Parte 1: Sentencia if

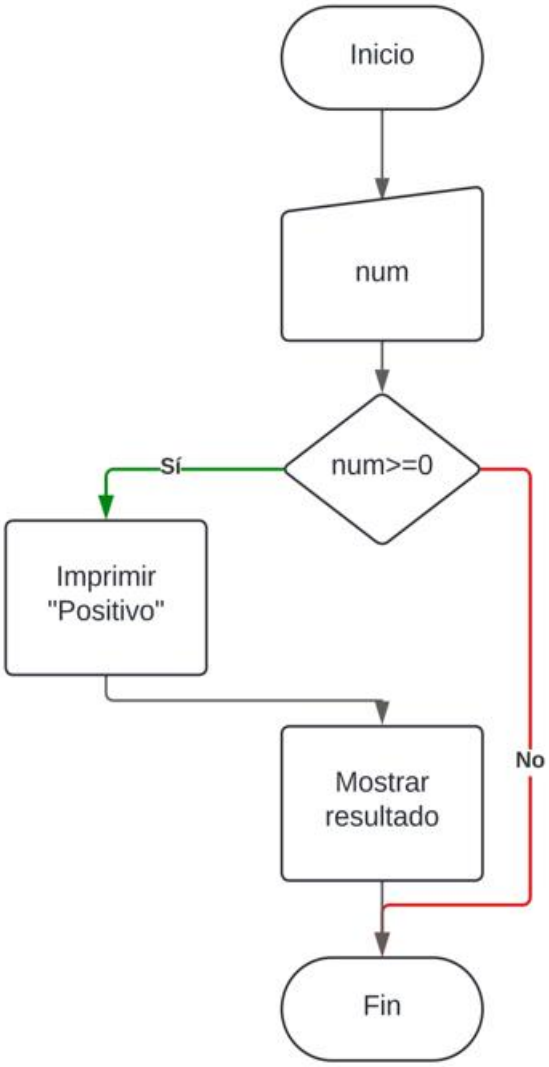
1. Determina si un número es positivo.

- Problema; Dado un número, determinar si es positivo.
- Datos de entrada: Un número dado por el usuario.
- Solución:

    1. Leer número: ingresado por el usuario.
    2. Si el número>0, imprimir “es positivo”
    3. Salida: Mostrar la decisión de la condicional.

- Salida: Mostrar si el número es positivo.

La condicional if es útil para mostrar si el número ingresado es positivo, siempre y cuando la condición se cumpla, de lo contrario no mostrará nada.



2. Verifica si una persona es mayor de edad.

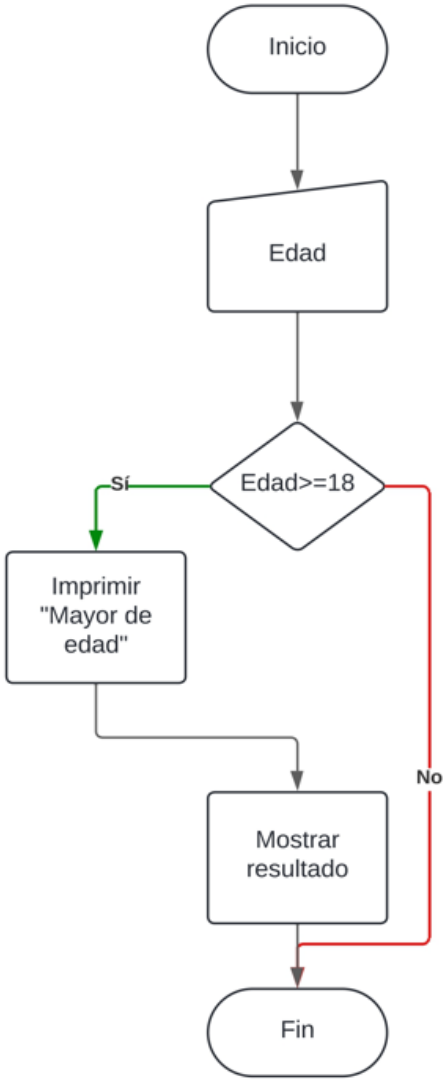
- Problema: Dada la edad de una persona, identificar si es mayor de edad.
- Datos de entrada: Edad de la persona.
- Solución:

    1. Leer número: Ingresado por el usuario.
    2. Aplicar: Si edad>= 18, se imprime “Usted es mayor de edad”.
    3. Mostar salida: resultado de la decisión por la sentencia if.

- Salida: Mostrar si la persona es mayor de edad.

Para este enunciado, la sentencia if es útil y se aplica de la siguiente forma.

Si edad>= 18, se imprime “Usted es mayor de edad”, ya que la sentencia da True y se imprime el resultado.



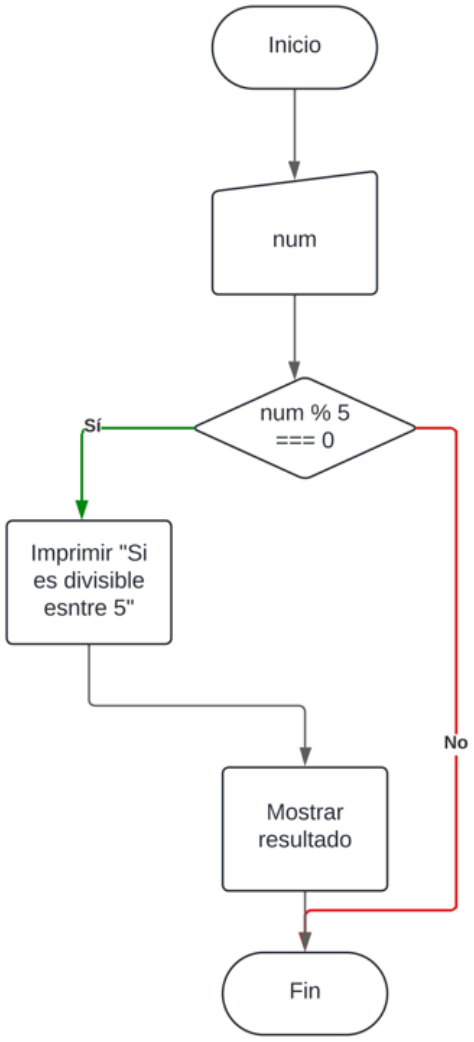
3. Verifica si un número es divisible por 5.

- Problema: Dado un número, verificar si es divisible entre 5.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:

    1. Leer número: ingresado por el usuario.
    2. Aplicar: Si número % 5 === 0, se imprime “El número es divisible entre 5”.
    3. Mostrar salida: El resultado de la sentencia if (decisión).

- Salida: Mostrar si el número ingresado es divisible entre 5.

En este caso, la sentencia if trabaja con el operador % que divide de manera entera un número. Con esto, se aplica: si número % 5 === 0, se imprime “El número es divisible entre 5”, esta pasa porque la decisión da como resultado True y se da la orden de imprimir el texto.



4. Comprueba si un año es bisiesto.

- Problema: Comprueba si un año es bisiesto.
- Datos de entrada: Año
- Solución:

    1. Leer año: ingresado por el usuario.
    2. Si año % 4 = 0, Imprimir “ es bisiesto”
    3. Mostrar Salida: Imprimir el resultado de la decisión.

- Salida: Mostrar si el año bisiesto.

La condicional if nos ayuda a decidir si un año es bisiesto, lo usé porque con ayuda de % (Módulo), si es divisible entre 4, el año ingresado es bisiesto, de lo contrario, la sentencia resulta false y no se impirme nada.

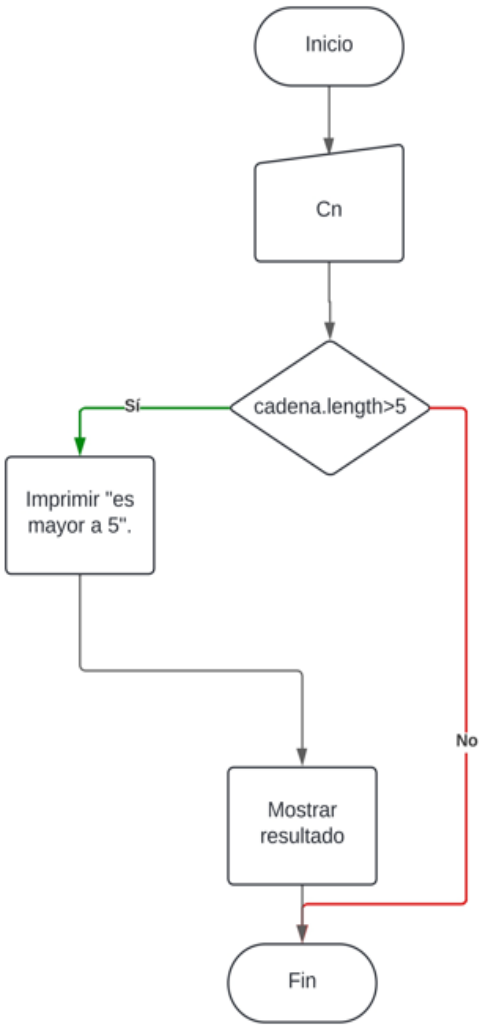
5. Verifica si una cadena tiene más de 5 caracteres.

- Problema: Verifica si una cadena tiene más de 5 caracteres.
- Datos de entrada: Cadena de caracteres dada por el usuario.
- Solución:

    1. Leer cadena: Dada por el usuario.
    2. Aplicar: Cadena de caracteres length = Total de caracteres.
    3. Si total de caracteres>5, Imprimir “es más de 5 caracteres”.
    4. Mostrar salida: Imprimir el resultado de la decisión.

- Resultado: Mostrar el resultado de la decisión.

Además de usar la condicional “if”, use una propiedad llamada “length”, la cual permite contar cuántos caracteres hay en una cadena, si length resulta mayor a 5, la sentencia “if” imprimirá “es más de 5 caracteres” pues la condición se cumplió.



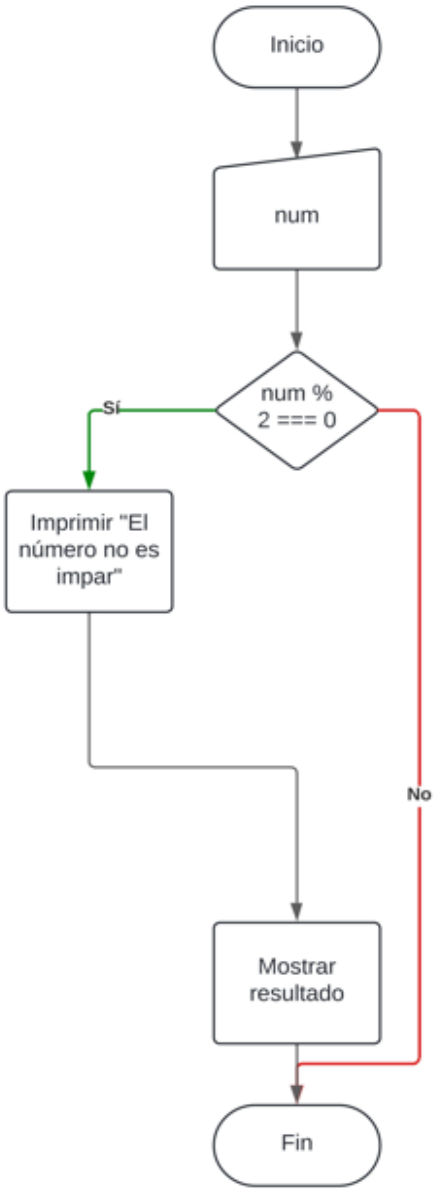
6. Comprueba si un número es impar.

- Problema: Comprueba si un número es impar.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:

    1. Leer el número: El usuario ingresa el número.
    2. Decidir: Si número % 2 === 1, imprimir “es impar”
    3. Mostrar Salida: Imprimir el resultado de la decisión,

- Salida: Mostrar si el número ingresado es par o impar.

La condicional “if” lo usé para decidir si el número es par, esto con la ayuda de % (módulo) el cual divide al número, pero de forma entera; si deja un residuo de 1, es impar y se muestra el mensaje de que es impar.



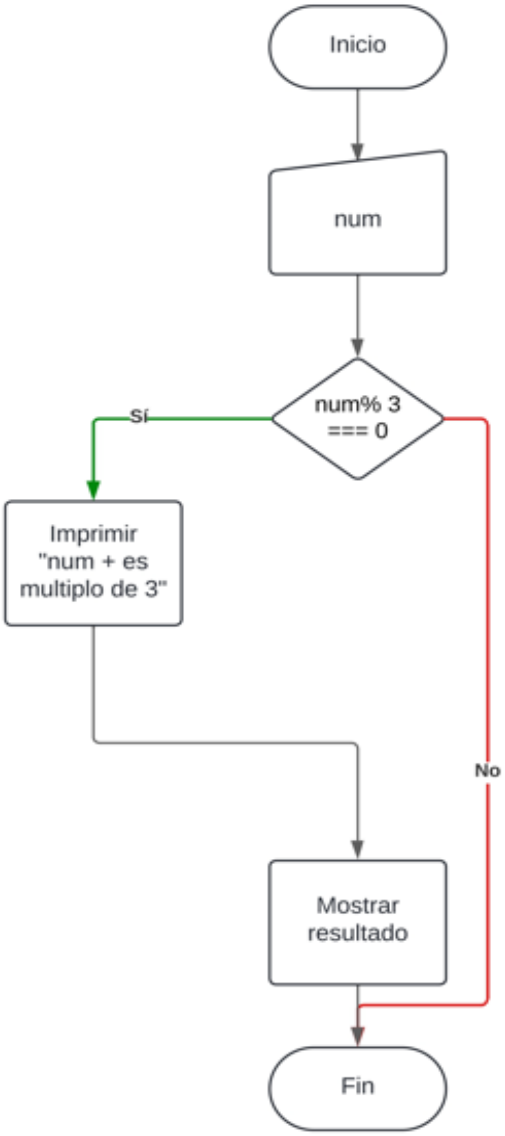
7. Verifica si un número es múltiplo de 3.

- Problema: Dado un número, verificar si es múltiplo de 3.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:

    1. Leer número: Recibir el número dado por el usuario.
    2. Aplicar: si número % 3 === 0, imprimir 'número' + "es múltiplo de 3"
    3. Salida: Mostrar el resultado de la decisión.

- Salida: Mostrar si el número ingresado es múltiplo de 3 o no.

Usé la concisión "if" y la expresión %, primero el módulo divide el número ingresado entre 3, si es múltiplo de 3, es porque el residuo fue 0 e imprime "El número es múltiplo de tres".



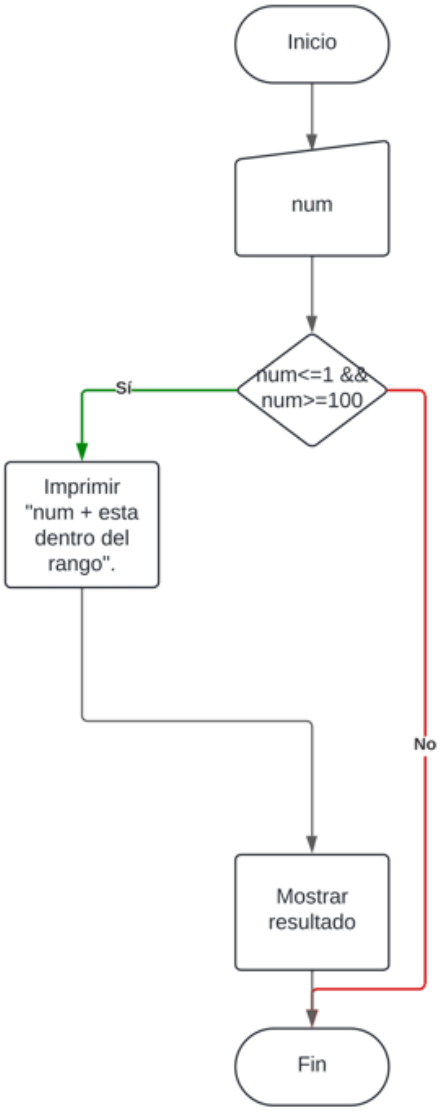
8. Comprueba si un número está entre 1 y 100.

- Dado un número, verificar si se encuentra entre el 1 y el 100.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:

    1. Leer número: Ingresado por el usuario.
    2. Decidir: Si el num<=1 && num>=100, Imprimir: 'número' + "está dentro del rango".
    3. Mostrar salida: Mostrar el resultado de la decisión.

- Salida: Mostrar si el número ingresado está dentro del conjunto o no.

Utilice la sentencia if para saber si el número ingresado está dentro del rango, la condición es que si num<=1 && num>=100, imprima "está dentro del rango".



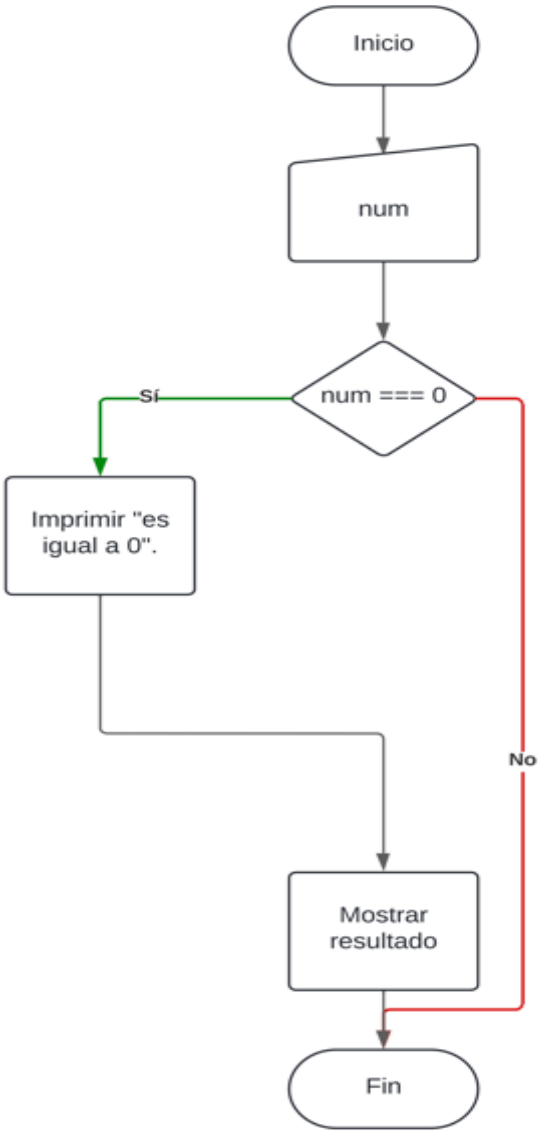
9. Verifica si un número es igual a 0.

- Problema: Dado un número, verificar si es igual a 0.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:

    1. Leer número: ingresado por el usuario.
    2. Aplicar: Si número = 0, imprimir “Es igual a 0”
    3. Salida: Mostrar el resultado de la decisión.

- Salida: Mostrar el resultado si el número ingresado es igual a 0.

Este caso es muy sencillo, con el uso de la sentencia If, se compara el número ingresado si resulta igual a 0, se cumple la condición y se imprime “Es igual a 0”.



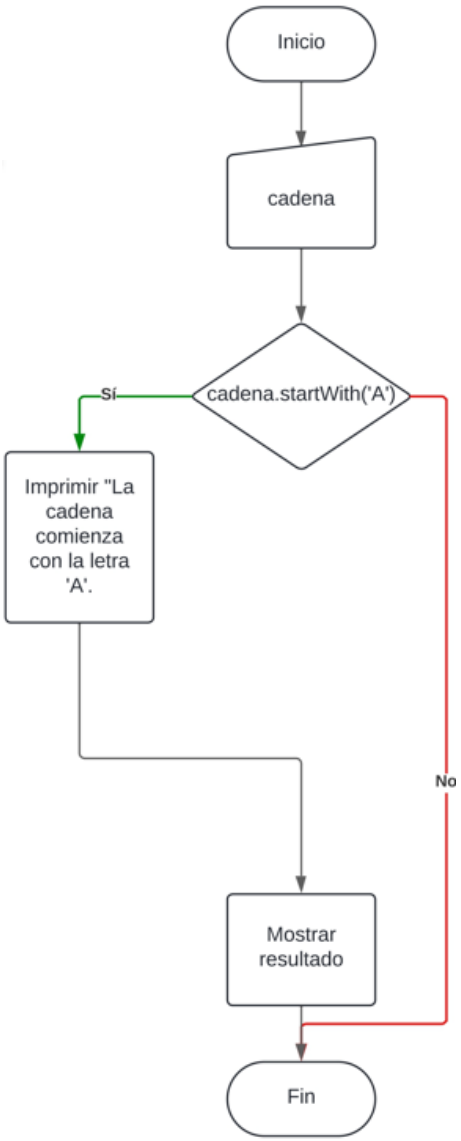
10. Verifica si una cadena comienza con la letra “A”.

- Problema: Dada una cadena de caracteres, verificar si esta comienza con “A”.
- Datos de entrada: Cadena ingresada por el usuario.
- Solución:

    1. Leer cadena: dada por el usuario.
    2. Aplicar: Si cadena.startsWith (‘A’), imprimir “La cadena comienza con la letra ‘A’.”
    3. Salida: Mostrar el resultado de la condicional.

- Salida: Mostrar el resultado si la cadena comienza con A.

Para realizar esta verificación, usé una sentencia if que contenía: cadena.startsWith(‘A’), esta propiedad se encarga de revisar si una cadena de caracteres comienza con un carácter específico, si la propiedad startsWith indica que la cadena empieza con A, “if” dará True y imprimirá “La cadena comienza con la letra ‘A’” .



Parte 2: Sentencia if-else de dos alternativas

1. Verifica si un número es par o impar.

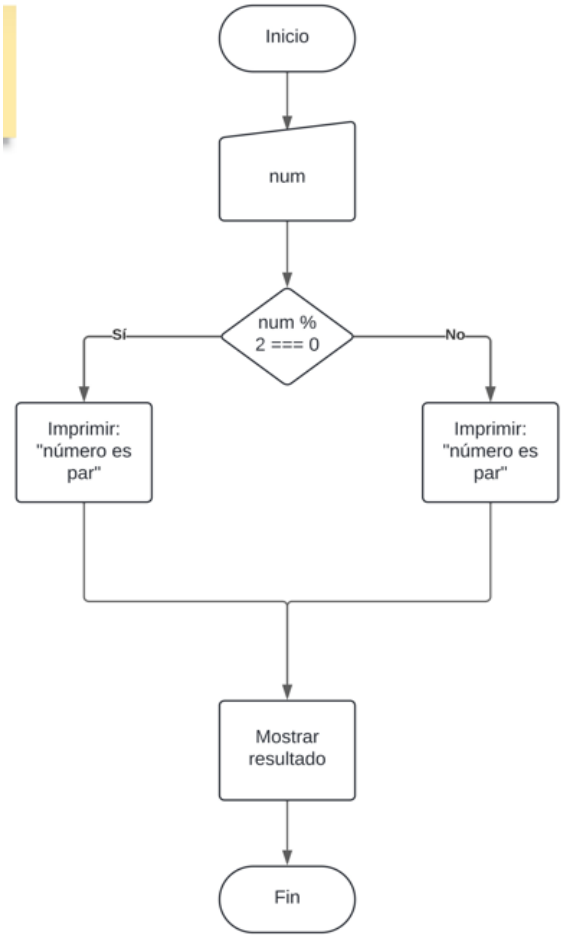
- Problema: Dado un número, verificar si es par o impa
- Datos de entrada: Un número dado por el usuario.
- Solución:

    1. Leer número: dado por el usuario.
    2. Aplicar: Si num % 2 === 0, imprimir ‘num’ + “es par” si no, imprimir ‘num’ + “es impar”

- Salida: Mostrar si el número ingresado es par o impar.

La sentencia if-else nos ayuda a saber si el número ingresado es par o impar.

Si la condicional por medio de módulo de 2 = 0, es true se imprime “es par”, si no, imprime “es impar”.



2. Determina si un número es mayor que 100.

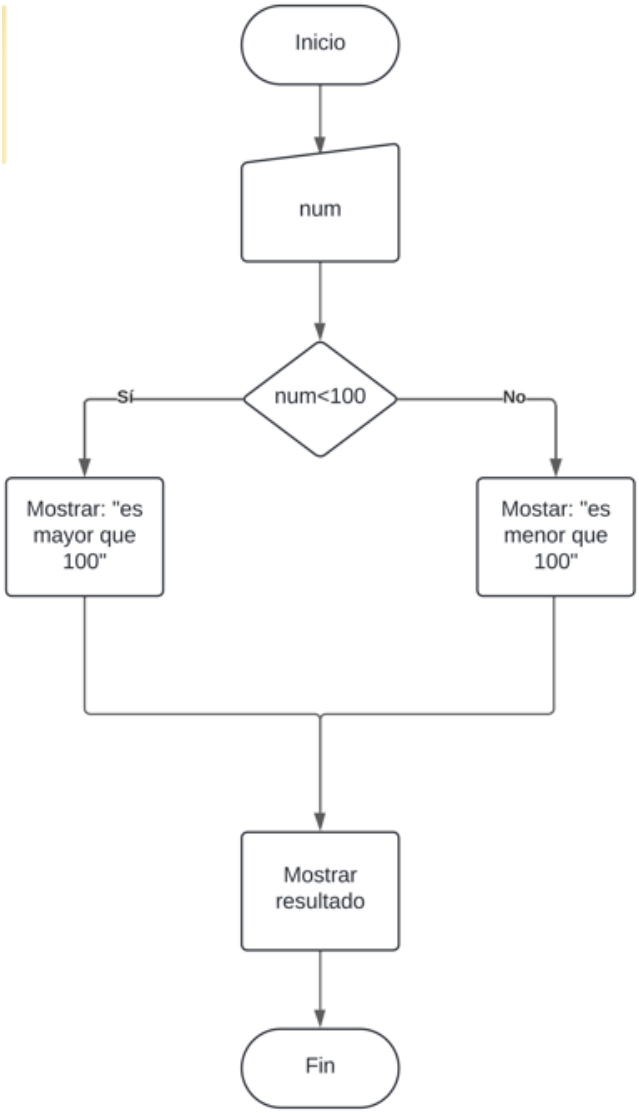
- Problema: Dado un número, comprobar si esa mayor a 100.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:

    1. Leer el número: Ingresado por el usuario.
    2. Si número<100, Imprimir “es mayor que 100”,

    si no imprimir “es menor que 100”.

