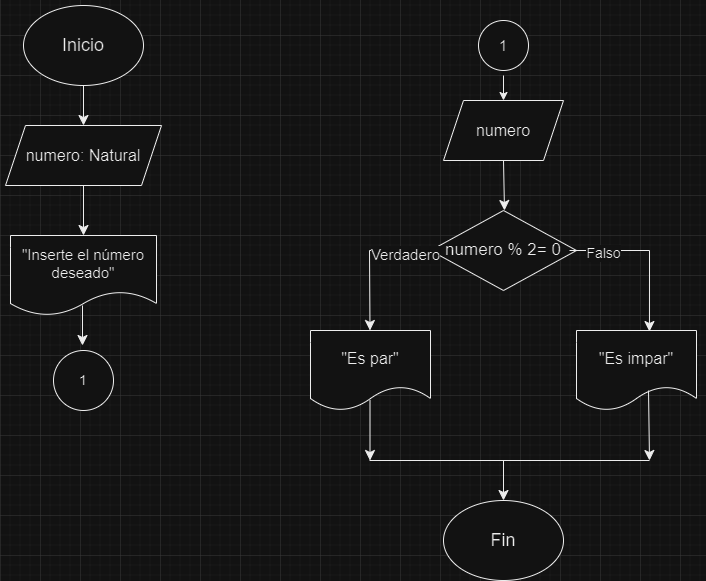
|  |  |  |  |
| --- | --- | --- | --- |
| Iteración | Base | Altura | Salida |
| 1 | 0 | -5 | - |
| 2 | 5 | 4 | 10 |
| 3 | 9 | 3 | 27/2 |

**2-Determine si un número es par (utiliza el %) indicar el número que ingresó, en caso contrario indicar el número y si es impar.**

Datos de entrada: Número real

Datos de salida: Determinar si es par o impar

Restricciones: El número no puede ser igual a 0



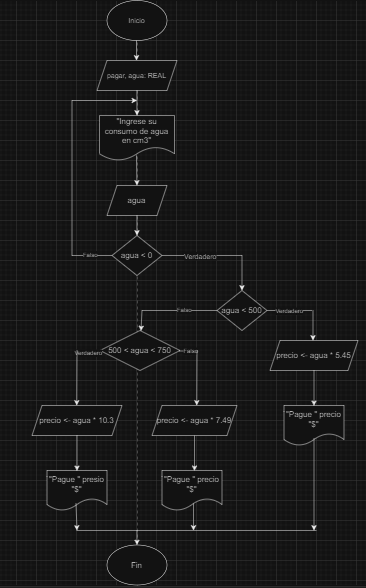
|  |  |  |
| --- | --- | --- |
| Iteración | Número | Salida |
| 1 | 0 | - |
| 2 | 6 | Es par |
| 3 | 19 | Es impar |

**3-Solicitar al usuario los centímetros cúbicos para el pago de agua y debe ser mayor a cero, en caso contrario se le debe volver a pedir el valor, los rangos son: De 1 a 500 centímetros cúbicos pagará 5.45 por centímetro cúbico, de 501 a 750 centímetros cúbicos pagará 7.49 por centímetro cúbico y mayor a 750 centímetros cúbicos pagará 10.3 por centímetro cúbico.**