- Salida: Mostrar si el número ingresado es mayor a 100 o no.

Use la sentencia if-else para decidir si el número ingresado es mayor a 100; si el número ingresado es mayor a 100, resulta en true y muestra “es mayor que 100”, de lo contrario, al resultar false, muestra “es menor que 100”.



3. Verifica si una persona puede votar (mayor o igual a 18 años).

- Problema: Dada la edad de una persona, verificar si puede votar.
- Datos de entradas: Edad de la persona.
- Solución:

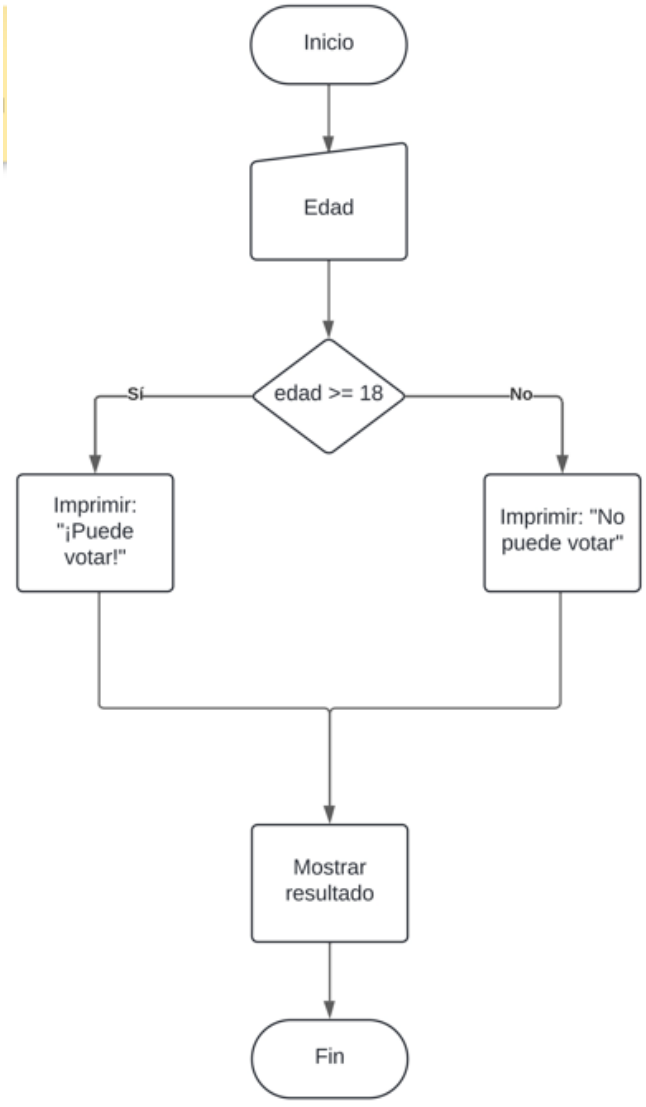
    1. Leer edad: Ingresada por el usuario.
    2. Aplicar: Si edad>=18, imprimir “¡Puede votar!”,

    si no lo es, imprimir “No puede votar.”

- Salida: Mostrar si la persona puede votar o no.

El uso de la sentencia if-else de este enunciado, facilita la decisión de si una persona puede votar o no con la siguiente condicional.

Si edad>=18, imprime “¡Puede votar!”, y que la condición es true, si resulta false, imprime “No puede votar.”



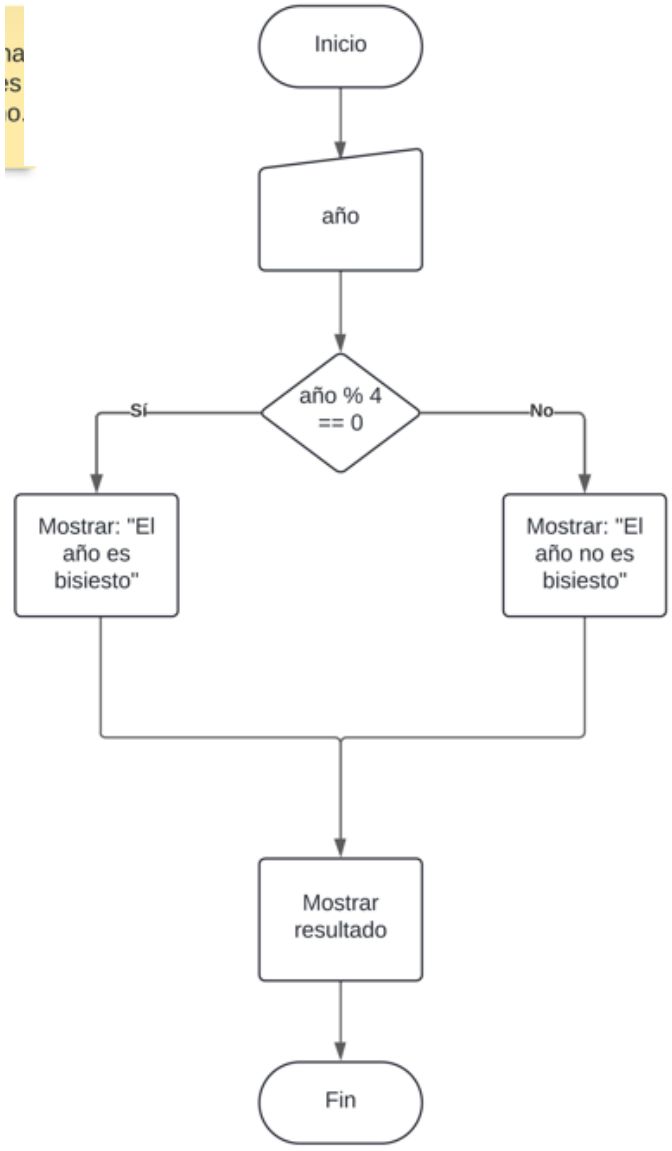
4. Determina si un año es bisiesto o no.

- Problema: Dado un año, verificar si es bisiesto.
- Datos de entrada: Un año ingresado por el usuario.
- Solución:

    1. Leer el año: ingresado por el usuario.
    2. Aplicar: Si año % 4 == 0, se imprime “El año es bisiesto”; de lo contrario, se imprime “El año no es bisiesto”

- Salida: Mostrar si el año ingresado es bisiesto o no.

Módulo (%), divide el año ingresado entre 4 con la característica que siempre lo hará entero, es decir, si el número es par dara 0 si no, dara 1 qué es el residual. La sentencia if-else, toma en cuenta las 2 respuestas, si recibe 0, la sentencia dará true e imprimirá “El año es bisiesto”, si no, es false y mostrara “El año no es bisiesto”



5. Verifica si una cadena tiene menos de 10 caracteres.

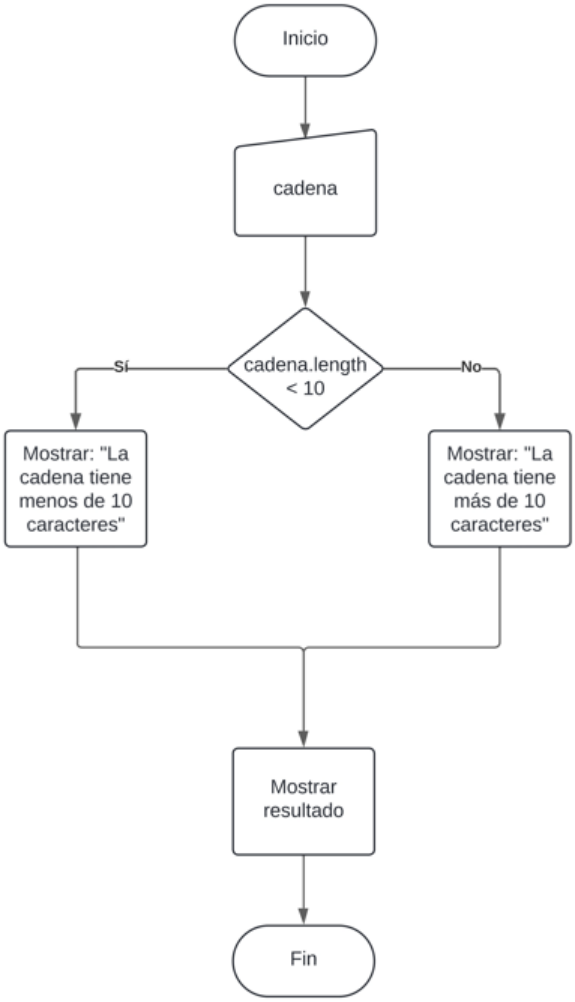
- Problema: Dada una cadena, comprobar si tiene menos de 10 caracteres.
- Datos de entrada: Una cadena ingresada por el usuario.
- Solución:

    1. Leer la cadena: ingresada por el usuario.
    2. Aplicar cadena.length < 10.
    3. Si la propiedad da menos de 10, se mostrará “La cadena tiene menos de 10 caracteres”.

    Sí no, se mostrará “La cadena tiene más de 10 caracteres”.

- Salida: Mostrar si la cadena ingresada tiene menos de 10 caracteres a no.

Con la ayuda de la propiedad .length, si esta da como resultado menos de 10, la sentencia if, sera true e imprime “La cadena tiene menos de 10 caracteres”, sino, la sentencia es false y muestra “La cadena tiene más de 10 caracteres”.



6. Determina si un número es positivo o negativo.

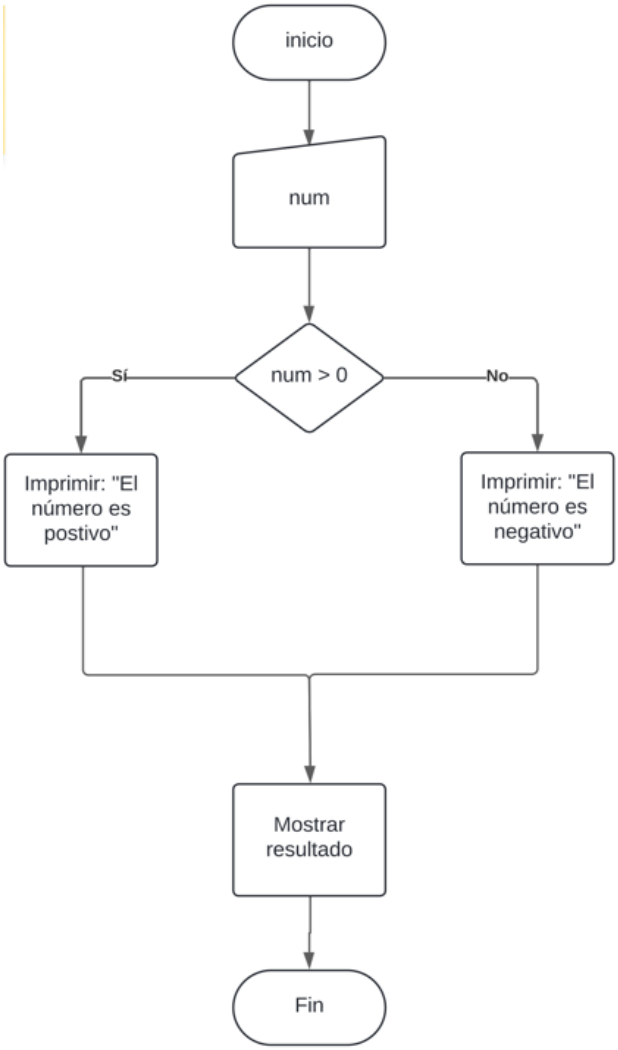
- Problema: Dado un número, determinar si es positivo o no.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:

    1. Leer número: ingresado por el usuario.
    2. Aplicar: si número>0, la sentencia da como resultado true y muestra “El número es positivo”,

    si no, se muestra “El número es negativo”

- Salida: Mostrar si el número ingresado es positivo o negativo.

Este caso es muy sencillo, la sentencia if comprueba si el número ingresado es positivo, si es así entonces el resultado es true e imprime “El número es positivo”, si no se cumple esa condición, el resultado es false e imprime “El número es negativo”



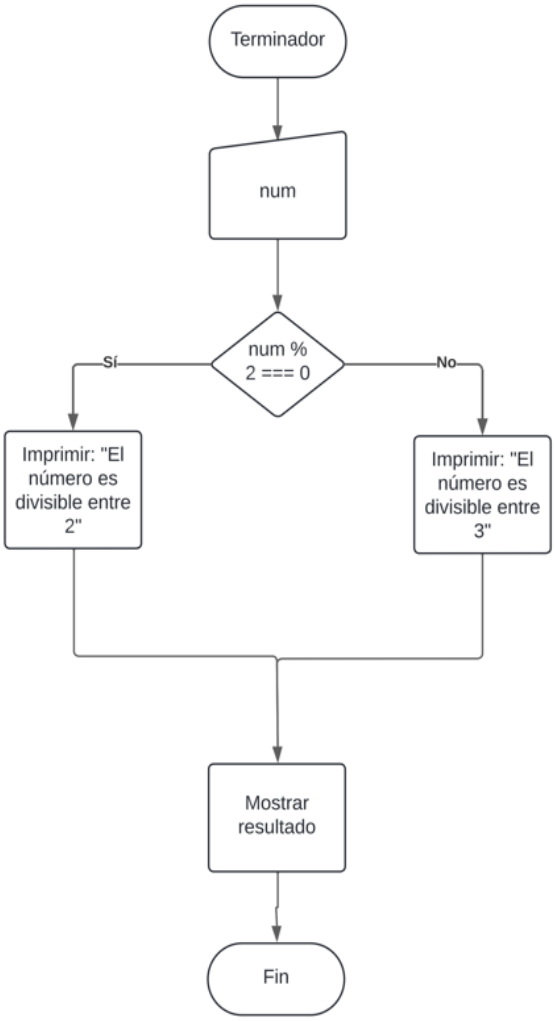
7. Verifica si un número es divisible por 2 o por 3.

- Problema: Dado un número, determinar si es divisible por 2 o por 3.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:

    1. Leer número: ingresado por el usuario.
    2. Aplicar: Si número % 2 === 0, el número es divisible por 2 y se muestra “El número es divisible entre 2”; si no se cumple esa condición, número % 3 === 0, el número es divisible por 3 y se imprime “El número es divisible entre 3”.

- Salida: Mostrar si el número es divisible por 2 o por 3.

Con el resultado de la división de módulo (%) de 2, la sentencia if-else, muestras si el número es divisible entre 2 (true si se cumple la condición), o si es divisible entre 3 (false, entra en esta opción pues la primera no se cumplió).



8. Comprueba si una letra es vocal o consonante.

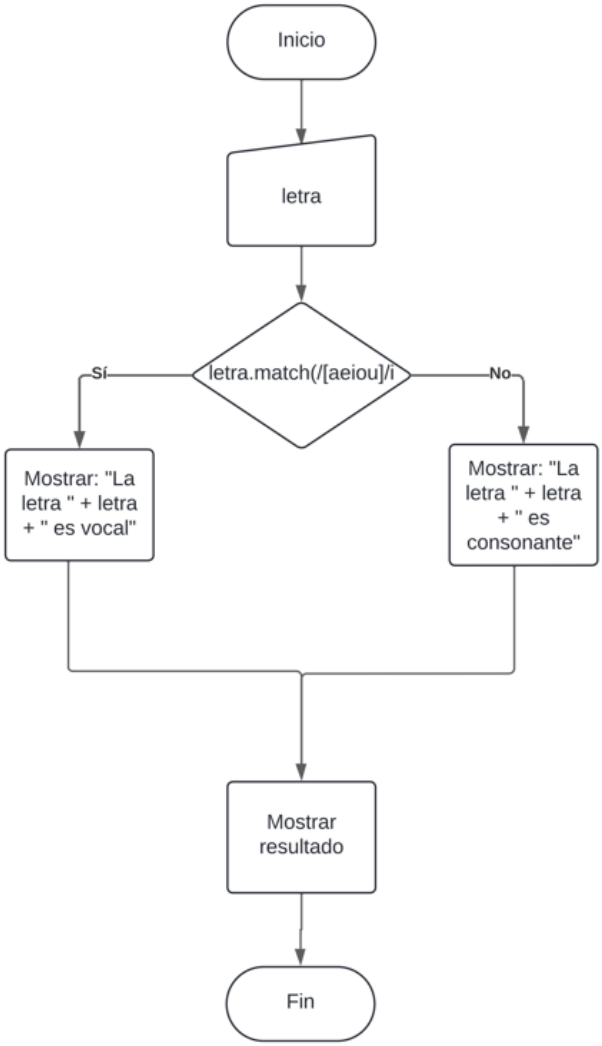
- Problema: Dada una letra, determinar si es una vocal o una consonante.
- Datos de entrada: Una letra ingresada por el usuario.
- Solución:

    1. Leer letra: ingresada por el usuario.
    2. Aplicar:
    3. Si la letra.match(/[aeiou]/i), el resultado es true y se muestra “La letra es una vocal”.

    Si no, el resultado es false y se muestra “La letra es una consonante”.

- Salida: Mostrar si la letra es vocal o consonante.

Este algoritmo usa letra.match(/[aeiou]/i) para verificar si la letra ingresada es una vocal (true, entrar en la primera opción), y else para cualquier otra letra, que sería una consonante (false, por lo que entra en la segunda opción).



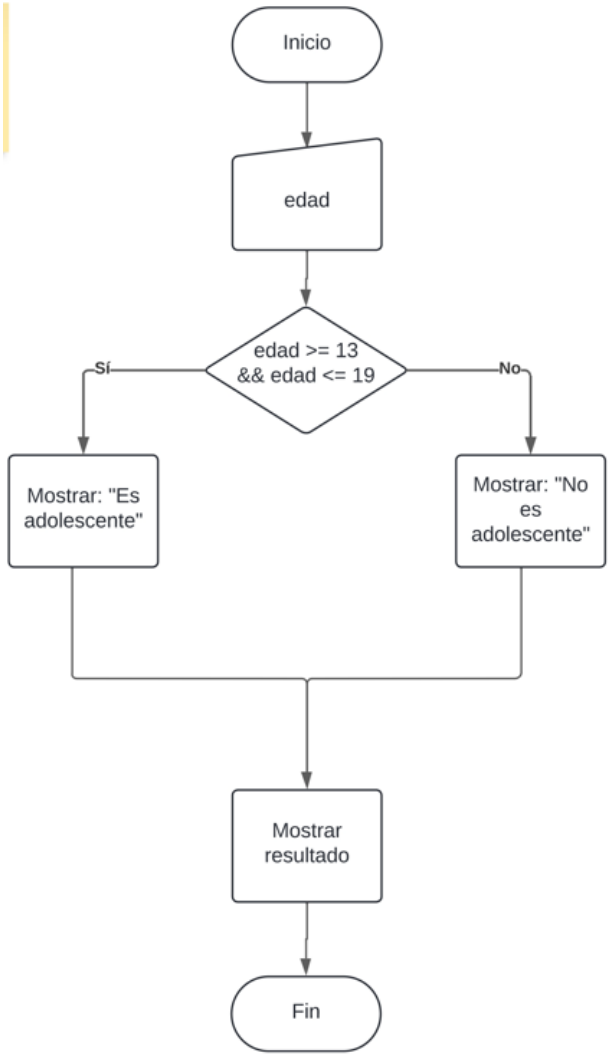
9. Verifica si una persona es adolescente (entre 13 y 19 años).

- Problema: Dada la edad de una persona, verificar si es un adolescente.
- Datos de entrada: Edad de la persona ingresada por el usuario.
- Solución:

    1. Leer edad: ingresada por el usuario.
    2. Aplicar: Si edad >= 13 && edad <= 19, el resultado es true y se muestra “Es adolescente”.
    3. Si no, el resultado es false y se muestra “No es adolescente”.

- Salida: Mostrar si la persona es adolescente o no.

En este enunciado, usé if para comprobar si una persona está dentro del rango indicado, si la condición se cumple, se mostrará “Es adolescente”, de lo contrario será false y pasará a else y mostrará “No es adolescente”.



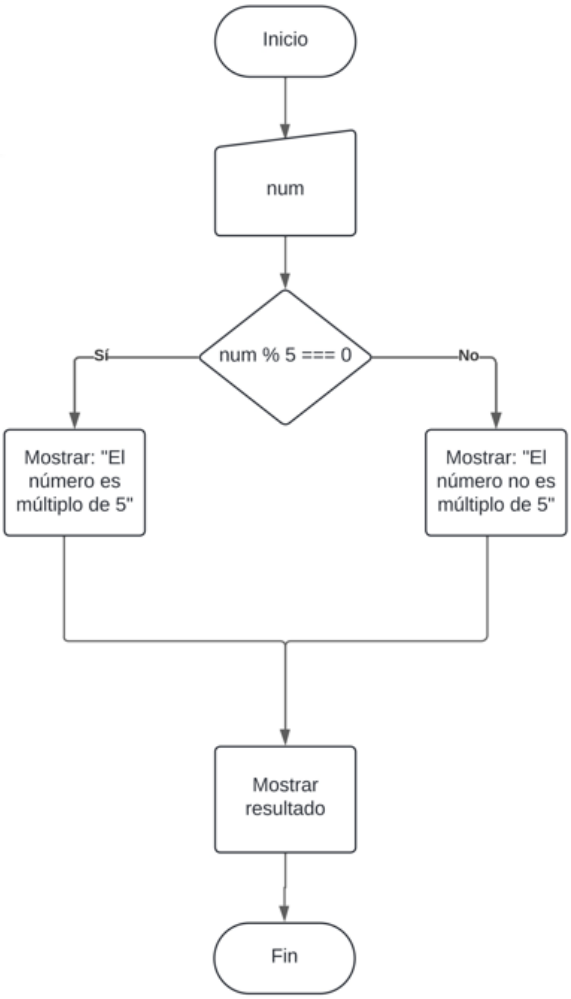
10. Comprueba si un número es múltiplo de 5.

- Problema: Dado un número, determinar si es múltiplo de 5.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:

    1. Leer número: ingresado por el usuario.
    2. Aplicar: Si número % 5 === 0, el resultado es true y se muestra “El número es múltiplo de 5”.
    3. Si no, el resultado es false y se muestra “El número no es múltiplo de 5”.

- Salida: Mostrar si el número es múltiplo de 5 o no.

En este planteamiento, la sentencia if tiene la tarea de verificar si el número ingresado es múltiplo de 5 (true) e imprime “El número es múltiplo de 5”, de lo contrario entrará en la opción de else en imprimirá “El número no es múltiplo de 5”.



Parte 3: Sentencias if-else anidadas

1. Determina si un número es positivo, negativo o cero.

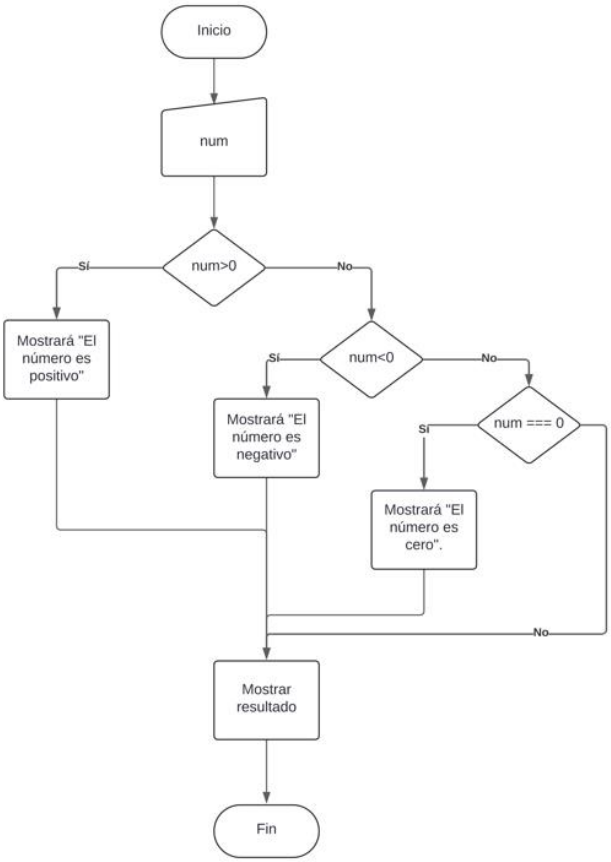
- Problema: Dado un número, determinar si es positivo, negativo o cero.
- Datos de entrada: Un número dado por el usuario.
- Solución:

  1. Leer número: ingresado por el usuario.
  2. Aplicar: Si número>0, mostrará “El número es positivo”

  sí no, número<0, y mostrará “El número es negativo” sí no, número y se mostrará “El número es cero”.

- Salida: Mostrar si el número ingresado es positivo, negativo o cero.

La sentencia if-else anidada, permite elegir entre las 3 opciones que se nos dan; para este planteamiento, si se simple la primera condición, se muestra “’número’ es positivo”, si no, se cumple la segunda y se muestra “’número’ es negativo” y si no se cumple ninguna, entra en la tercera opción y se muestra “es cero”.



2. Verifica si una persona puede votar, conducir o ambas.

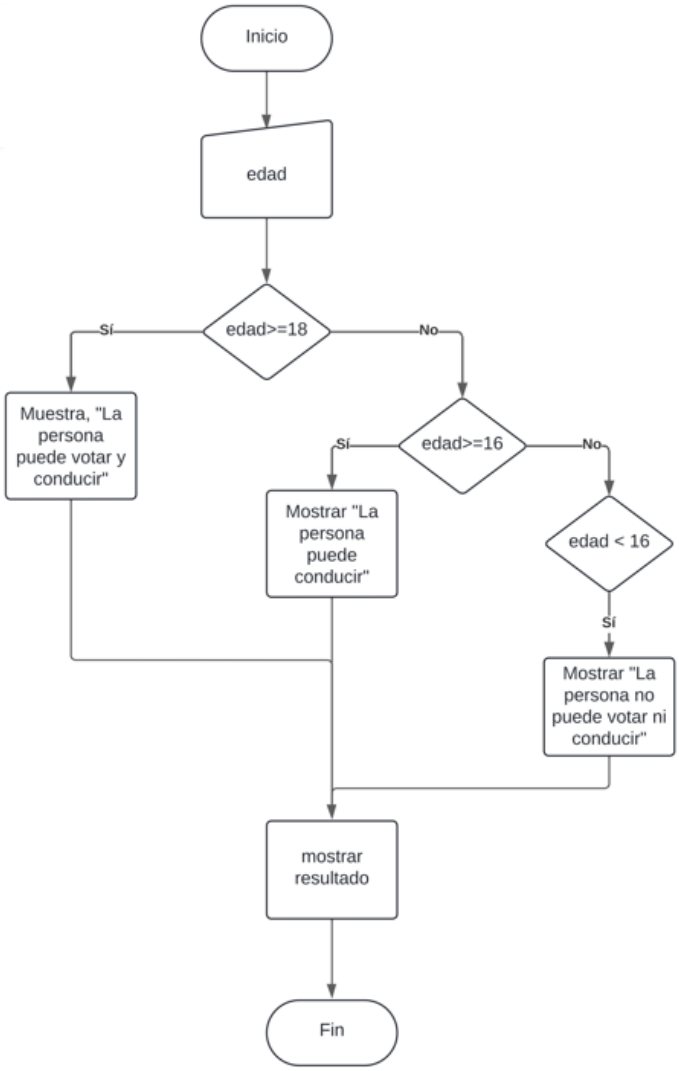
- Problema: Verificar si una persona puede votar, conducir o ambas.
- Datos de entrada: edad de la persona.
- Solución:

    1. Leer edad: ingresada por la persona.
    2. Aplicar: Si edad >= 18, imprimir “La persona puede votar y conducir”.

Si no, si edad >= 16, imprimir “La persona puede conducir”.

Si no, edad < 16 e imprimir “La persona no puede votar ni conducir”.

Para este enunciado, use la sentencia if-esle anidadas, las cuales me permiten elegir entre más opciones; en este caso, si edad>=18, se imprimirá “La persona puede votar”, si no, se pasará a la segunda opción que es sí no edad>=16, imprimir “La persona puede conducir” y si no es ninguna de las anteriores, va a imprimir “La persona puede votar y conducir”.



3. Verifica si un número es divisible por 2, 3 o ambos.

- Problema: Dado un número, verificar si es divisible entre 2, 3 o ambos.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:

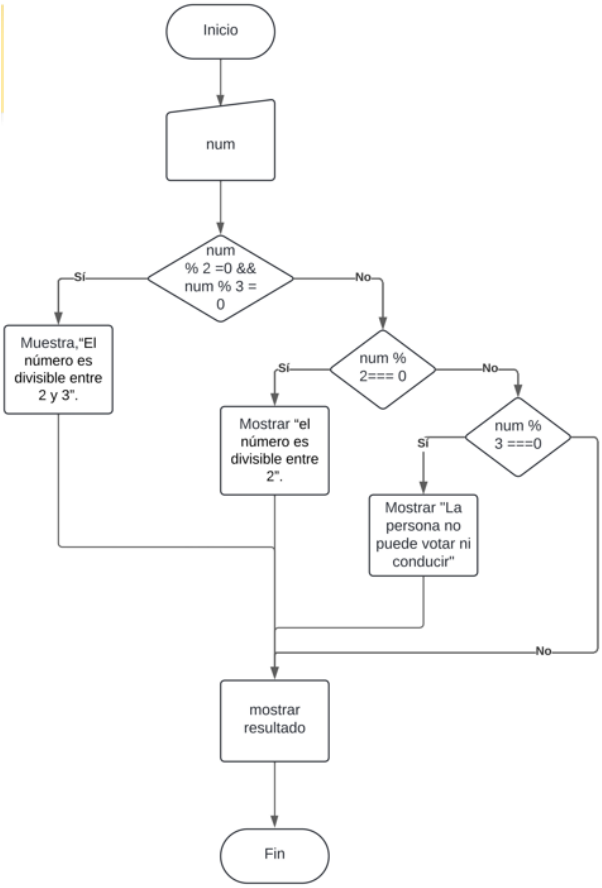
1. Leer el número ingresado.
2. Aplicar: Si número % 2 =0 && numero % 3 = 0, imprimirá “El número es divisible entre 2 y 3”.

   si no, numero % 2= 0, imprimir “el número es divisible entre 2”.
   si no, numero % 3 = 0, e imprime “El número es divisible entre 3”.

- Salida: Mostrar si el número ingresado es divisible entre 2, 3 o ambos.

Con el uso de la sentencia if-else, se logra conocer si el número ingresado es divisible entre los números dados.

Si número % 2 =0 && numero % 3 = 0 número % 2 = 0, se muestra que el número se puede dividir entre las dos opciones. Si número % 2= 0, el número se puede dividir entre 2 y muestra “El número es divisible entre 2”, si no, pasa a la opción siguiente y muestra que el número es divisible entre 3, pero si en ninguna se cumple, no se imprime nada.



4. Verifica el estado del clima (frío, templado o caliente).

- Problema: Verificar el estado del clima.
- Datos de entrada: Temperatura ingresada por el usuario.
- Solución:

    1. Leer temperatura: ingresada por el usuario.
    2. Aplicar: si temperatura >= -67 && temp <= 10, se muestra “El clima es frío”

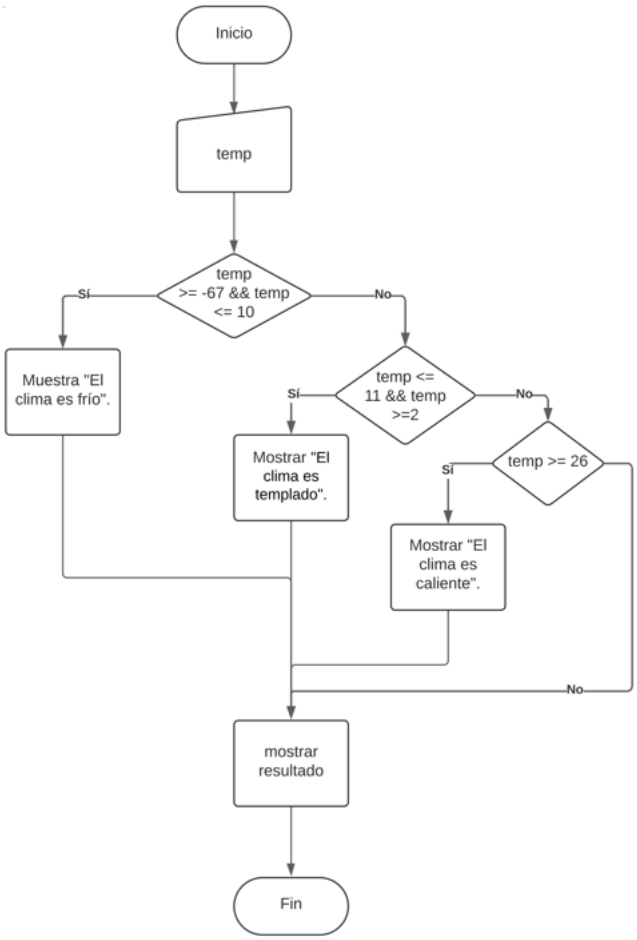
    Si no, la temp >= 11 && temp <=25, y muestra “El clima es templado”

    Si no, la temp >= 26, y ,mostrará “EL clima es caliente”.

- Salida: Mostrar si el clima es frío, templado o caliente.

El uso de la sentencia if-else anidada permite decidir entre tres opciones.

Para este enunciado, se debe decidir si el estado de clima es frío, templado o caliente. Si la temperatura >= -67 y <= 10, se muestra “El clima es frío”; si no, la temperatura >= 11 y <=25, y muestra “El clima es templado”, Si no, la temp >= 26, y ,mostrará “El clima es caliente”. Se establece rangos, pues el propósito es que se introduzca la temperatura y se refleje es estado.



5. Determina si una cadena tiene entre 5 y 10 caracteres.

- Problema: Dada una cadena, verificar si tiene entre 5 y 10 caracteres.
- Datos de entrada: Una cadena ingresada por el usuario.
- Solución:

    1. Leer cadena: ingresada por el usuario.
    2. Aplicar: cadena.length <= 10 && cadena.length >= 5, mostrar “La cadena tiene entre 5 y 10 caracteres”

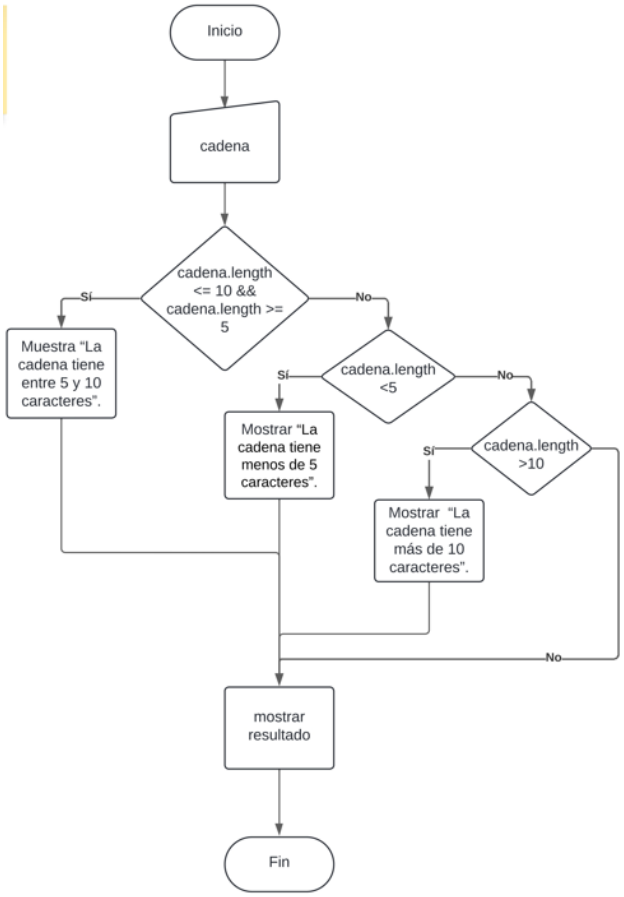
    si no, cadena.length <5, mostrar “La cadena tiene menos de 5 caracteres”.

    si no, cadena.length >10, mostrar “La cadena tiene más de 10 caracteres”

- Salida: Mostrar si la cadena ingresada tiene manos, está en el rango o tiene más caracteres de lo indicado.

En este caso se pide trabajar con una cadena de caracteres, entonces a parte de usar la sentencia if-else, se usa la propiedad .length que se encarga de contar los caracteres.

Se aplica, Si cadena.length <= 10 & & cadena.length >= 5, se imprime “La cadena tiene entre 5 y 10 caracteres”. Si no se cumple la primera opción, cadena.length <=5, se imprime “La cadena tiene menos de 5 caracteres” y si no se cumple la segunda opción, cadena.length >= 10, se imprime “La cadena tiene más de 10 caracteres”, y termina el proceso.



6. Verifica si un número está entre 0 y 50, entre 51 y 100, o más de 100.

- Problema: Dado un número, verificar si un número está entre 0 y 50, entre 51 y 100, o más de 100.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:

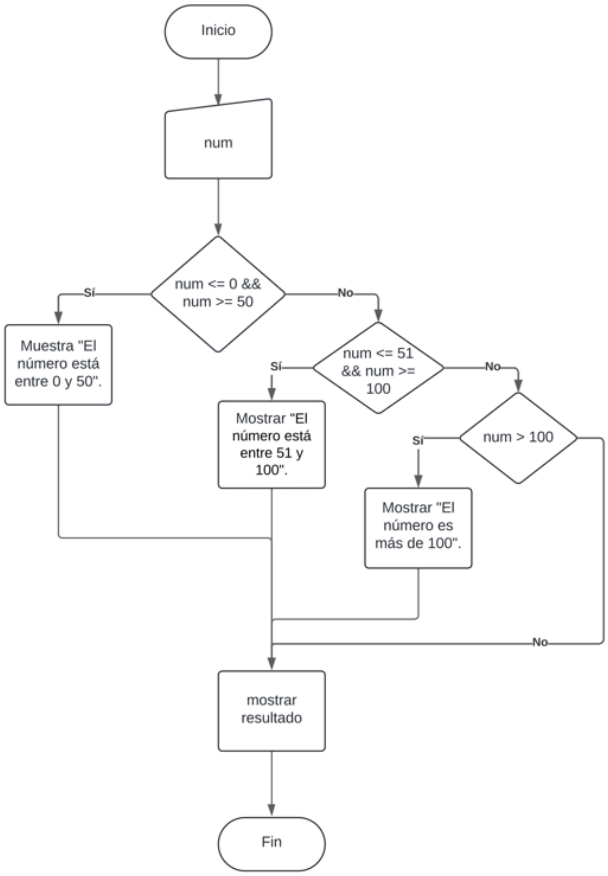
    1. Leer cadena: ingresada por el usuario.
    2. Aplicar: Si número<=0 && numero>=50, y se imprime “El número está entre 0 y 50”

    Si no, número<=51 && numero>=100, e imprime “El número está entre 51 y 100”

    si no, número < 100, e imprime “El número es más de 100”

- Salida: mostrar si la cadena ingresada está entre 0 y 50, entre 51 y 100, o más de 100 caracteres.

El enunciado nos da la instrucción de trabajar con rangos, entonces, si número<=0 && numero>=50, se muestra “El número está entre 0 y 50”. Si no se cumple la primera opción, entonces número<=51 && numero>=100, se muestra “El número está entre 51 y 100” y si no se cumplen la segunda opción, se aplica número < 100 y se muestra “El número es más de 100”.



7. Verifica si una letra es vocal o consonante, y si es mayúscula o minúscula.

- Problema: Dada una letra, verificar si es minúscula, mayúscula.
- Datos de entrada: Una letra ingresada por el usuario.
- Solución:

    1. Leer carácter (letra): Ingresada por el usuario.
    2. Aplicar: Si letra.match(/[AEIOU]/), Imprimir “La letra es una vocal y es mayúscula”

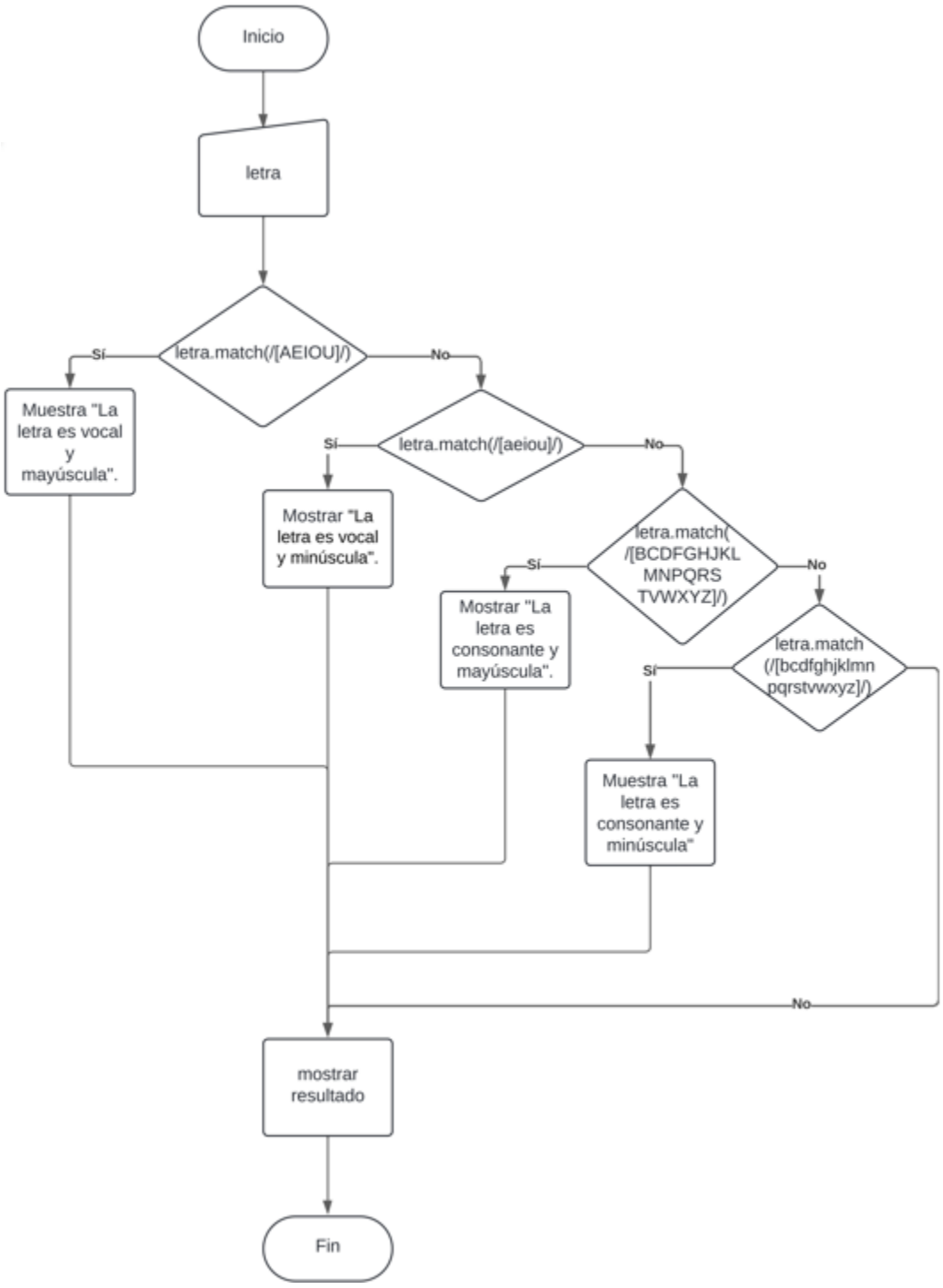
    Si no, letra.match(/[aeiou]), y se imprime “La letra es una vocal minúscula”.

    Si no, letra.match(/[BCDFGHJKLMNÑPQRSTVXZWY]/), se imprime “La letra es una consonante mayuscula”

    Si no, letra.match(/bcdfghjklmnñpqrstvwxz/) y se imprime “La letra es una consonante mayuscula”

- Salida:Mostrar si la letra ingresada es vocal o consonante, y si es mayúscula o minúscula.

Para este enunciado, se usa la propiedad .match para buscar coincidencias. Si letra.match(/[AEIOU, imprimir “La letra es una vocal y es mayúscula”. Si no se cumple, aplicar letra.match(/[aeiou]/) e imprimir “La letra es una vocal minúscula”. Si no se cumple, aplicar letra.match(/[BCDFGHJKLMN PQRSTVXZWY]/) y imprimir “La letra es una consonante mayúscula”. Si tampoco se cumple, aplicar letra.match(/bcdfghjklmnñpqrstvwxz/) y imprimir “La letra es una consonante mayúscula”. Termina el proceso.


Inicio
letra
letra.match(/[AEIOU]/)
Sí
No
Muestra "La letra es vocal y mayúscula".
letra.match(/[aeiou]/)
Sí
No
Mostrar "La letra es vocal y minúscula".
letra.match(/[BCDFGHJKL MNPQRS TVWXYZ]/)
Sí
No
Mostrar "La letra es consonante y mayúscula".
letra.match (/[bcdfghjklmn pqrstvwxyz]/)
Sí
No
Muestra "La letra es consonante y minúscula"
mostrar resultado
Fin

8. Verifica si un número es múltiplo de 2, 3 o ninguno.

- Problema: Dado un número, verificar si un número es múltiplo de 2, 3 o ninguno.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:

    1. Leer número: ingresado por el usuario.
    2. Aplicar: si número % 2 === 0, mostrar “El número es múltiplo entre 2”.

    Si no, número % 3 === 0, “El número es múltiplo entre 2”.

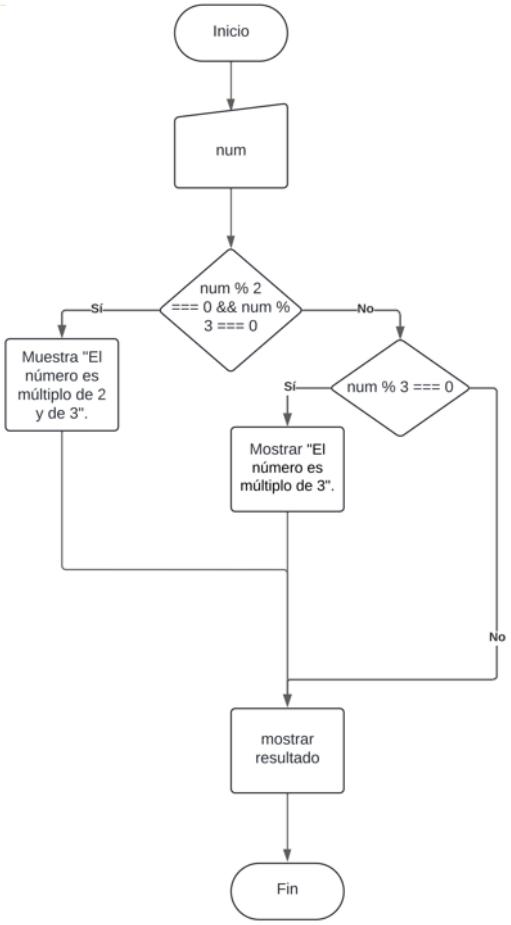
    Si no, se mostrará, “el número no es divisible entre 2 ni de 3”.

- Salida: Mostrar si el es múltiplo de 2, 3 o ninguno.

La sentencia if-else junto con la expresión (%).

Con esto aplicamos: si número % 2 === 0, mostrar “El número es múltiplo entre 2”. Si no se cumple, se aplica: número % 3 === 0, “El número es múltiplo entre 2”.

Y no se cumple ninguna condición anterior, se muestra “el número no es divisible entre 2 ni de 3” y termina el proceso.



9. Verifica el estado de una batería (bajo, medio, alto).

- Problema: Verificar el estado de una batería.
- Datos de entrada: Un porcentaje, dado por el usuario.
- Solución:

1. Leer el porcentaje: dado por el usuario.
2. Aplicar: Si porcentaje >0 & & porcentaje <=20, se muestra “Batería baja”.

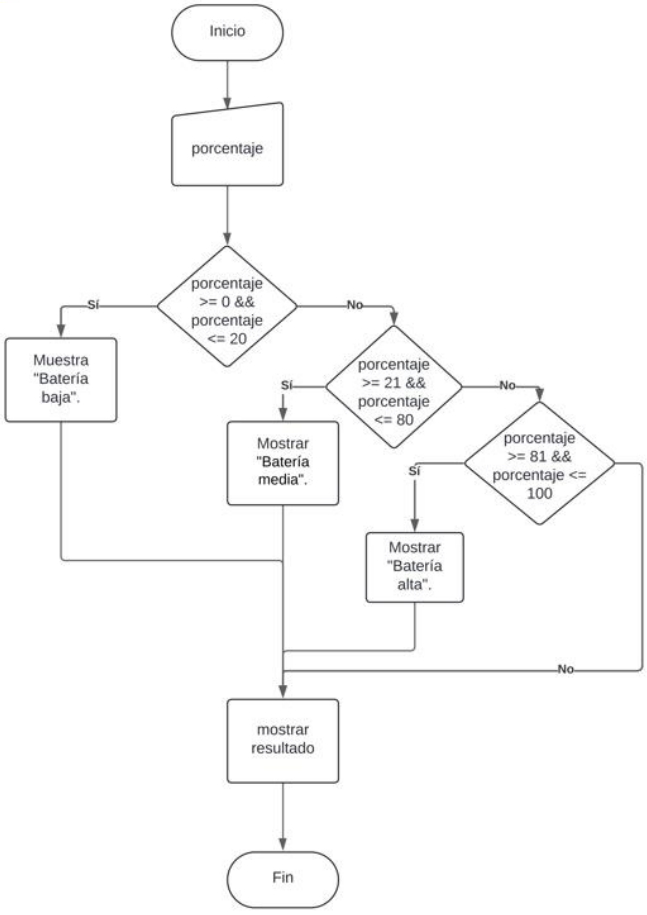
   Sí no, porcentaje >= 21 & & porcentaje <=80, se muestra “Batería media”.

   Sí no, porcentaje >=81 & & porcentaje <=100, se muestra “Batería alta”.

- Salida: Mostrar si el estado de la batería es bajo, medio o alto.

Al analizar el enunciado, se entiende que hay que trabajar con rangos aunque no lo diga textualmente.

Se aplica una sentencia if-else anidada: Si porcentaje >0 & & porcentaje <=20, se muestra “Batería baja”. Al no cumplirse, se aplica: porcentaje >= 21 & & porcentaje <=80, se muestra “Batería media”. Y si tampoco llega a cumplirse, por último se aplica: porcentaje >=81 & & porcentaje <=100, se muestra “Batería alta” y termina el proceso.



10. Determina si un número es positivo y divisible por 5.

- Problema: Dada un número, determinar si un número es positivo y divisible por 5.
- Datos de entrada: Un número dado por el usuario.
- Solución:

1. Leer número: Ingresado por el usuario.
2. Aplicar: Si número<=0 & & número % 5 === 0, imprimir “El número es positivo y divisible por 5”.