Datos de entrada: Número de la cantidad de agua consumida

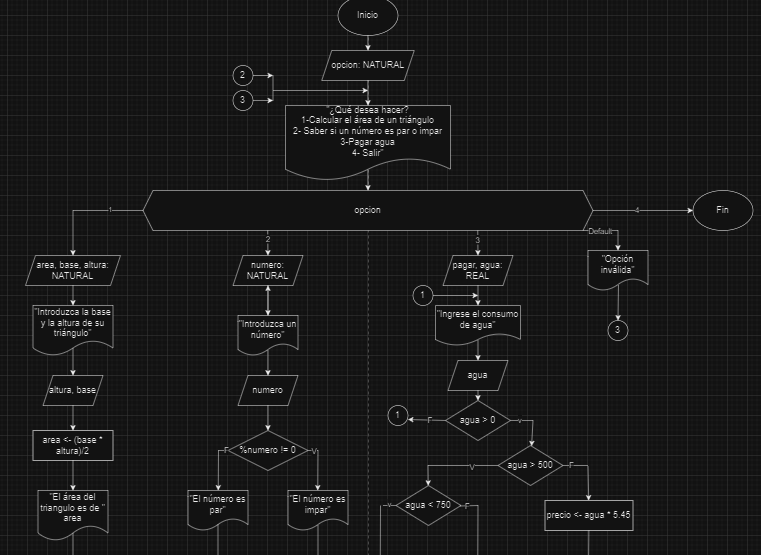
Datos de Salida: Total a pagar

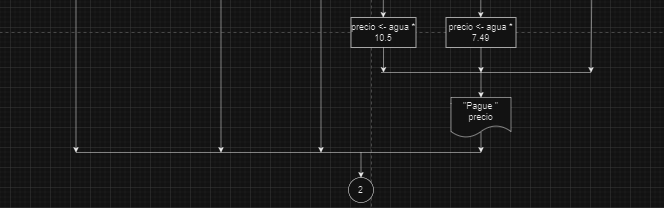
Restricciones: El consumo no puede ser igual o menor a cero

****

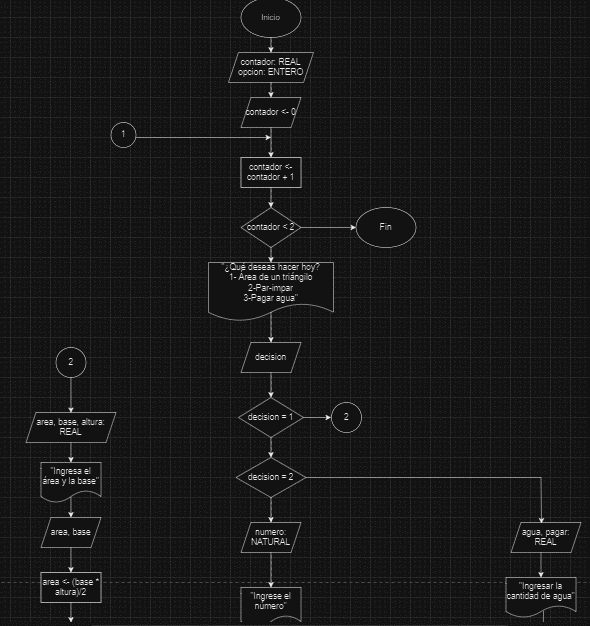
|  |  |  |
| --- | --- | --- |
| Iteración | Cantidad de agua | Total a pagar |
| 1 | -6 | - |
| 2 | 100 | 545$ |
| 3 | 600 | 4,494$ |

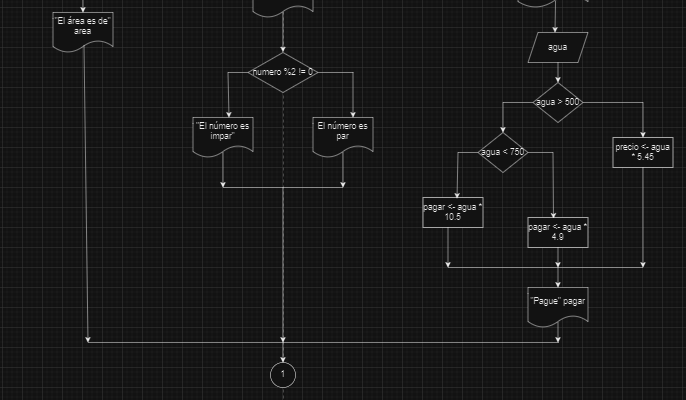
**4.-Genera un menú de los 3 ejercicios anteriores con la estructura CONDICIONAL MÚLTIPLE y HACER MIENTRAS (el usuario no sale del menú hasta que elija la opción 4 que es salir).**





**5.-Genera un menú de los 3 ejercicios anteriores con la estructura condicional anidada, este programa solo se debe repetir 2 veces, utiliza la estructura MIENTRAS o PARA, este programa no necesita la opción de salir.**



****

**Conclusiones:**

El diagrama de flujo presenta ser una herramienta muy útil para poder crear un programa de software, representando de manera gráfica un algoritmo. Así, para poder crear un buen algoritmo es requerido seguir una lista de reglas, siendo así, más fácil para unos utilizar este método, mientras que, a otros, les resultará más práctico el utilizar un algoritmo, siendo este último mi caso

**Bibliografía:**

Estructuras condicionales anidadas. (s. f.). https://www.tutorialesprogramacionya.com/javaya/detalleconcepto.php?codigo=81&

García E. et al. (2022) Manual de Prácticas del laboratorio de Fundamento de programación, recuperado de <http://lcp02.fi-b.unam.mx/>