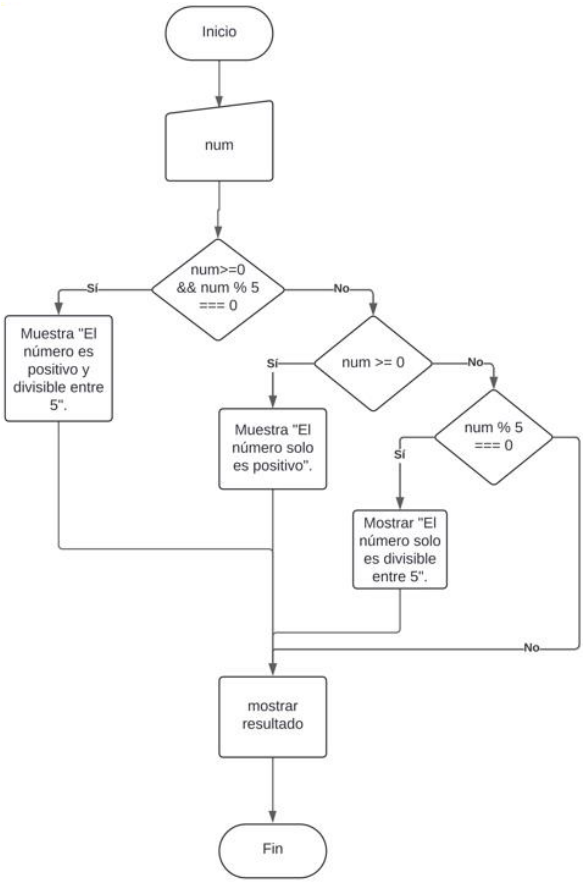
Sí no, número<=0, imprimir “El número solo es positivo.”

Si no, número % 5 === 0, “El número solo es divisible entre 5”.

- Salida: Mostrar si el número ingresado es positivo y divisible por 5.

En este enunciado se nos pide que un dato que tenga dos características, que sea positivo y sea divisible entre 5.

Sabiendo eso se aplica (sentencia if-else anidada): número<=0 & & número % 5 === 0, y se imprime “El número es positivo y divisible por 5”. Si no se cumple, se aplica: número<=0, imprimir “El número solo es positivo”. y si tampoco se llega a cumplir, se aplica: número % 5 === 0, “El número solo es divisible entre 5”, y se termina el proceso.



Parte 4: Sentencia switch-case

1. Determina el día de la semana a partir de un número.

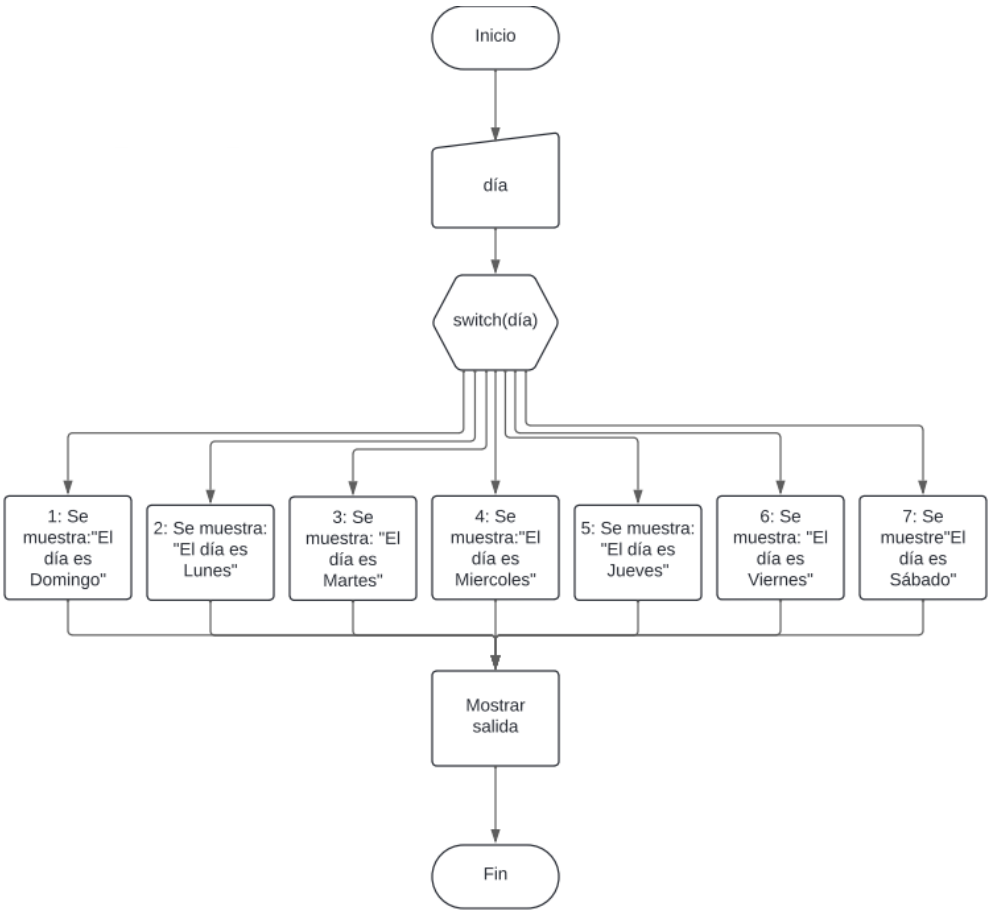
- Problema: Dado un número, determinar el día que le corresponde.
- Datos de entrada: un número dentro del rango de 1 - 7, dado por el usuario.
- Solución:

    1. Leer número: Ingresado por el usuario.
    2. Aplicar: Si el usuario ingresa 2, se imprime “Lunes” y así sucesivamente hasta el 7, pues no hay más días de la semana.

- Salida: Mostrar el día correspondiente al número ingresado.

En este planteamiento, el uso de sentencia switch-case, permite elegir entre varias opciones, en este caso, se pide relacionar un número a un día de la semana.

Se aplica: el usuario ingresa un número, switch-case busca entre las opciones que tiene y cuando encuentra un número igual al ingresado, imprime el día correspondiente y termina el proceso.



2. Verifica si una letra es vocal (solo considera minúsculas).

- Problema: Dada una letra, verificar si es vocal y minúscula.
- Datos de entrada: Una letra dada por el usuario.
- Solución:

    1. Leer letra: ingresada por el usuario.
    2. Aplicar: Si el usuario ingresa e, se imprime “es vocal” y así sucesivamente hasta el u pues solo son las vocales, si se pone otra cosa, no será válido.

- Salida: Mostrar si la letra ingresada es bocal y minúscula.

En este caso, se nos pide que, al ingresar una vocal minúscula, se imprima que “es vocal”, switch-case se encarga de encontrar una coincidencia entre los casos que se le dio con anterioridad, si no los encuentra, imprime “letra no válida” y termina el proceso.

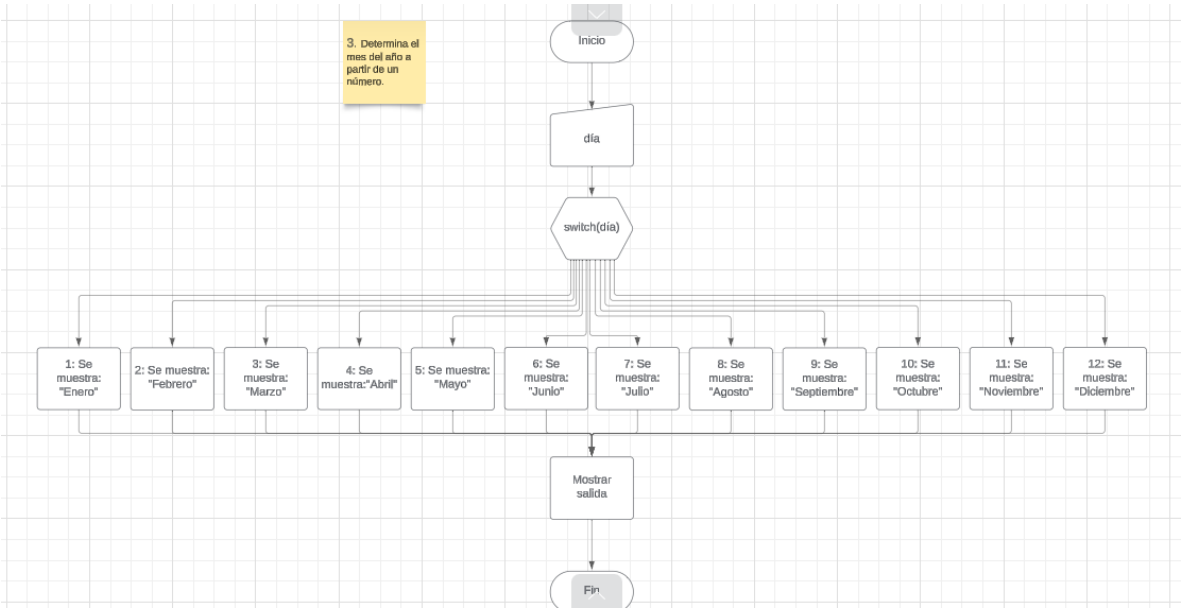
3. Determina el mes del año a partir de un número.

- Problema: Dado un número, verificar a qué mes corresponde.
- Datos de entrada: Un número dado por el usuario.
- Solución:

    1. Leer el número: Dado por el usuario.
    2. Aplicar: Si el usuario ingresa 1, se muestra “Enero” y así sucesivamente hasta el 12, porque solo hay 12 meses en el año.

- Salida: Mostrar el mes que le corresponde al número ingresado.

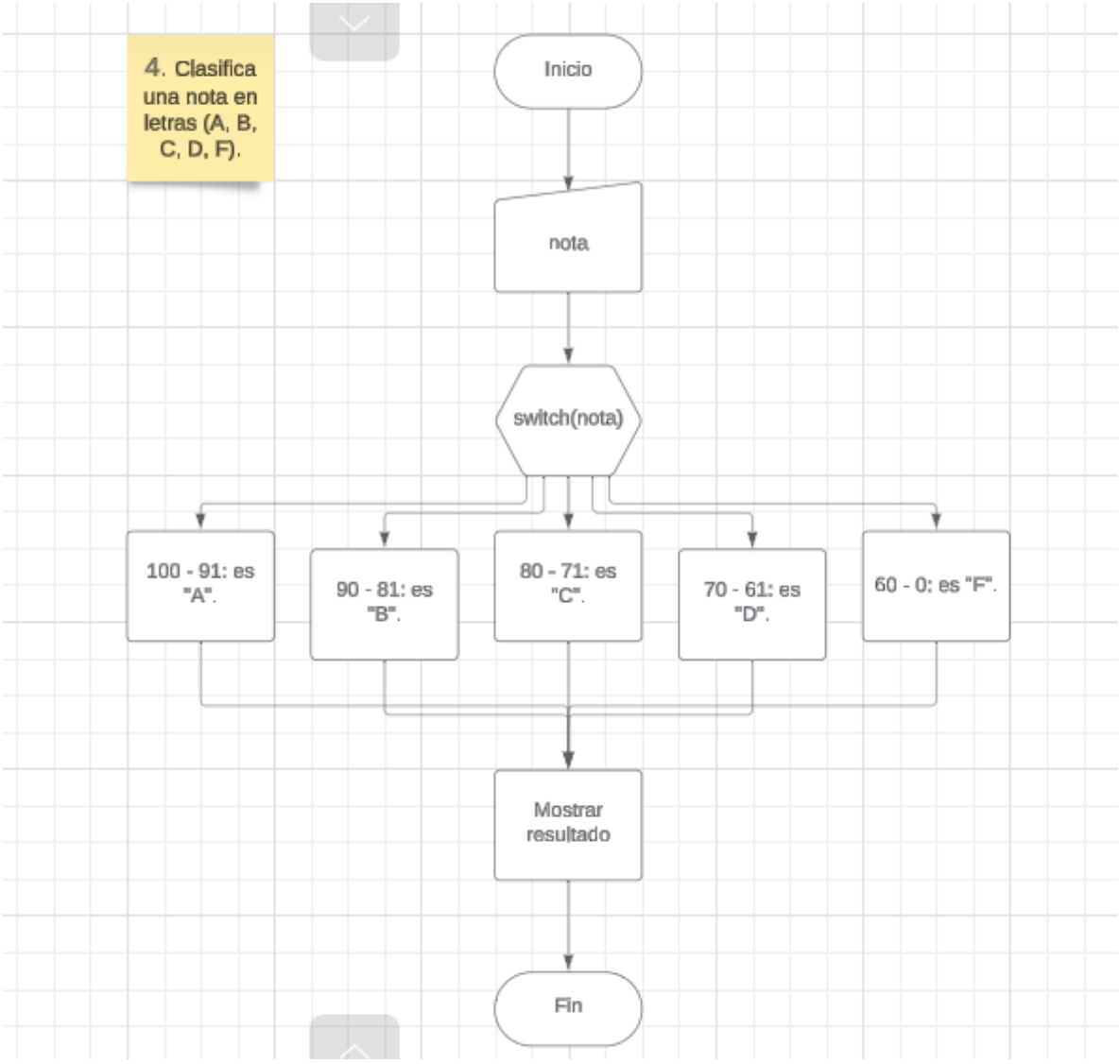
Con la ayuda de switch case, se puede asignar valores a varias opciones, entonces ingresar un número, la sentencia buscará uno que coincida con una de las opciones en imprimirá el mensaje correspondiente y termina en proceso.



4. Clasifica una nota en letras (A, B, C, D, F).

- Problema: Clasificar las notas por letras.
- Datos de entrada: Nota ingresada por el usuario
- Solución:
  1. Leer nota: ingresada por el usuario.
  2. Aplicar: Si el usuario ingresa: 10; se muestra: A. Este proceso se repite hasta la F, pues la nota más baja.
- Salida: Mostrar la letra correspondiente a la nota ingresada.

En este planteamiento se nos pide que relacionemos una nota con una letra. Con la sentencia switch case, podemos indicar el numero y la letra y así cuando se ingrese la nota, esta ultima se muestra.



5. Verifica si una cadena es “Hola”, “Adiós” o “Desconocido”.

- Problema: Dada una cadena, verificar si es “Hola”, “Adiós” o “Desconocido”.
- Datos de entrada: Una cadena ingresada por el usuario.
- Solución:
  1. Leer cadena: ingresada por el usuario.
  2. Aplicar: Si la cadena es “Hola”, imprimir “La cadena es ‘Hola’”.
     Si la cadena es “Adiós”, imprimir “La cadena es ‘Adiós’”.
     Si la cadena no es “Hola” ni “Adiós”, imprimir “La cadena es ‘Desconocido’”.
- Salida: Mostrar si la cadena ingresada es una de las palabras dadas en un principio.

Par este anunciado, solo se nos pide tres opciones; entonces se aplica: Si el usuario ingresa “hola” la cual es una cadena, switch case buscara la opción que coincida y mostrara el texto indicado, en este caso es “Es: hola” y así con las demás palabras y si no es ninguna, mostrara que no es una palabra valida y terminara el proceso.

6. Determina si una cadena es “Sí” o “No”.

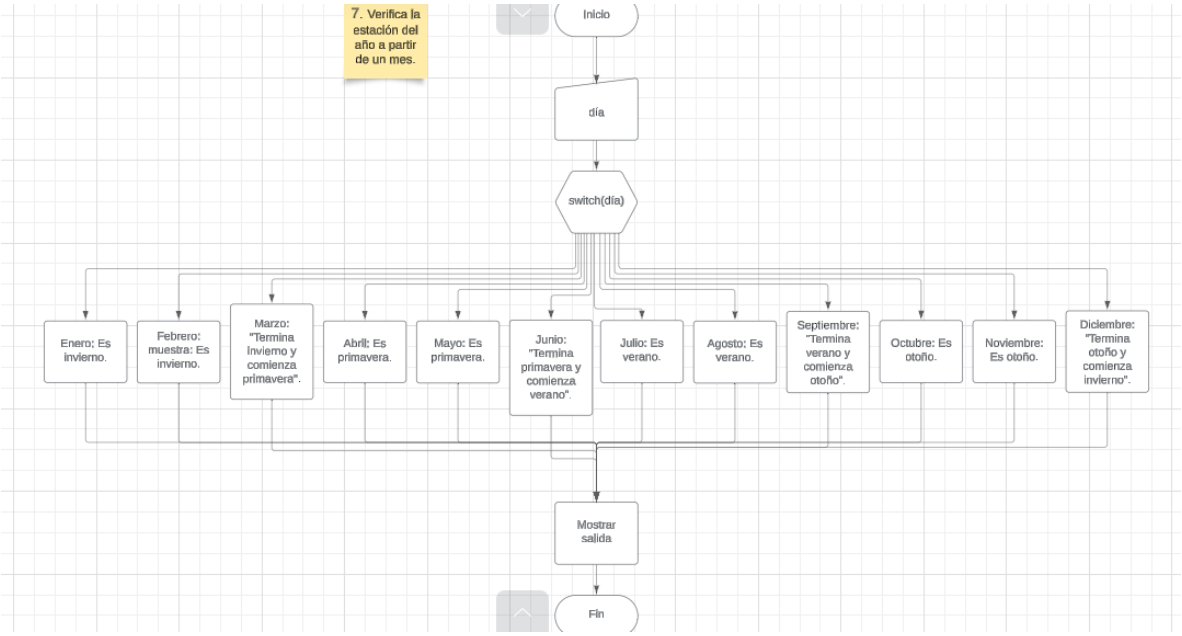
- Problema: Dada una cadena, determinar si es “Sí” o “No”.
- Datos de entrada: Una cadena ingresada por el usuario.
- Solución:
  1. Leer cadena: ingresada por el usuario.
  2. Aplicar: Si el usuario ingresa “Sí”, se muestra: “La cadena es: Sí”.
     Si se ingresa “No”, se mostrará: “La cadena es: No”
- Salida: Mostrar cual es la cadena ingresada.

Cuando se ingrese la cadena “Sí”, esta va a coincidir con la primera opción, mostrando: “La cadena es: Sí”; si se ingresa “No”, va a coincidir con la segunda opción y se mostrara: “La cadena es: No”. Si no coincide con ninguna, Se mostrará: “Cadena no valida.”

7. Verifica la estación del año a partir de un mes.

- Problema: Dado un mes, determinar que estación del año le corresponde.
- Datos de entrada: Un mes ingresado por el usuario.
- Solución:
  1. Leer el mes: dado por el usuario.
  2. Aplicar: Si el usuario ingresas marzo, se mostrará que es primavera, y así con los demás meses (cada estación abarca 3 meses).
- Salida: Mostrar la estación que corresponde al mes ingresado.

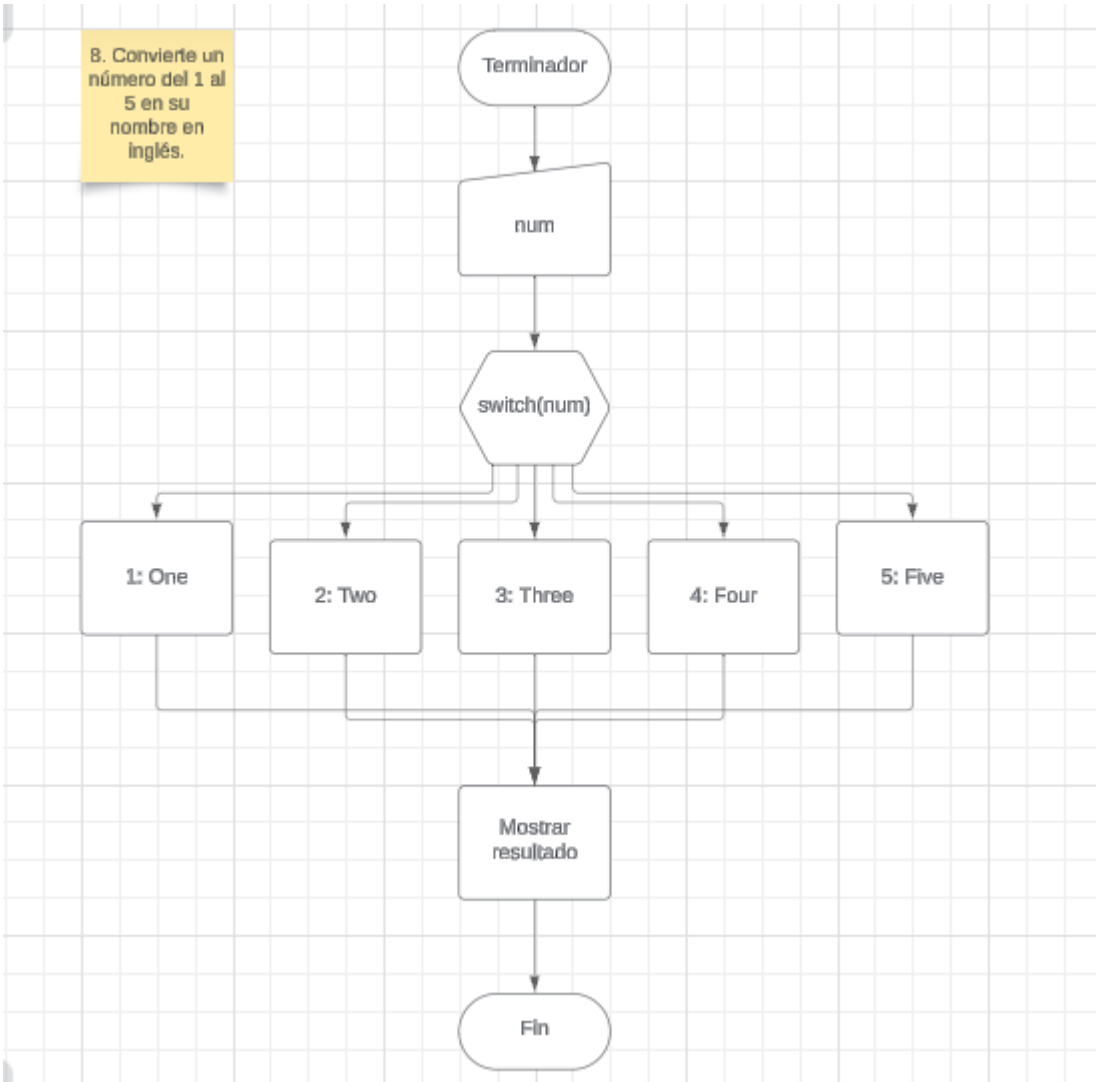
Cada estación abarca 3 meses, por lo que, si el usuario ingresa un mes, se mostrará si la estación está comenzando, está en su plenitud o está terminado. Por ejemplo: si se ingresa marzo, se mostrará: Está empezando la primavera; si se ingresa junio, se mostrará: Esta terminado la primavera, y así sucesivamente con los demás meses.



8. Convierte un número del 1 al 5 en su nombre en inglés.

- Problema: Dado un número, dar su nombre en inglés.
- Datos de entrada: Un numero dado por el usuario.
- Solución:
    1. Leer número: ingresado por el usuario.
    2. Aplicar: Si el usuario ingresa 1, se muestra: “One”, ya que es su nombre en inglés, así sucesivamente hasta el 5.
- Salida: El nombre en inglés del número ingresado.

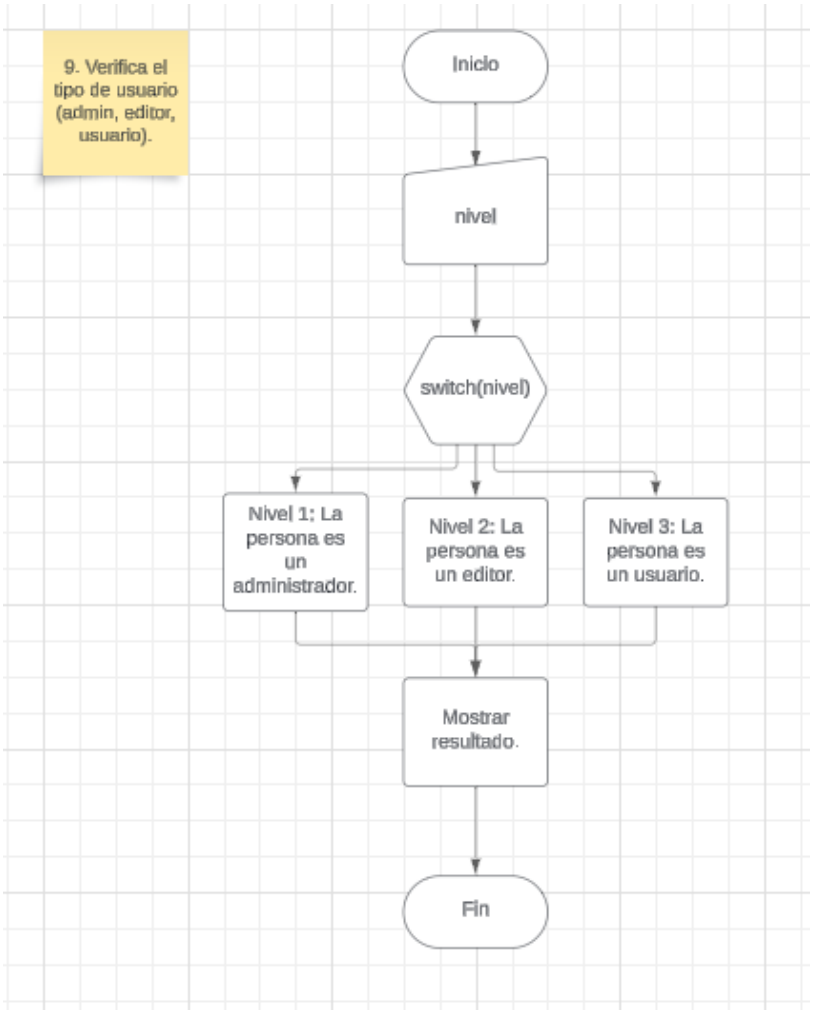
Este enunciado, nos pide que, al ingresar un número, se debe devolver el nombre del mismo, pero en inglés. Con ayuda de la sentencia switch case, fue asignas a cada número, su nombre correspondiente en inglés. Por ejemplo: si se ingresa, se mostrará: “Five” y así sucesivamente, al llegar a la opción 5, terminará el proceso porque ya no hay más opciones.



9. Verifica el tipo de usuario (admin, editor, usuario).

- Problema: Determinar el tipo de usuario según su rol: admin, editor o usuario.
- Datos de entrada: el nivel, ingresado por el usuario.
- Solución:
    1. Leer el nivel: Ingresado una persona:
    2. Aplicar: Si la cadena es “nivel 1”, imprimir “ La persona es un administrador”.
       Si la cadena es “nivel 2”, imprimir “L apersona es un editor”.
       Si la cadena es “nivel 3”, imprimir “La persona es un usuario”.
       Si la cadena no coincide con ninguno de los roles anteriores, imprimir “Tipo de persona desconocida”.
- Salida: Mostrar el mensaje correspondiente al nivel ingresado.

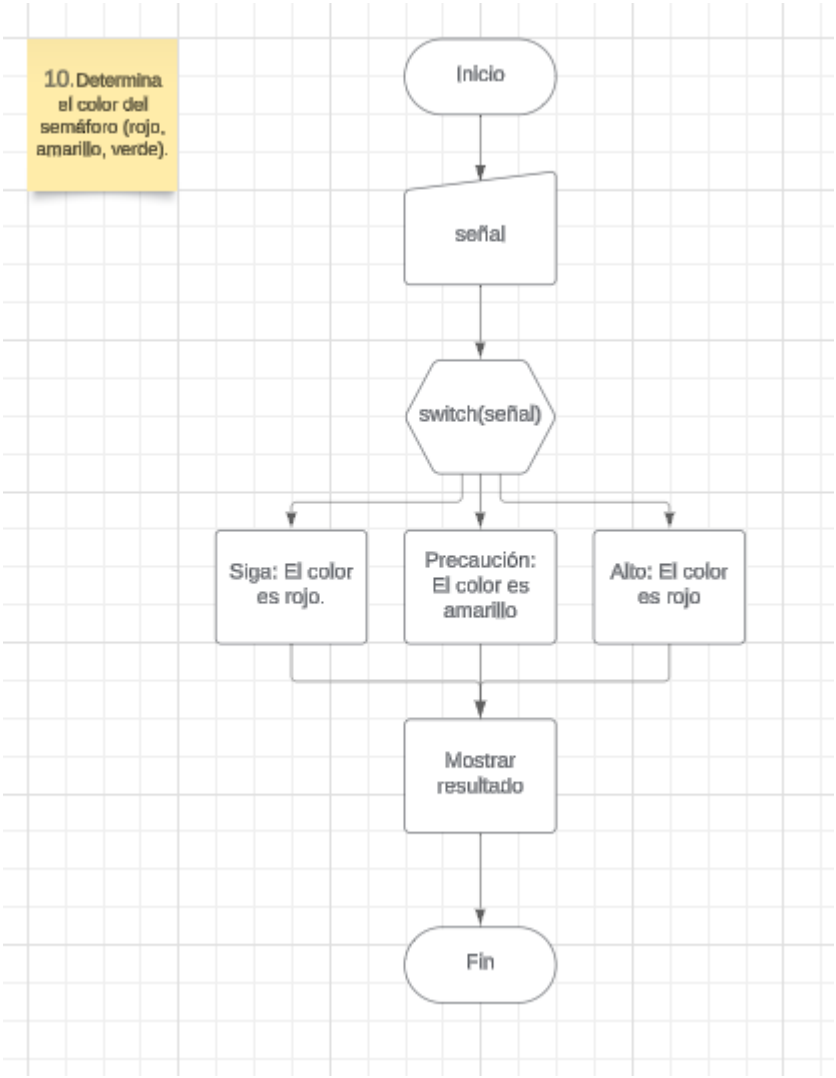
En este planteamiento, se nos pide que verifiquemos el rol de una persona, les asigne los roles dados a niveles. Se ejecuta de la siguiente manera: “nivel 1”, se mostrará “La persona es un administrador”; si se ingresa “nivel 2”, se mostrará “La persona es un editor” y si se ingresa “nivel 3”, se mostrará “La persona es un usuario”. Si no hay ninguna coincidencia, se mostrar que la persona es desconocida.



10. Determina el color del semáforo (rojo, amarillo, verde).

- Problema: Determinar el color del semáforo el significado de la señal.
- Datos de entrada: Señal que el usuario ingresa.
- Solución:
    1. Leer señal: Ingresada por el usuario.
    2. Aplicar: Si se ingresa “Siga”, se muestra “Verde”;
       si se ingresa “Precaución”, se muestra “Amarillo”;
       si se ingresa “Alto”, se muestra “Rojo”;
       si se ingresa otra cosa se muestra “Señal no valida”.
- Salida: Mostrar el color de la señal ingresada.

La forma de determinar el color de un semáforo, se tiene que relacionar con lo que quiere decir cada color, teniendo eso en cuenta, se aplica: si se ingresa “Siga”, se muestra “Verde”, si se ingresa “Precaución”, se muestra “Amarillo” y si se ingresa otra cosa se muestra “Señal no valida”. Al no encontrar coincidencia, muestra de que lo ingresado no es válido y termina el proceso.



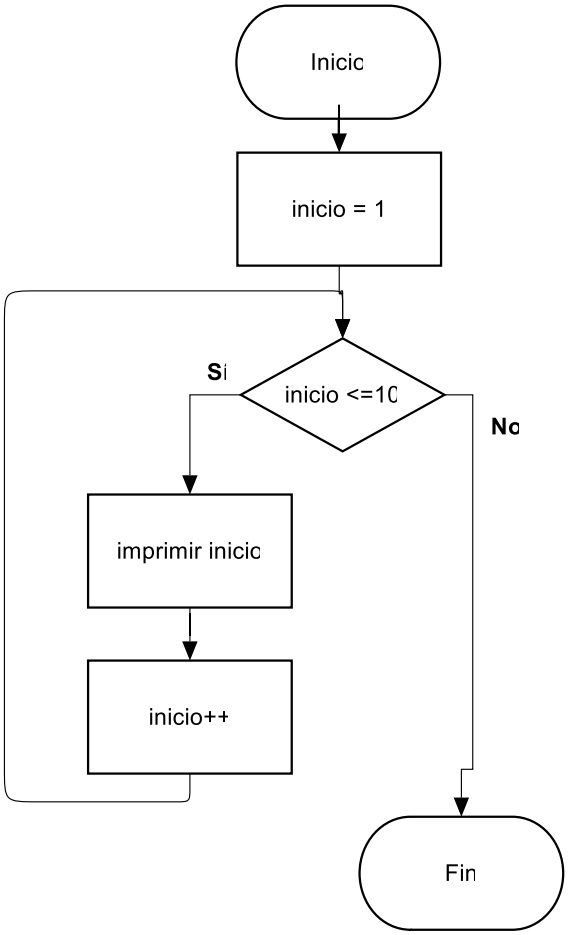
Parte 5: Estructuras de control de iteraciones.

Sentencia for:

1. Imprime los números del 1 al 10.

- Problema: Imprimir los numero del 1 al 10.
- Datos de entrada: En este caso, no hay datos de entrada, pues ya hay un rango establecido
- Solución:
    1. Aplicar: Inicio = 1; se inicia en 1,
    2. Inicio <= 10; se pone la condición para que no pase de 10,
    3. Inicio++; se incrementa el inicio para llegar a 10,
- Salida: Mostrar los 10 primeros números.

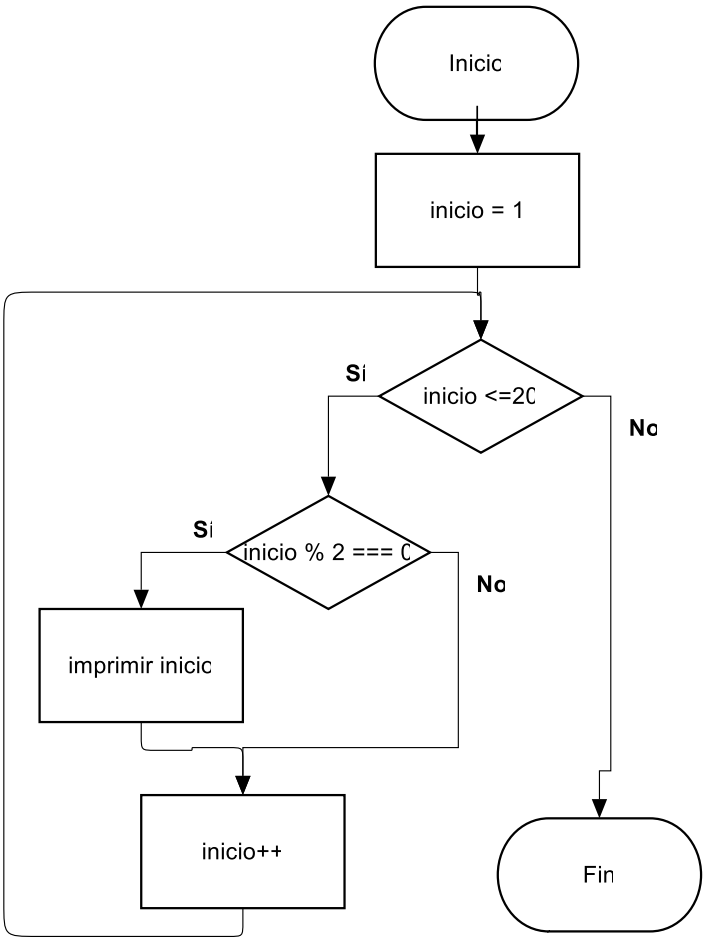
La sentencia for, funciona en bucle, el proceso empieza por el 1, pues se debe declarar una inicialización, después, la condición es que no se debe pasar del número ingresado, e imprime el número; para llegar a 10, hay un incremento que hace que imprima el número siguiente, al llegar al número indicado (en este caso 10), termina el bucle y es el fin del proceso.



2. Imprime los números pares entre 1 y 20.

- Problema: Imprimir los numero pares entre 1 y 20.
- Datos de entrada: No hay, pues ya hay un rango establecido.
- Solución:
    1. Aplicar: Aplicar: Inicio = 1; se inicia en 1,
    2. Inicio <= 20; condición para no pasar de 20,
    3. inicio++; para que el inicia vaya incrementando,
    4. inicio % 2 == 0, para que seleccione solo los números pares.
- Salida: Mostrar los números pares que hay entre 1 y 20.

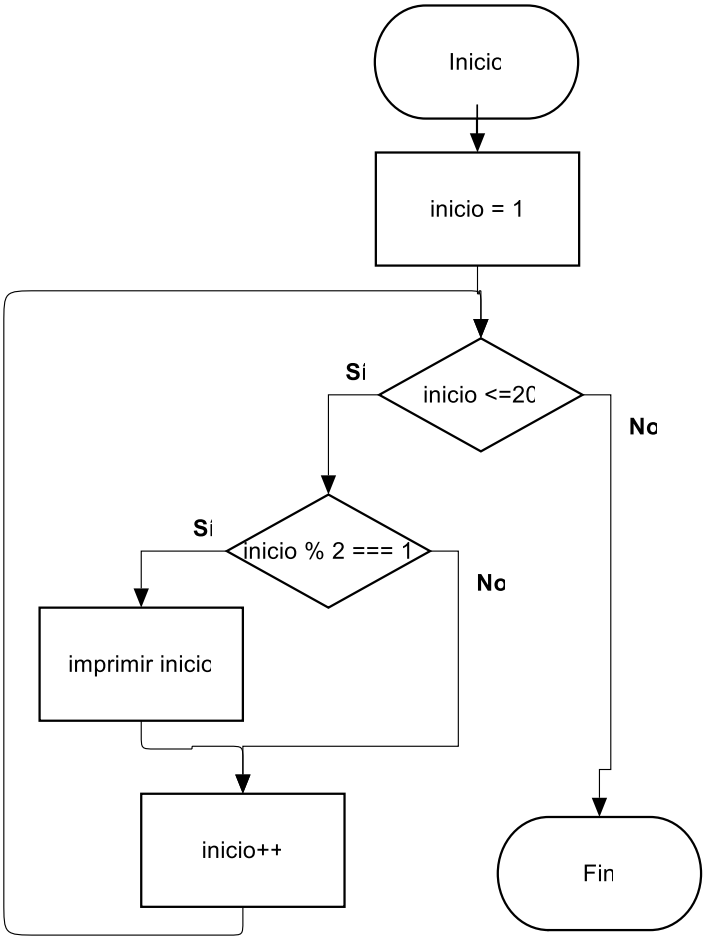
El enunciado nos da el rango del 1 al 20, debemos hacer que el programa muestre los números pares que hay entre rango. Para ello aplicamos: el Inicio = en 1, la condición es que no se debe pasar de 20 y el inicio se incremente, también se agrega una condicional if, que hace que solo los numero pares se muestren (inicio % 2 === 0), y termina el proceso.



3. Imprime los números impares entre 1 y 20.

- Problema: Imprimir los numero pares entre 1 y 20.
- Datos de entrada: No hay, pues ya hay un rango establecido.
- Solución:
    1. Aplicar: Aplicar: Inicio = 1; para que inicie en 1.
    2. Inicio <= 20; condición para que no se pase de 20.
    3. inicio++; para que el inicio vaya incrementando.
    4. inicio % 2 == 1. Para seleccionara solo los numero impares.
- Salida: Mostrar los números impares que hay entre 1 y 20.

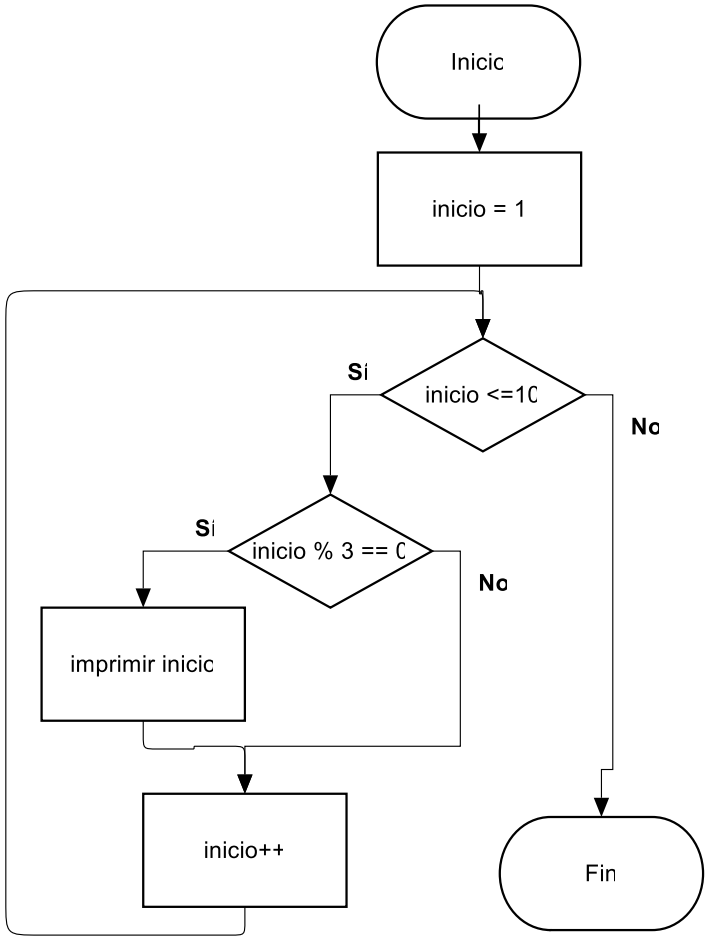
El enunciado nos da el rango del 1 al 20, debemos hacer que el programa muestre los números pares que hay entre rango. Para ello aplicamos: el Inicio = en 1, la condición es que no se debe pasar de 20 y el inicio se incremente, también se agrega una condicional if, que hace que solo los numero impares se muestren (inicio % 2 == 1), y termina el proceso.



4. Imprime los primeros 10 números múltiplos de 3.

- Problema: Imprimir los primero 10 números múltiplos de 3.
- Datos de entrada: No hay, pues ya hay un rango definido.
- Solución:
  1. Aplicar: inicio=1; se inicia en 1,
  2. Inicio<=10; se le pone la condición de no pasarse de 10,
  3. Inicio++; se incrementa el número inicial,
  4. if (inicio % 3 == 0); selecciona los números que sean divisibles entre 3 de forma entera y los muestra.
- Salida: Mostrar los múltiplos de 3.

En este caso, ya se nos dio un rango, entonces se aplica: inicia desde 1, se declara una condición de no pasarse de 10, que el inicio se incremente de 1 en 1 y al finan una condicional if para que solo los numero que sean múltiplos de 3 se muestren y termina el proceso.

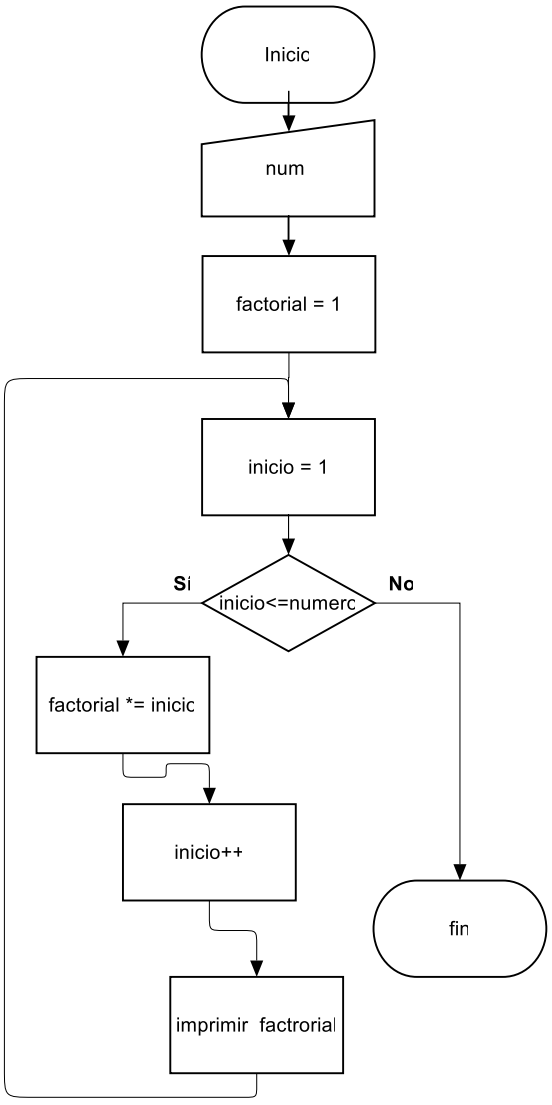


5. Calcula la factorial de un número.

- Problema: Dado un número, calcular su factorial.
- Datos de entrada: Un número dado por el usuario.
- Solución:
  1. Leer número: Dado por el usuario.
  2. Inicializar: inicio=1;
  3. Limitamos: inicio <= número dado;
  4. Incrementamos: inicio++, para que el número dado llegue al límite establecido.
- Salida: Mostrar la factorial del número ingresado.

En este enunciado, se nos pide calcular la factorial de un número ingresado por el usuario.

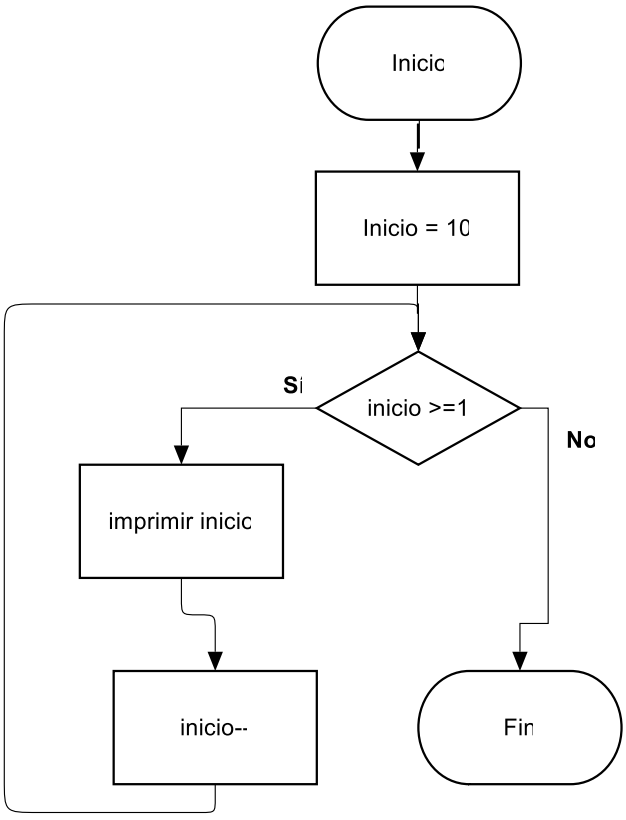
Aplicamos: Leemos el número ingresado por el usuario. Inicializamos en 1, y ponemos un límite con una condición: inicio <= número dado e incrementamos el número de inicio y multiplicamos cada número que salga por el siguiente: 1*2*3*4*5… Para finalizar, se muestra el resultado de las multiplicaciones que fueron necesarias y finaliza el proceso.



6. Imprime los números del 10 al 1 en orden inverso.

- Problema: Imprimir los numero pares entre 1 y 20.
- Datos de entrada: No hay, pues ya hay un rango establecido.
- Solución:
    5. Aplicar: Aplicar: Inicio = 10; se inicia en 10,
    6. Inicio <= 1; condición para no pasar de 1,
    7. Inicio--; para que el inici0 vaya en decremento.
- Salida: Mostrar de manera inversa los números del 1 al 10.

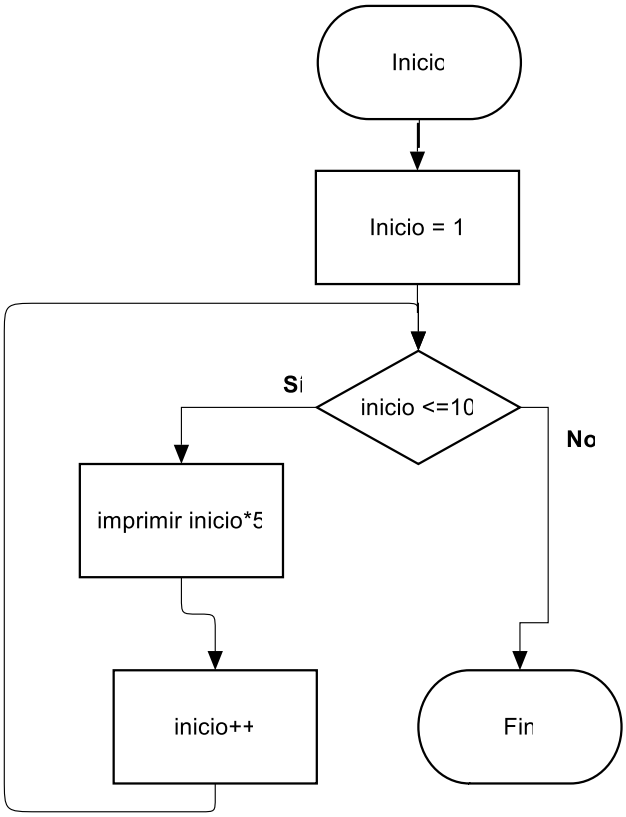
En este planteamiento, se nos da un rango, pero en vez de imprimir en orden ascendente los números, se nos pide que sea al inverso. Se aplica el inicio en 10, luego se agrega una condición para que no se pase de 1, luego se declara un decremento, pues debe ir en descendente e imprimir los numero, cuando llega a 1, termina el proceso.



7. Imprime la tabla de multiplicar del 5.

- Problema: Imprimir la tabla del 5.
- Datos de entrada: Ya se nos está dando un dato que es el número 5.
- Solución:
  1. Se inicializa en 1, pues el primer número a multiplicar.
  2. Se agrega una condición inicio<=10, pues el ultimo número a multiplicar.
  3. También se agrega un incremento, inicio++, para que el número inicial se vaya incrementado de 1 en 1.
  4. Finalmente, se multiplica el numero dado en un principio por el inicio (inicio*5).
- Salida: Mostrar los resultados de la multiplicación que son la taba del 5.

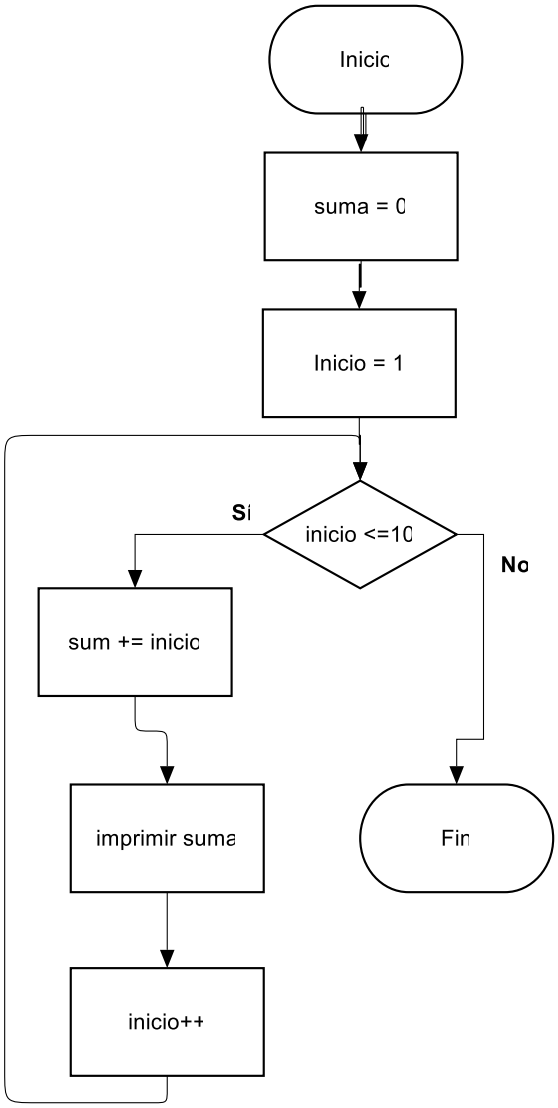
En este planteamiento, se nos pide imprimir la tabla del 5, para ello, declaramos la inicialización en 1, después agregamos un condicional para que se detenga cuando llegue a 10, para que el número inicial llegue a 10, agregamos un incremento y al momento de imprimir, mostramos los resultados de la sentencia for y los multiplicamos por 5, lo que nos da la tabla del 5.



8. Suma los primeros 10 números.

- Problema: Sumar los primeros 10 números.
- Datos de entrada: Se nos da un rango del 1 a 10, con eso ya podemos trabajar.
- Solución:
  1. Iniciamos la sentencia: inicio = 1;
  2. Agregamos una condicional: inicio<=10, para no pasar del límite indicado;
  3. Incrementamos el inicio: inicio++, para que vaya incrementado de 1 en 1 hasta llegar al 10;
  4. Aplicar: suma += inicio, para la suma de todos los números resultantes.
- Salida: Mostrar la suma de los numero del 1 al 10.

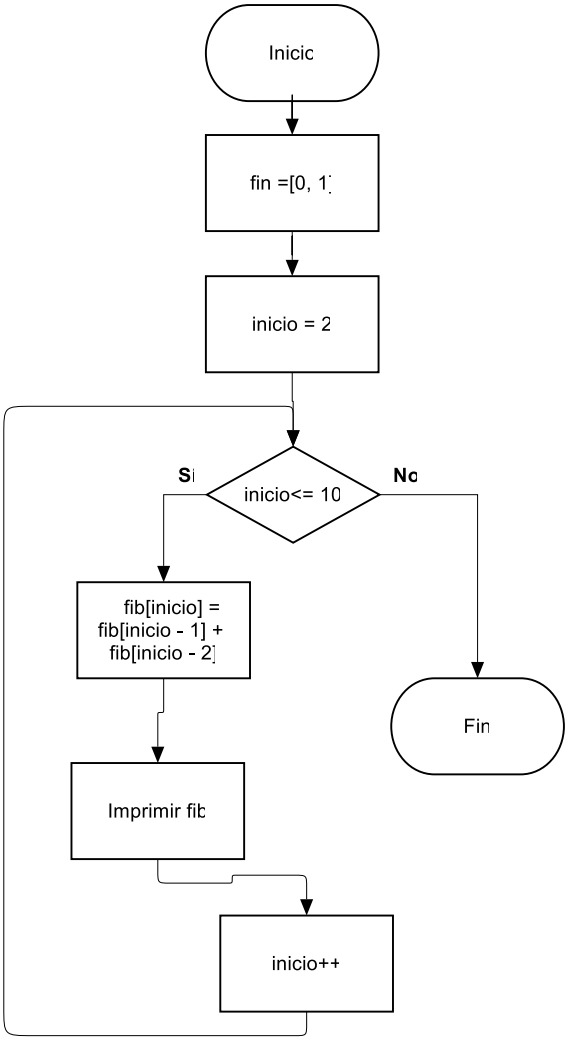
Se nos da un rango de los primeros 10 números, los cuales témenos que sumar. Para ellos, iniciamos sumando desde el 1: inicio = 1; después establecemos un límite con una condicional, en este caso 10: inicio<=10; para que llegue a diez, el inicio debe incrementarse: inicio++ y al final todos los números se suman: suma += inicio.



9. Imprime los primeros 10 términos de la serie Fibonacci.

- Problema: Imprimir los 10 primero términos de la serie Fibonacci.
- Datos de entrada: No hay pues ya hay un rango definido.
- Solución:
  1. Se aplica un array con los dos primeros términos de la serie Fibonacci: fib [0, 1];
  2. Inicializamos la sentencia for en 2, pues ya los primeros están declarados;
  3. se agrega una condición para no pasar del límite: inicio<=10
  4. Para calcular el siguiente término de la secuencia, se suman los dos términos anteriores (fib[inicio-1] + fib[inicio-2]).
- Salida: Mostrar los primero 10 de la serie Fibonacci.

Este caso fue un poco desafiante, para resolverlo, primero se declara un array con los dos primeros términos de la serie Fibonacci: 0 y 1; después se inicia la sentencia for en 2, pues ya están los primeros, se determina que solo serán 10 términos: inicio<=10 y que el inicio se incremente: inicio++. Para que sen muestren los demás términos, se aplica: (fib[inicio-1] + fib[inicio-2]), las restas que se ven, no lo son, quiere decir que regresan uno y dos índices correspondientemente, cuándo llega al termino 10, muestra el resultado y termina el proceso.

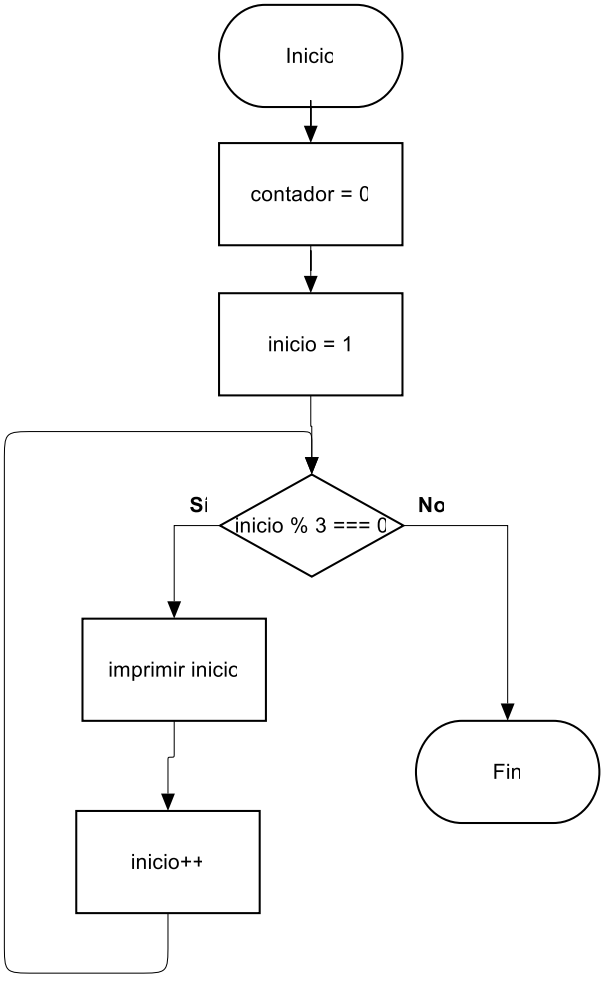


10. Cuenta cuántos números entre 1 y 100 son divisibles por 3.

- Problema: Mostrar la cantidad de números divisibles entre 3 que hay del 1 al 100
- Datos de entrada: No hay, pues ya hay un rango ya dado.
- Solución:
  1. Declaramos una variable para llevar el conteo.
  2. Inicializamos la sentencia for en 1: inicio=1;
  3. Determinamos el rango con una condición: inicio<=100;
  4. Incrementamos el número inicial: inicio++;
  5. Agregamos una sentencia if con %3 para saber que el número es divisible entre 3;
  6. Para finalizar, cada número que cumpla la sentencia if, va a incrementar al contador.
- Salida: Mostrar la cantidad de numero divisibles entre 3 entre 1 y 100.

Para este enunciado, se pide contar la cantidad de número resultantes.

Primero declaramos una variable que tendrá a la función de contador, inicializamos la sentencia for en 1: inicio=1, determinamos el rango pedido: inicio<=100 y hacemos que incremente con: inicio++, para encontrar los números divisibles entre 3, se agrega una sentencia if con % 3, cada número que cumpla con esa condición, si imprime y aumenta uno en uno al contador, cuando llegue al 100, se mostrara la cantidad de números y terminara el proceso.

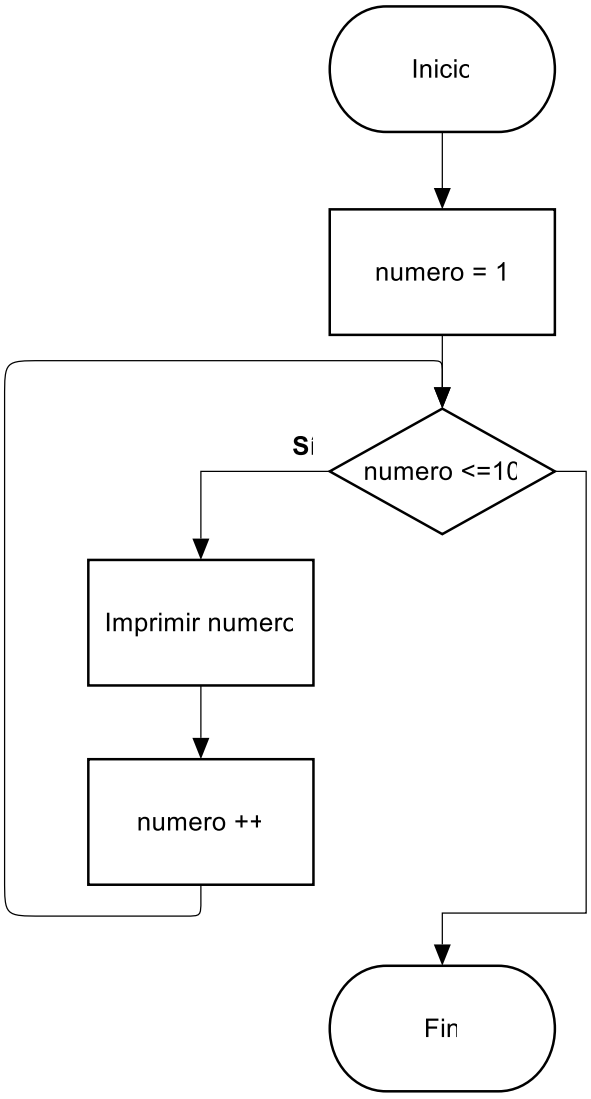


Sentencia while:

1. Imprime los números del 1 al 10.

- Problema: Imprimir los números consecutivos del 1 al 10
- Datos de entrada: No hay, pues los números ya están definidos.
- Solución:
  1. Se declara una variable que será el número con el que comenzaremos.
  2. Se inicia la sentencia while con una condición: numero<=10;
  3. Se muestra el número actual y se incrementa de 1 en 1: numero++.
- Salida: Imprimir los números del 1 al 10.

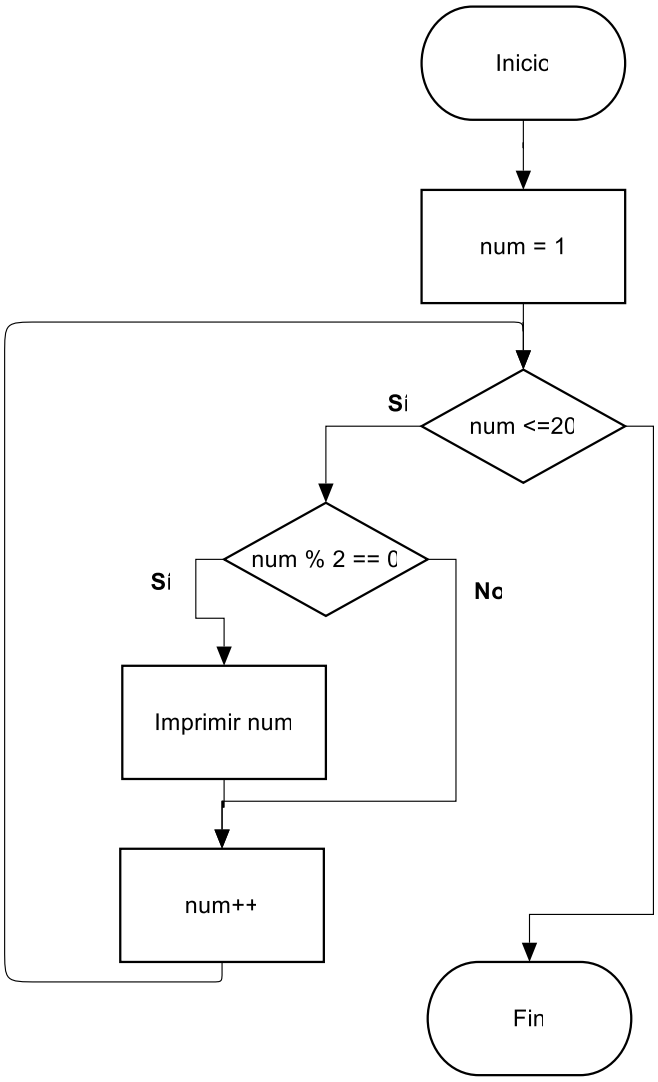
La sentencia while, trabaja con una condición, mientras esta se cumpla, se hará un bucle. Para este caso, se inicia con una variable, con su valor se crea la condición para la sentencia while: numero<=10, después se imprime en número actual (1 en este caso) y para que se imprima hasta el 10, se incrementa de 1 en 1 con: numero++ y termina el proceso.



2. Imprime los números pares entre 1 y 20.

- Problema: Imprimir los numero pares que hay entre 1 y 20.
- Datos de entrada: No hay, pues ya se dio un rango con el cual trabajar.
- Solución:
  1. Se inicia la variable con valor 1: numero = 1;
  2. Se usa un bucle while que continúa mientras número sea menor o igual a 20.
  3. Dentro del bucle while:
     Verificar si número es par usando la condición número % 2 == 0.
     Si la condición es verdadera, imprimir número.
     Incrementar número en 1 al final de cada iteración.
- Salida: Los números pares entre 1 y 20 impresos en la consola.

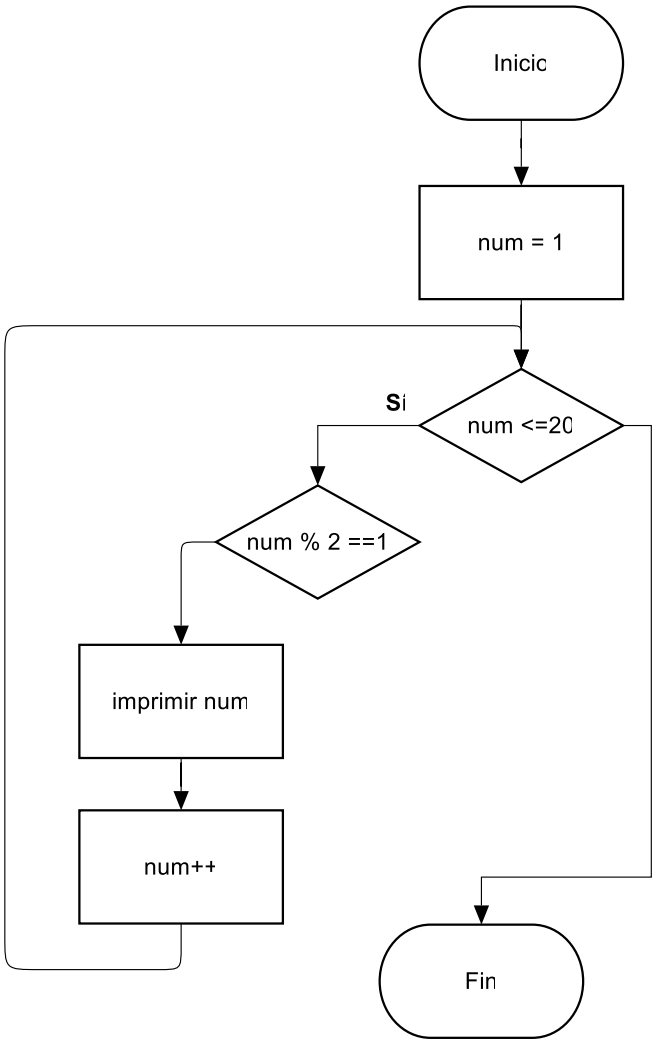
Se inicia con una variable en 1: numero =1; con esta se puede iniciar la sentencia while la cual tiene como condicional que el número inicial sea menor o igual 20, dentro del bucle, se agrega una sentencia f, la cual solo deja mostrar los numero pares: número % 2 == 0, finalmente se muestran en la consola y termina el proceso.



3. Imprime los números impares entre 1 y 20.

- Problema: Imprimir los numero impares que hay entre 1 y 20.
- Datos de entrada: No hay, pues ya se dio un rango con el cual trabajar.
- Solución:
    4. Se inicia la variable con valor 1: numero = 1;
    5. Se usa un bucle while que continúa mientras número sea menor o igual a 20.
    6. Dentro del bucle while:
       Verificar si número es par usando la condición número % 2 == 1.
       Si la condición es verdadera, imprimir número.
       Incrementar número en 1 al final de cada iteración.
- Salida: Los números impares entre 1 y 20 impresos en la consola.

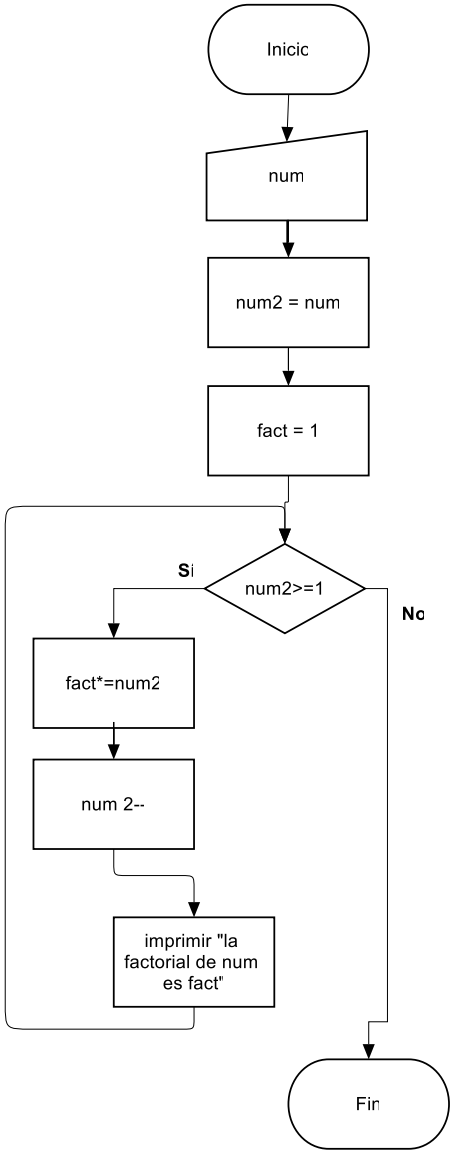
Se inicia con una variable en 1: numero =1; con esta se puede iniciar la sentencia while la cual tiene como condicional que el número inicial sea menor o igual 20, dentro del bucle, se agrega una sentencia f, la cual solo deja mostrar los numero impares: número % 2 == 1, finalmente se muestran en la consola y termina el proceso.



4. Calcula la factorial de un número.

- Problema: Dado un número calculara su factorial.
- Datos de entrada: Un número dado por el usuario.
- Solución:
  1. Leer el número: Dado por el usuario.
  2. Se agrega una variable para almacenar el resultado de la factorial: fact = 1;
  3. Iniciar la sentencia while: número>=1
  4. Se multiplica fact*=numero
  5. Y se decrementa el número inicial: numero—
- Salida: La factorial del número ingresado.

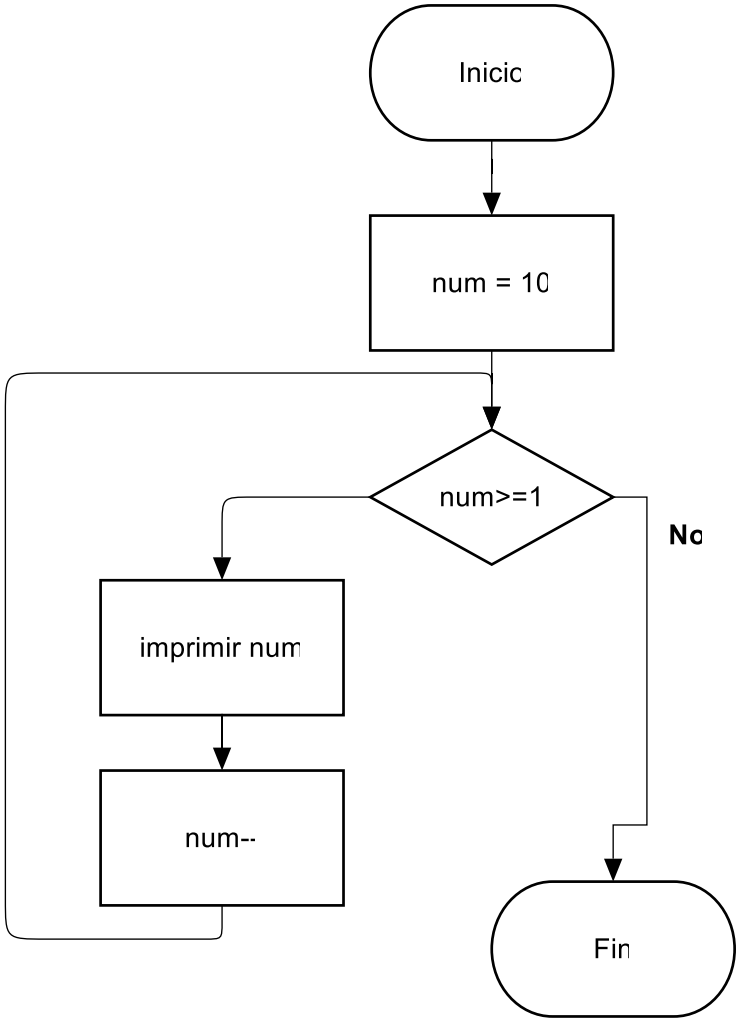
Iniciamos con una variable que va a ser el número ingresado, iniciamos la sentencia for con la condición del que el número sea igual o mayor que 1. Multiplicamos la fact*el número y decrementamos número—y se muestra el resultado.



5. Imprime los números del 10 al 1 en orden inverso.

- Problema: Imprimir los numero del 1 la 10 inversamente.
- Datos de entrada: No hay, pues ya hay un rango definido.
- Solución:
  1. Definir la variable de inicio: num=10
  2. Iniciar la sentencia while: num>=1
  3. Se imprime.
- Salida: Mostrar los numero inversamente.

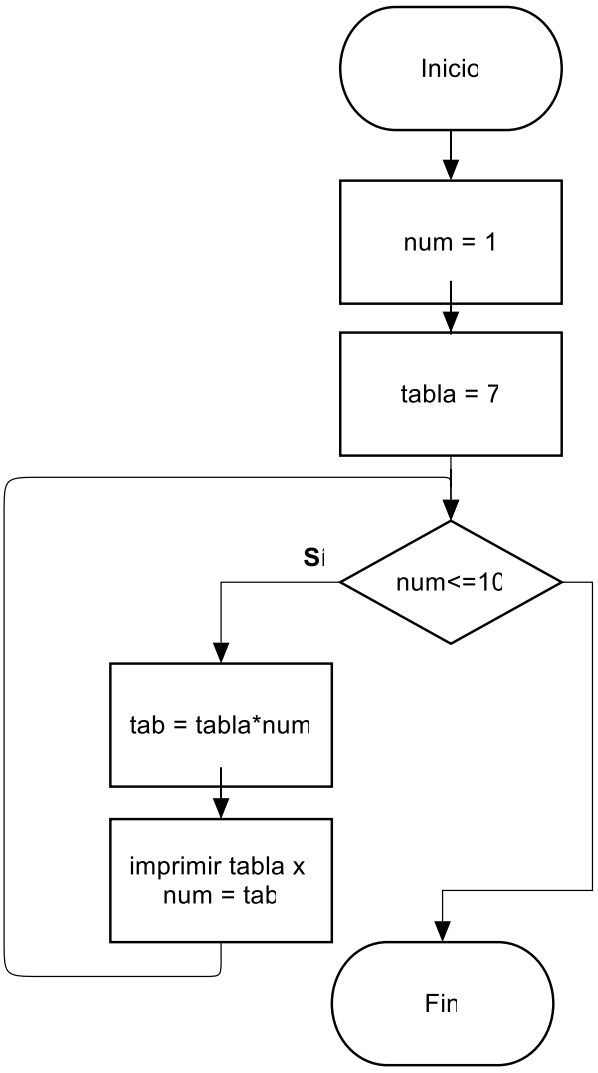
La sentencia while, trabaja con una condición, mientras esta se cumpla, se hará un bucle. Para este caso, se inicia con una variable, con su valor se crea la condición para la sentencia while: numero>=1, después se imprime en número actual (10 en este caso) y para que se imprima hasta el 1, se decrementa de 1 en 1 con: número—y termina el proceso.



6. Imprime la tabla de multiplicar del 7.

- Problema: Imprimir la tabla del 7.
- Datos de entrada: No hay, pues ya se dio un dato.
- Solución:
  1. Declarar una variable de inicio: numero = 1;una para almacenar resultados y una la tabla pedida: tabla = 7;
  2. Iniciar la sentencia while: con la condición de numero<=10;
  3. Lo siguiente es: multiplicar la tabla * número.
  4. Finalmente se incrementa: numero++, para llegar a multiplicar hasta 10.
- Salida: Mostrar la tabla del 7.

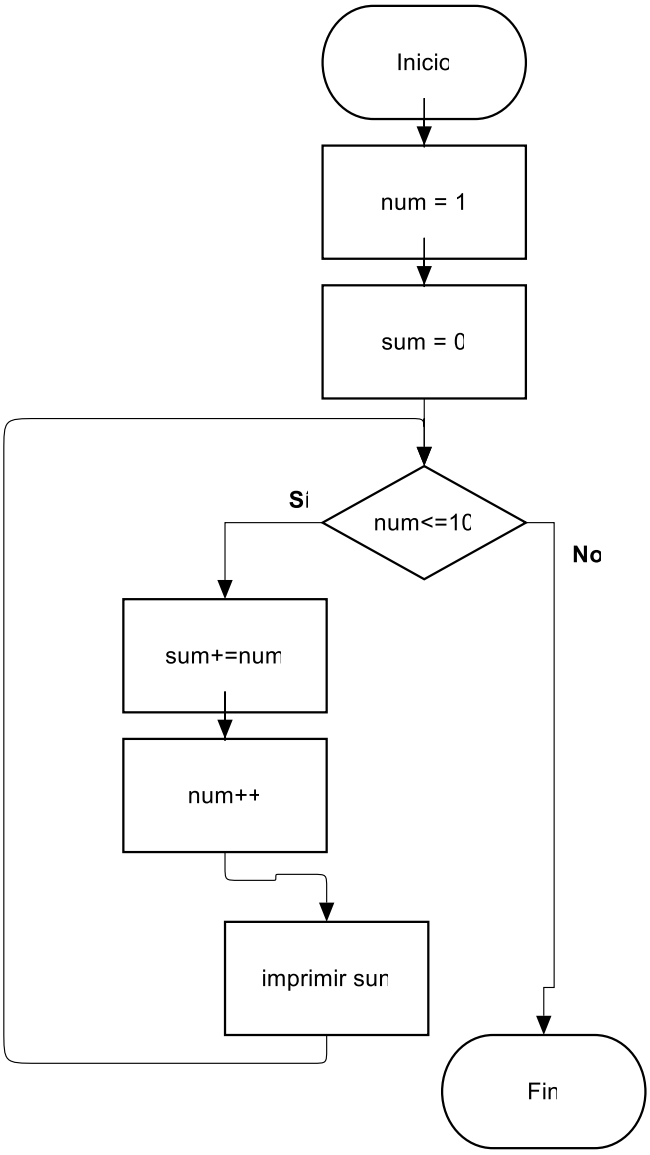
Para este planteamiento, se inicia declarando una variable de inicio: numero = 1; otra para los resultados y otra que es la tabla pedida: tabla = 7, con esta se inicia la sentencia while con la condición número<=10 para no pasar a multiplicar más de 10, para mostrar el resultado, primero se multiplica tabla * numero; para terminar de multiplicar por los demás números, se incrementa: numero++, después, cuándo la condición no se cumple, termina el proceso.



7. Suma los primeros 10 números.

- Problema: Sumar los primeros 10 números.
- Datos de entrada: No se ingresan datos, pues ya hay un rango definido.
- Solución:
  1. Declarar variable de inicio: numero =1; y otra para almacenar el resultado: Suma = 0.
  2. Iniciar sentencia while: con la condición número<=10.
  3. Se suman los números: suma+=numero
  4. Incrementamos el número inicial: numero ++.
- Salida: La suma de los primeros 10 números.

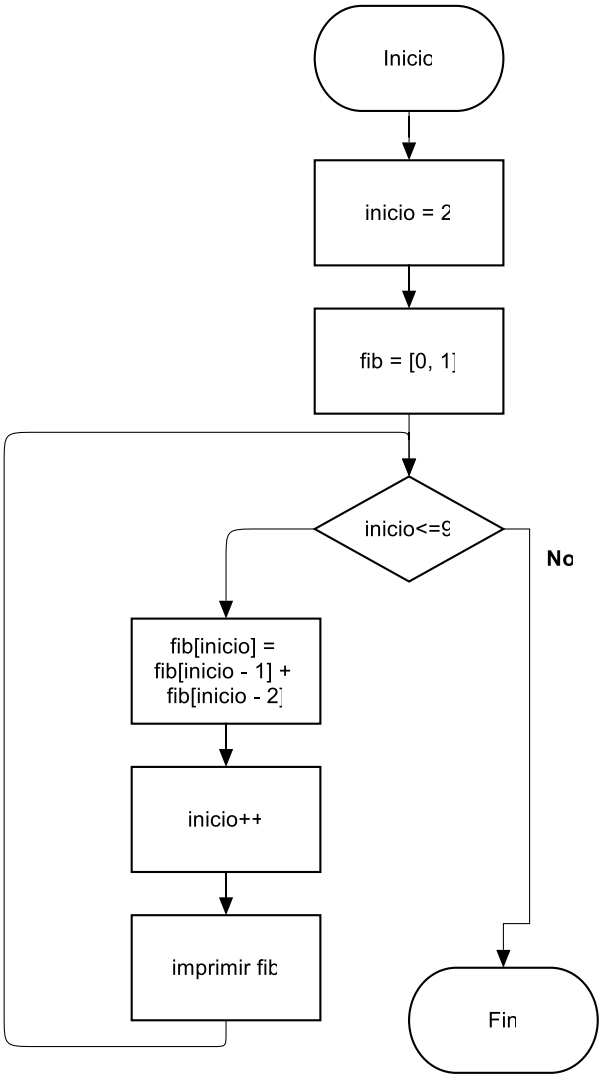
Para sumar los primeros números se inicia con una variable con valor 1 y otra para almacena el resultado: suma. Se comienza la sentencia while con la condición de num<=10, después se suman los números resultantes: suma+=numero; se muestra el resultado y termina el proceso.



8. Imprime los primeros 10 términos de la serie Fibonacci.

- Problema: Imprimir los 10 términos de la serie Fibonacci.
- Datos de entrada: No hay entrada manual, pues ya se dio un dato en el enunciado.
- Solución:
  1. Se inicia declarando una variable de inicio: inicio = 2 y otra de los primeros términos: fib = [0, 1].
  2. Comenzamos la sentencia while: Inicio = 9, pues se cuenta desde el 0.
  3. Se aplica la formula: fib[inicio] = fib[inicio - 1] + fib[inicio - 2]
  4. Se incrementa el número inicial: inicio++.
- Salida: Mostrar los primeros 10 términos de la serie Fibonacci.

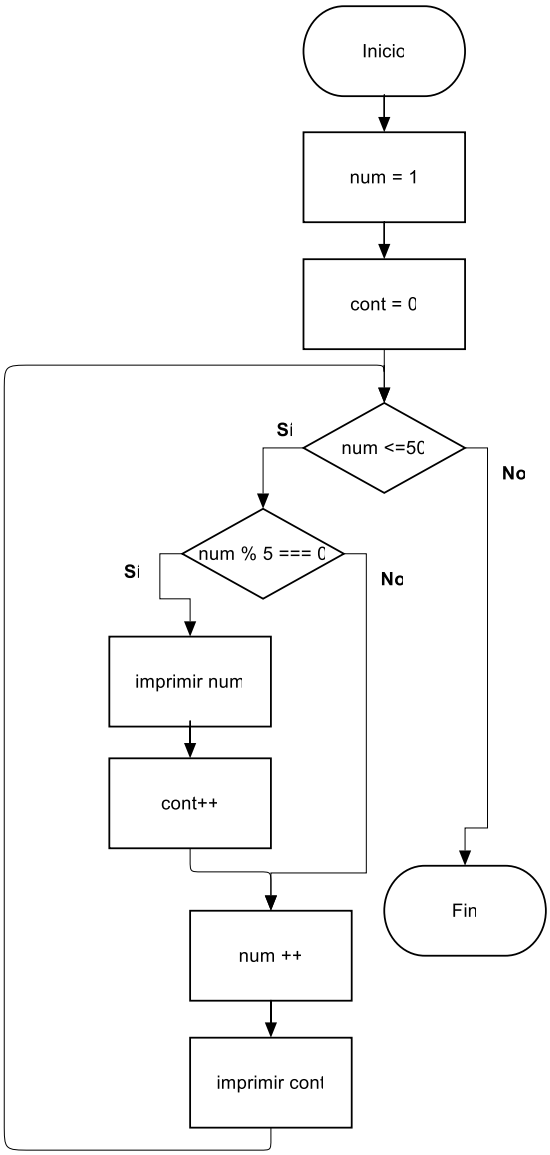
Para mostrar los 10 primeros términos de la serie de l Fibonacci, comencé declarando una variable de inicio: inicio = 2 y otra con los 2 primeros términos fib = [0, 1]; con esto, inicié la sentencia while y con la condición de inicio<=9 pues se empieza desde el 0. Ya teniendo el bucle, se aplica la formula fib[inicio] = fib[inicio - 1] + fib[inicio - 2] para obtener lo siguientes términos con la ayuda de un incremento.



9. Cuenta cuántos números entre 1 y 50 son divisibles por 5.

- Problema: Cuenta cuántos números entre 1 y 50 son divisibles por 5.
- Datos de entrada: El rango dado en el enunciado.
- Solución:
  1. Declarar una variable de inicio: num = 1; y una de contador: cont = 0;
  2. Iniciar la sentencia while: determinando el rango = num<=50.
  3. Aplicar una sentencia if: num % 5 === 0, cada vez que se cumpla, aumente 1 en el contador y así sucesivamente hasta que la condición de while ya no se cumpla:
- Salida: Mostrar los números divisibles entre 5 y la cantidad.

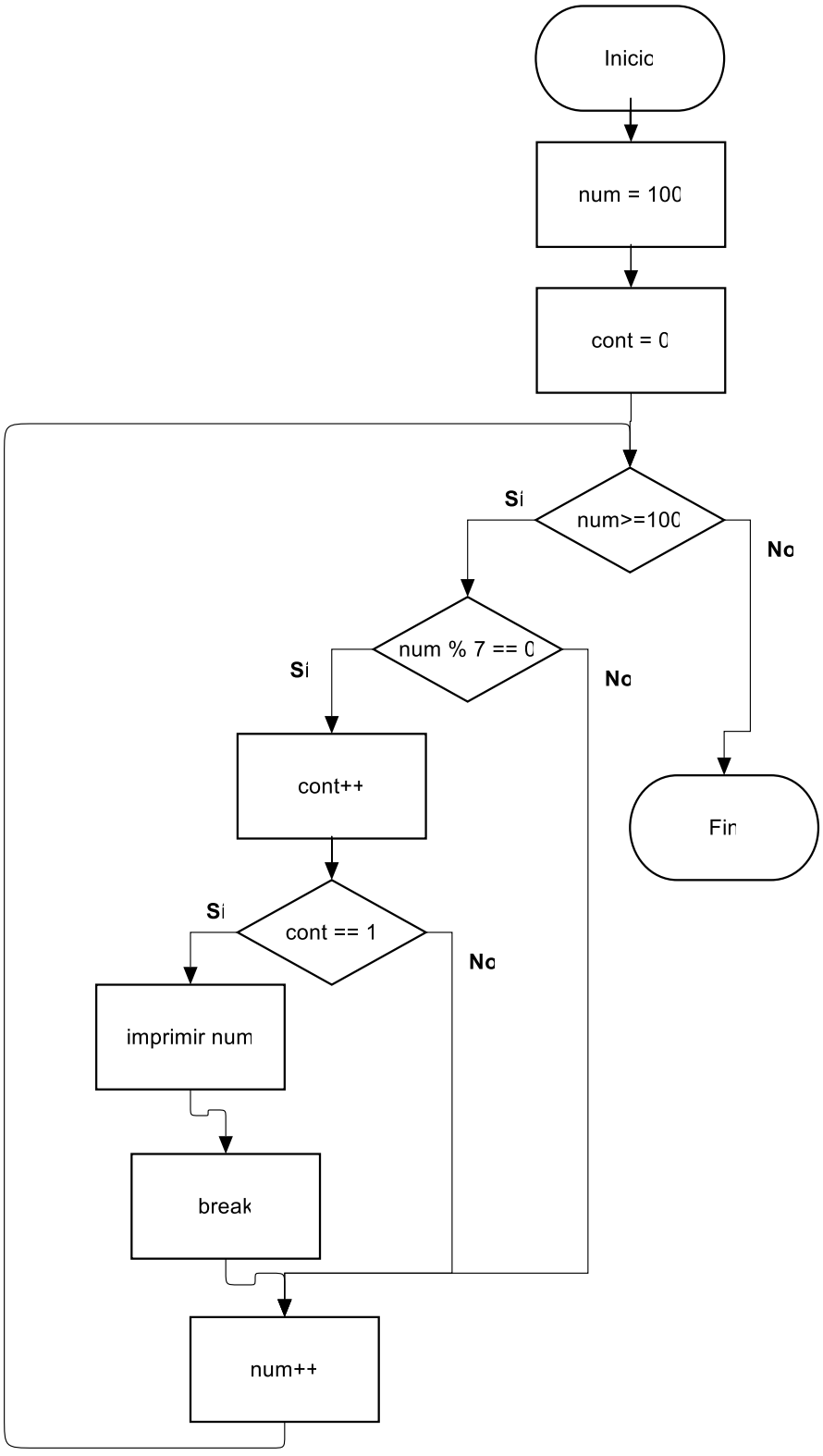
Este problema es sencillo, se inicia declarando do variables, una de inicio en 1 y la otra para funcionar como contador. Después, se inicia la sentencia while con la condición num<=50, cada vez que se cumpla el contador incrementa 1, para que el número inicial llegue a 50, también hay que incrementar el número inicial. Cuando la condición de while no se cumple, termina el proceso.



10. Encuentra el primer número divisible por 7 después de 100.

- Problema: Encuentra el primer número divisible por 7 después de 100.
- Datos de entrada: En este caso no hay entrada Manuel, pero en el enunciado se nos indica que el divisor es 7 y el rango es mayor o igual a 100.
- Solución:
  1. Declarar variable de inicio: numero = 1;
  2. Declarar variable de conteo: contador = 0;
  3. Iniciar sentencia while: con la condición: numero<=50;
  4. Dentro del bucle se agrega una sentencia if: numero % 5 === 0;
  5. Si la condición se cumple, se incrementa el contador: contador++.
  6. Incrementar el número inicial: numero++.
- Salida: Mostrar el primer número después de 100 que es divisible entre 7.

Para dar solución a este planteamiento, me di cuenta que tiene dos condiciones. Comenzamos declarando una variable inicial (debe empezar de 100 en adelante): número = 100 y un contador en 0; iniciamos la sentencia while con una condición: numero >=100, después la primera condición: num % 7 === 0, esta verifica si el numero que entre es divisible netre 7, si se cumple incrementa 1 en el contador. La segunda condición consiste: cont == 1, e imprime el número y usa el break para romper el bucle y termina el proceso.

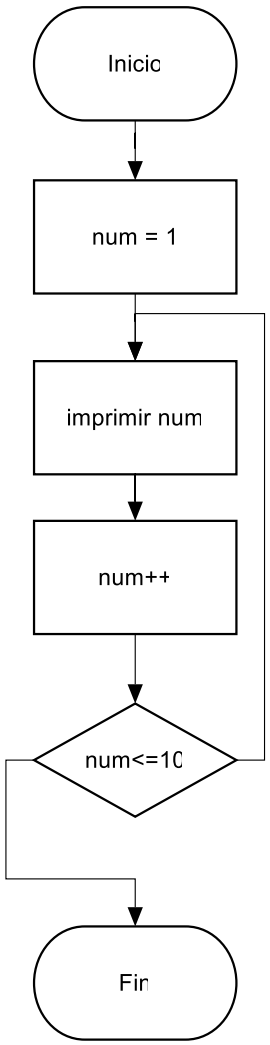

Inicio
num = 100
cont = 0
Si
num>=100
No
Si
num % 7 == 0
No
cont++
Fin
Si
cont == 1
No
imprimir num
break
num++

Sentencia do-while:

1. Imprime los números del 1 al 10.

- Problema: Imprimir los primeros 10 números.
- Datos de entrada: El enunciado nos da un rango de 1 al 10.
- Solución:
    1. Declarar variable: numero = 1;
    2. Abrir do: dentro, imprimir número e incrementarlo 1 en 1;
    3. Agregar sentencia while: numero<=10.
- Salida: Mostrar los 10 primeros números.

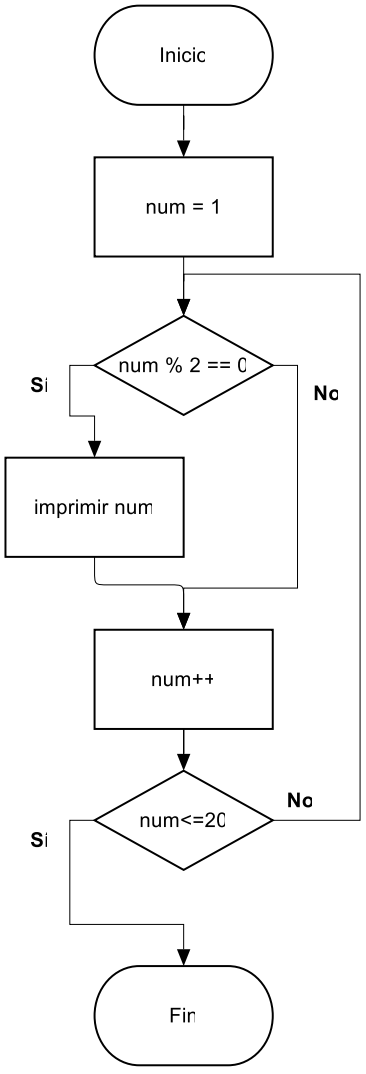
Para darle solución a este planteamiento, se inicia declarando la variable de inicio: inicio = 1. Se abre un bucle do, dentro del cual se imprimirá el número actual y se incrementará en 1 con cada iteración. La sentencia while verifica si el número es menor o igual a 10, permitiendo que el bucle continúe mientras esta condición sea verdadera.



2. Imprime los números pares entre 1 y 20.

- Problema: Imprimir los números pares entre 1 y 20.
- Datos de entrada: No hay datos de entrada de parte del usuario, solo el rango de 1 a 20.
- Solución:
    1. Declarar variable: numero = 1.
    2. Abrir una sentencia do...while:
       Dentro del bucle do, verificar si número es par.
       Si la condición se cumple, imprimir el valor de número.
       Incrementar número en 1.
    3. Agregar sentencia while para verificar que numero <= 20.
- Salida: Mostrar los números pares entre 1 y 20.

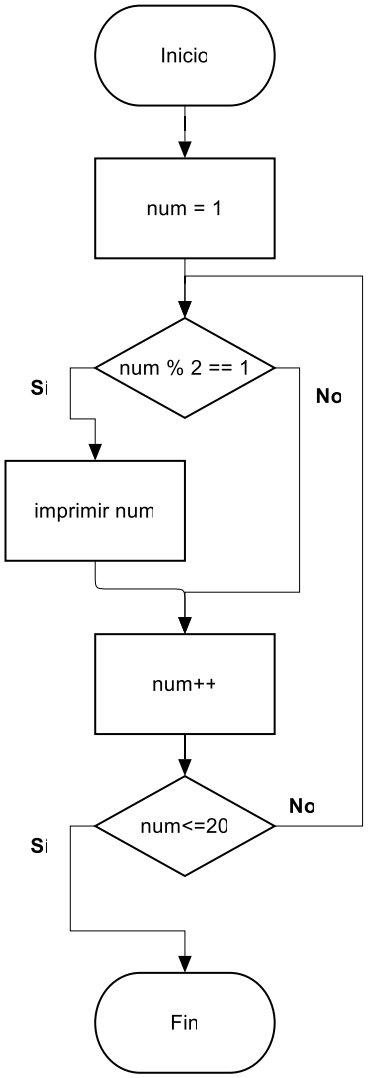
Para darle solución a este planteamiento, se inicia declarando la variable de inicio: numero = 1. Se abre un bucle do, dentro del cual se verifica si el número es par. Si es par, se imprime el número. Finalmente, se incrementa el número inicial y se verifica si es menor o igual a 20. Al llegar a 20, termina el proceso.



3. Imprime los números impares entre 1 y 20.

- Problema: Imprimir los números impares entre 1 y 20.
- Datos de entrada: No hay datos de entrada de parte del usuario, solo el rango de 1 a 20.
- Solución:
    4. Declarar variable: numero = 1.
    5. Abrir una sentencia do...while:
       Dentro del bucle do, verificar si número es impar.
       Si la condición se cumple, imprimir el valor de número.
       Incrementar número en 1.
    6. Agregar sentencia while para verificar que numero <= 20.
- Salida: Mostrar los números pares entre 1 y 20.

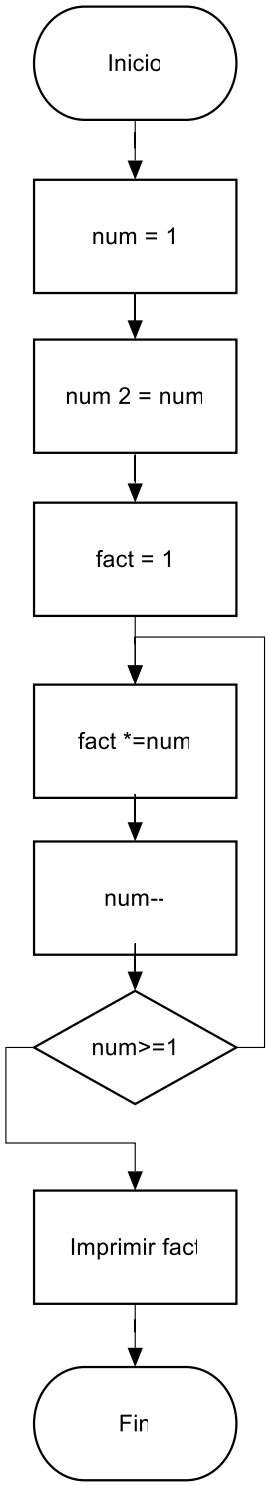
Para darle solución a este planteamiento, se inicia declarando la variable de inicio: numero = 1. Se abre un bucle do, dentro del cual se verifica si el número es impar. Si es impar, se imprime el número. Finalmente, se incrementa el número inicial y se verifica si es menor o igual a 20. Al llegar a 20, termina el proceso.



4. Calcula la factorial de un número.

- Problema: Imprimir los números impares entre 1 y 20.
- Datos de entrada: El rango dado es de 1 a 20.
- Solución:
  1. Declarar variable: numero = 1.
  2. Declarar otras variables para almacenar el resultado: fact = 1;
  3. Abrir una sentencia do...while:
     Dentro de do, la fact*=numero;
     y decrementa el número.
  4. Agregar sentencia while para verificar que numero <= 1.
- Salida: Mostrar los números pares entre 1 y 20.

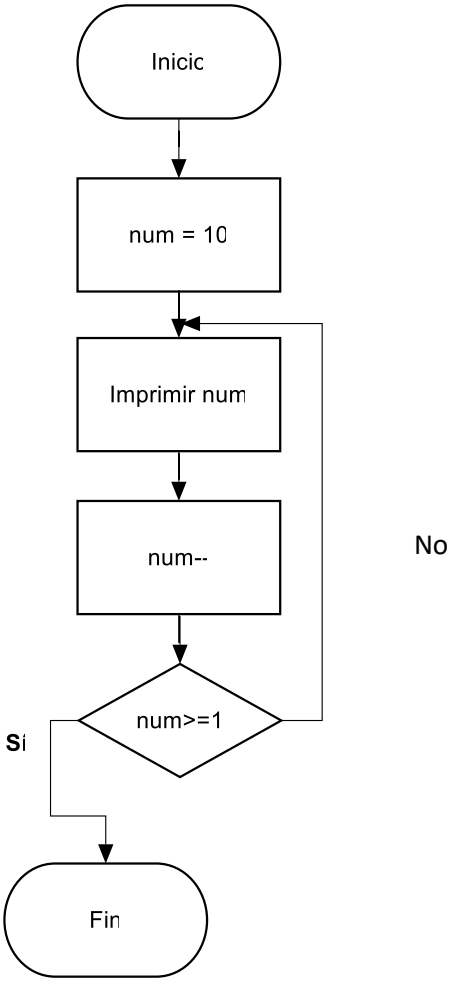
Para resolver este problema, comenzamos declarando una variable de inicio: inicio= ingresar numero y otra para almacenar los resultados: factorial. Dentro de do, la factorial se multiplicará por el numero ingresado y ese mismo va a decrementar para llegar a 1. La condición en sentencia while, es que el bucle no se detendrá hasta que numero sea mayor o igual a 1, imprimirá el resultado y terminara el proceso.


Inicio
num = 1
num 2 = num
fact = 1
fact *=num
num--
num>=1
Imprimir fact
Fin

5. Imprime los números del 10 al 1 en orden inverso.

- Problema: Imprimir los números del 1 al 10, inversamente.
- Datos de entrada: Se nos da un rango de 1 al 10 en el problema.
- Solución:
  1. Declarar una variable de inicio: numero =10;
  2. Abrir una sentencia do...while:
     Dentro, imprimir el número actual
     E decrementarlo para llegar a 1: numero--
  3. En la sentencia while: numero>=1.
- Salida: Mostrar los numero de

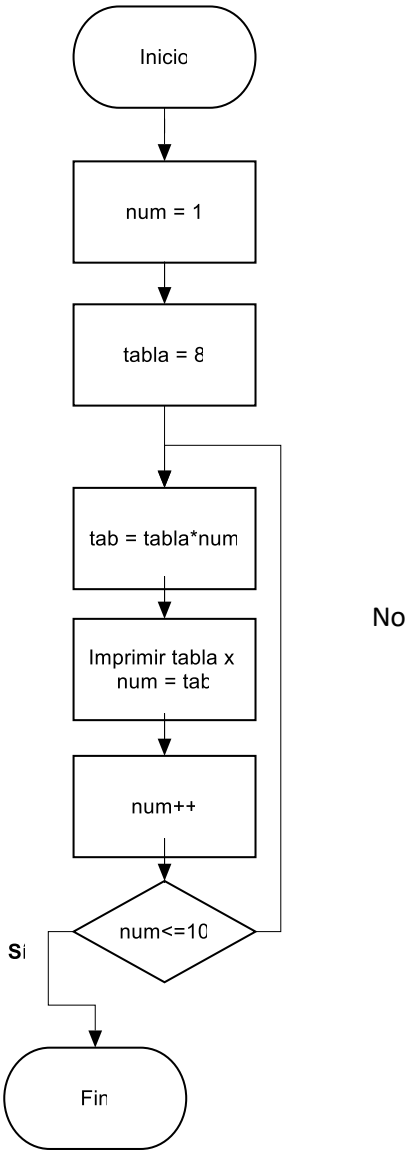
Se inicia declarando la variable de inicio: numero = 10. Se abre un bucle do, dentro del cual se imprime el número actual y se decrementa en 1 con cada iteración. La sentencia while verifica si el número es mayor o igual a 1, permitiendo que el bucle continúe mientras esta condición sea verdadera.



6. Imprime la tabla de multiplicar del 8.

- Problema: Imprimir la tabla del 8.
- Datos de entrada: El dato dado es que la taba es la del 8.
- Solución:
    1. Se declara una variable de inicio: numero = 1
    2. Se declara la variable de la tabla a multiplicar: tabla 8;
    3. Se inicia do:
       Dentro, se imprime “la tabla del 8 x números… es tabla*numero”
       Y se incrementa número inicial.
    4. En la sentencia while: la condición es que no pase de 10.
- Salida: Mostrar la tabla de multiplicar del 8.

Para esta cuestión, empezamos declarando el inicio en 1 y la tabla: 8; se inicia una sentencia do-while, en la parte de do, se imprime tabla*numero para

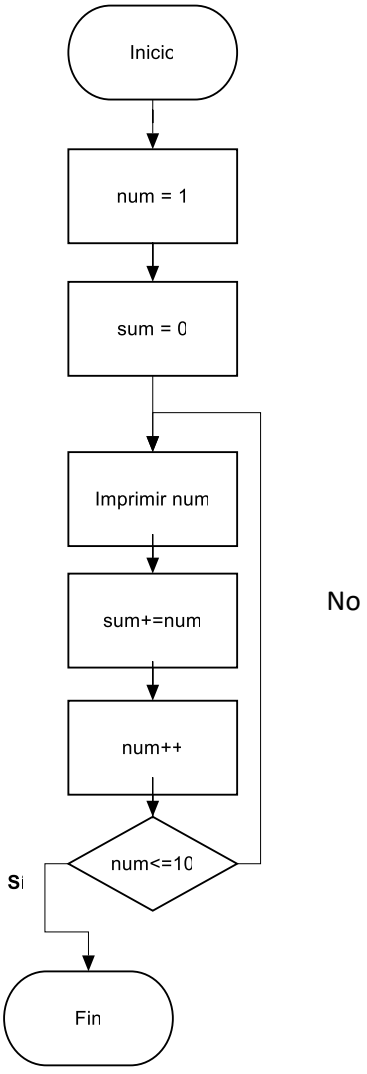


No

7. Suma los primeros 10 números.

- Problema: Sumar los primeros 10 números.
- Datos de entrada: Se nos da un rango de los 10 primeros números.
- Solución:
    1. Definir una variable: numero = 1, para que inicie en 1.
    2. Una segunda para almacenar resultado: suma = 0;
    3. Iniciar do while:
       Dentro de do: suma+=numero
       E incrementarlo: numero++
    4. En la parte de while: numero<=10, siempre que se cumpla esta condición, continuara le bucle.
- Salida: Mostrar la suma de los primero 10 números.

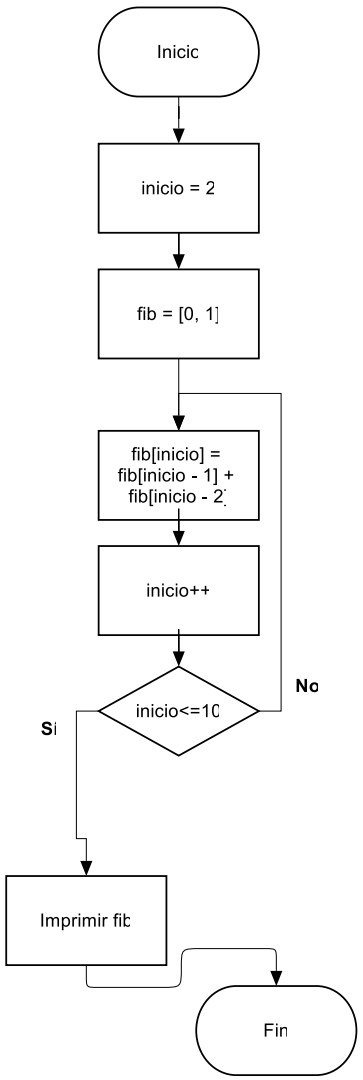
Con el rango de lo diez primeros números, comenzamos definiendo la variable inicial en 1 y suma para el resultado, comenzamos abriendo do imprimiendo el número actual, almacenándolo en la suma e incrementando numero inicial, así hasta que número inicial sea 10.



8. Imprime los primeros 10 términos de la serie Fibonacci.

- Problema: Imprimir los primeros términos de la serie Fibonacci.
- Datos de entrada: No hay datos manuales, se ingresan en el código.
- Solución:
    1. Declarar una variable de inicio: inicio = 2;
    2. Declarar un array con los dos primeros términos: fib = [0, 1];
    3. Comenzar la sentencia do- while:
       Dentro de do se aplica: fib[inicio] = fib[inicio - 1] + fib[inicio - 2].
       Y se incrementa: inicio++.
    4. En while, la condición es: inicio<=10.
- Salida: Mostrar los primeros 10 términos de la serie de Fibonacci.

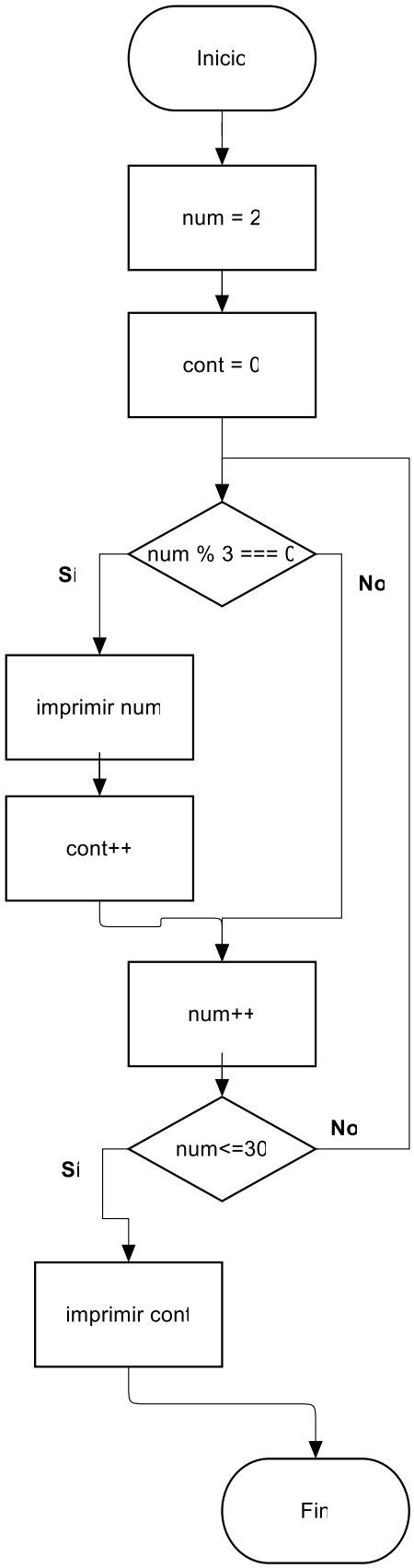
Para resolver este planteamiento, se inicia con una variable en 1 y un array con los dos primeros términos: 0 y 1; e abre un bucle do...while, donde se calcula cada nuevo término de la serie sumando los dos términos anteriores, se incrementa inicio y se verifica si es menor que 10 para continuar el bucle. Finalmente, se imprimen los términos resultantes.



9. Cuenta cuántos números entre 1 y 30 son divisibles por 3.

- Problema: Contar cuántos números entre 1 y 30 son divisibles por 3.
- Datos de entrada: Se trabaja con el rango definido de 1 a 30.
- Solución:
    1. Se declara una variable de inicio: numero = 1;
    2. Se declara una variable contador: contador = 0;
    3. Se inicia la sentencia do-while:
       Dentro de do, se agrega una sentencia if: numero % 3 === 0;
       Se imprime el número actual.
       Se incrementa el contador.
    4. En while: la condición es: nuemro<=30.
- Salida: Mostrar cuántos números entre 1 y 30 son divisibles por 3.

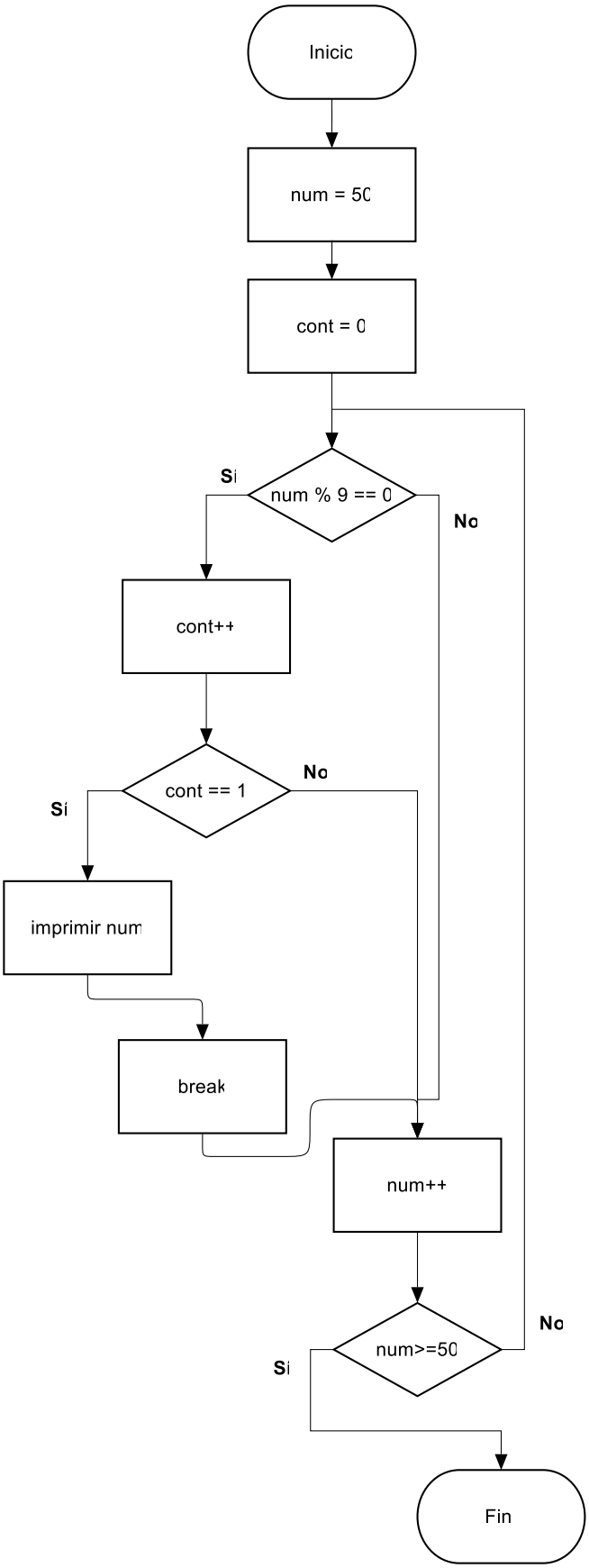
Se inicia con una variable para iniciar en 1: numero = 1 y una para el contador = 0. Se abre una sentencia do-while, donde se verifica si el número actual es divisible por 3. Si es así, se incrementa el contador. Finalmente, se incrementa el número y se verifica si es menor o igual a 30 para continuar el bucle. Al finalizar el bucle, se imprime el valor del contador.


Inicio
num = 2
cont = 0
num % 3 === 0
Si
No
imprimir num
cont++
num++
num<=30
No
Si
imprimir cont
Fin

10. Encuentra el primer número divisible por 9 después de 50.

- Problema: Encontrar el primer número después de 50, divisible entre 9.
- Datos de entrada: No hay datos manuales, se nos da un rango de 50 en adelante.
- Solución:
    1. Variable inicial: numero = 50;
    2. Variable para el contador: contador: 0;
    3. Sentencia do-while:
       Dentro de do: se agrega la condición: numero % 9 == 0, si se cumple, aumenta el contador.
       Después otra condición: contador == 1, si se cumple, imprime ese número y corta el bucle.
    4. En while: numero>=50.
- Salida: Mostrar primer número divisible por 9 después de 50.

Para encontrar el primer número divisible entre nueva después de 50, se inicia con una variable en 1 y otra en 0 para el contador. Con esto se inicia la sentencia do-while; dentro de do se se agrega la condición: numero % 9 == 0, si se cumple, aumenta el contador, después otra condición: contador == 1, si se cumple, imprime ese número y corta el bucle. En la parte de while, la condición es que el número inicial sea mayor o igual a 50, pero si el contador llega a incremar en 1, se corta el bucle, pues ya se encontró el número pedido.


Inicic
num = 50
cont = 0
num % 9 == 0
Si
No
cont++
cont == 1
No
Si
imprimir num
break
num++
num>=50
No
Si
Fin

Parte 6: Ejercicios combinados

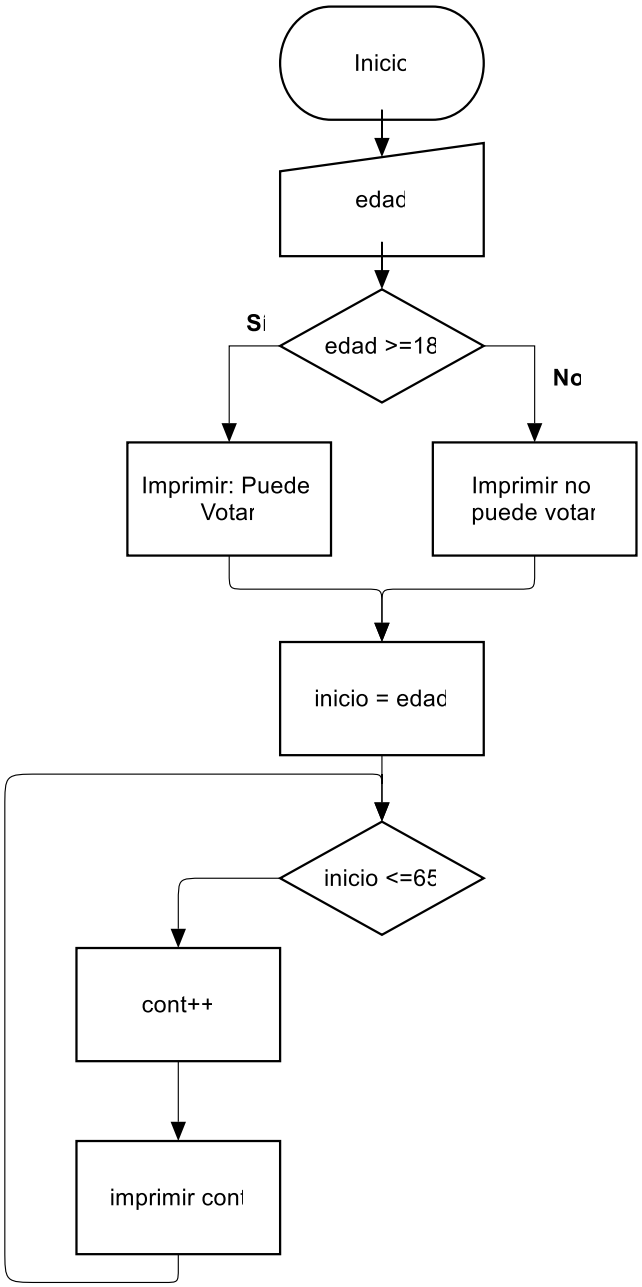
1. Realiza un programa que solicite al usuario su edad y dependiendo de su respuesta, indique si puede votar.

Luego, usa un ciclo for para imprimir los años que le faltan hasta la edad de retiro (65 años).

- Problema: Realiza un programa que solicite al usuario su edad y dependiendo de su respuesta, indique si puede votar.
- Datos de entrada: La edad de usuario, ingresada por el mismo.
- Solución:
    1. Leer edad: ingresada por el usuario.
    2. Con ayuda de If: se verifica si es mayor de 18 años: edad>=18
       Si se cumple, se muestra que puede votar.
    3. Se inicia un ciclo for :
       Primero, una variable en -1, porque for empieza a contar desde 0.
       Se inicia for, con un inicio=edad; el inicio debe ser menor o igual que 65; y se incrementa: inicio++.
       Con cada iteración, el contador aumenta 1, hasta que la condición en for ya no se cumpla.
- Salida: Mostrar si puede votar y cuantos años le falta para llegar a los 65.

Se inicia pidiendo la edad del usuario y almacenándola en una variable, con una sentencia if se verifica si es mayor de edad, si es así, se muestra que puede votar.

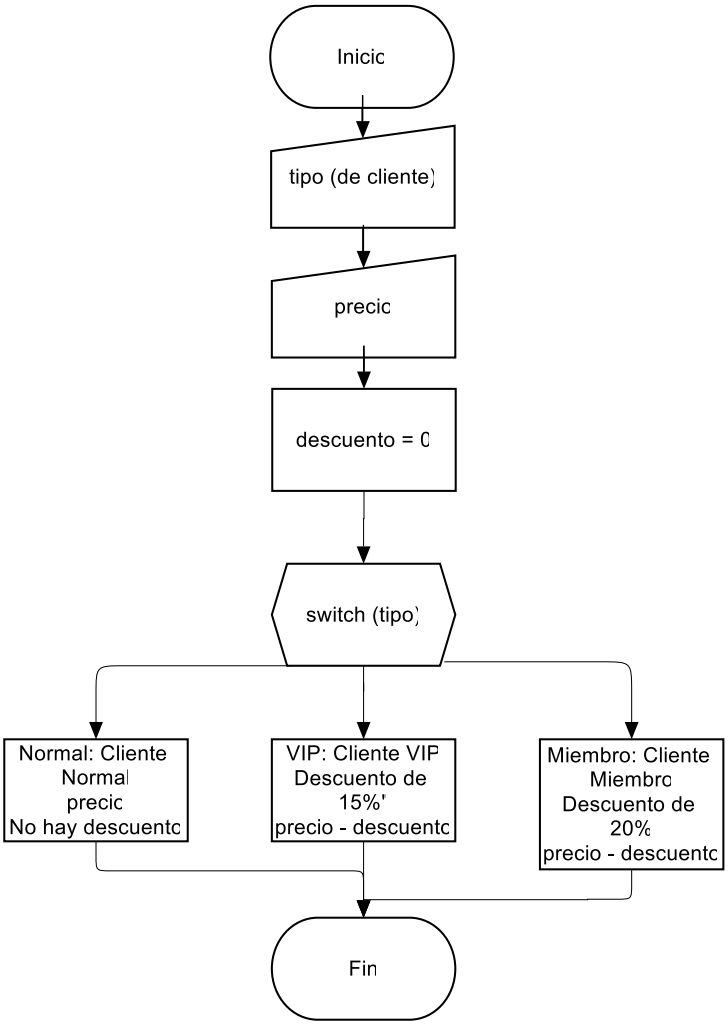
Para saber cuanto le falta para los 65, se inicia un contador en -1; se abre ciclo for con inicio en la edad dada, que no pase de los 65 y se incrementa, con cada iteración de for, el contador incrementa 1 hasta que la condición ya no se cumple, muestra resultado y termina el proceso.


Inicic
edad
Si
edad >=18
No
Imprimir: Puede Votar
Imprimir no puede votar
inicio = edad
inicio <=65
cont++
imprimir cont

2. Crea un programa que calcule el total a pagar en un restaurante. Elige el tipo de cliente (normal, VIP, miembro) usando switch y aplica un descuento en base a eso.

- Problema: Crea un programa que calcule el total a pagar en un restaurante. Elige el tipo de cliente (normal, VIP, miembro) usando switch y aplica un descuento en base a eso.
- Datos de entrada: Total del producto y tipo de cliente.
- Solución:
    1. Leer total de compra: ingresado por el usuario.
    2. Leer tipo de miembro: ingresado por el usuario.
    3. Iniciar ciclo switch: funciona con tipo de cliente.
       Si el cliente coincide con algún tipo ya registrado: se muestra el tipo de cliente, el descuento y el precio después del descuento.
- Salida: Mostrar el tipo de cliente, el descuento y el precio después del descuento.

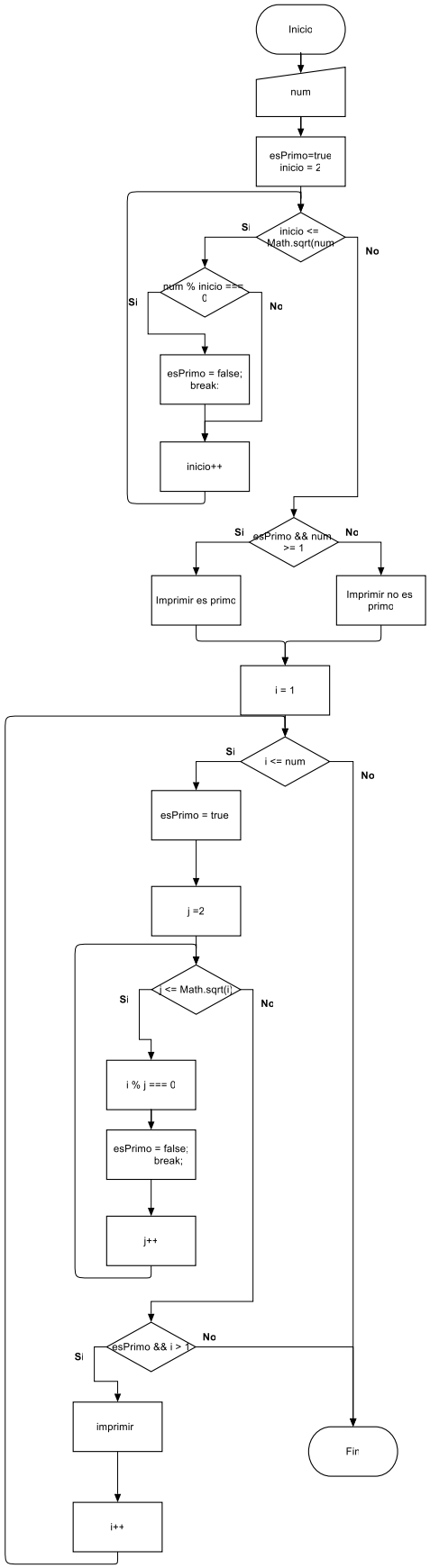
Este planteamiento es sencillo, pues se pide el precio total y el tipo de cliente. Con esa información, switch busca que coincida con una de sus opciones y muestra el texto correspondiente, hecho esto, termina el proceso.



3. Escribe un programa que determine si un número es primo utilizando un ciclo while. Luego, imprime todos los números primos entre 1 y el número dado.

- Problema: Determinar si un numero es primo y mostrar lo que hay entre el mismo y 1.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:
    1. Leer número: ingresado por el usuario.
    2. Inicializamos una variable esPrimoNum en true.
    3. Usamos una sentencia while con la condición si es divisible desde 2 hasta la raíz cuadrada del número.
       Si hay un divisor, se cambia true a false.
    4. Para imprimir el resultado, se usa if-else:
       Si se cumple, (esPrimo && numero ingresado >= 1, se imprime que es primo, de lo contrario, se mostrará que no es primo.
    5. Para imprimir los demás números, se hace lo mismo que al principio parar ver si era un numero primo, pero la diferencia es que la instrucción para imprime está en iteración, porque nos da los números restantes.
- Salida: Imprimir si el número ingresado es primo o no y todos los números primos entre 1 y el número ingresado.

Con while, se comprueba si el número ingresado es divisible entre otor dando como resultado un entero, si esto pasa, el número no es primo, y si es lo contrario, es primo. Esto se usa tanto para verificar si un número es primo como para imprimir las interacciones que son los números primos entre 1 y el número dado.


Inicio
num
esPrimo=true
inicio = 2
Si
inicio <=
Math.sqrt(num
No
num % inicio ===
0
Si
No
esPrimo = false;
break;
inicio++
Si
esPrimo && num
>= 1
No
Imprimir es primo
Imprimir no es
primo
i = 1
Si
i <= num
No
esPrimo = true
j =2
Si
j <= Math.sqrt(i)
No
i % j === 0
esPrimo = false;
break;
j++
Si
esPrimo && i > 1
No
imprimir
i++
Fin

4. Realiza un programa que determine el signo zodiacal basado en el día y mes de nacimiento del usuario utilizando if-else anidados.

- Problema: Determinar el signo zodiacal de una persona utilizando if-else anidados.
- Datos de entrada: El día y mes de nacimiento del usuario, ingresado por el mismo.
- Solución:
    1. Se lee el día y el mes: ingresadas por el usuario.
    2. Inicia la sentencia if, si se cumple el mes y el día, se imprime el signo,
       Si no, se pasa a la siguiente opción hasta que se cumple
    3. Si no se cumple ninguna, se muestra que no es un signo.
- Salida: El signo de la persona.

Este es un caso sencillo, se pide el día y mes, con eso, si coincide con algún rango de las opciones if-else, se imprime el signo correspondiente y termina el proceso.

5. Escribe un programa que permita ingresar una serie de calificaciones y determine la cantidad de aprobados y reprobados usando while y if-else.

- Problema: Escribe un programa que permita ingresar una serie de calificaciones y determine la cantidad de aprobados y reprobados usando while y if-else.
- Datos de entrada: Calificaciones, ingresadas por el usuario.
- Solución:
    1. Iniciar un contador de aprobados, otra de reprobados y otra de falso/verdadero.
    2. Leer calificaciones.
    3. Si la calificación es menor, resulta falso y rompe el ciclo.
    4. Si la calificación es mayor o igual de 70 y menor o igual de 100, es aprobatorio e incrementa al contador de aprobados.
    5. Si no, es reprobatorio y aumenta el contador de reprobados.
- Salida: Mostrar la cantidad de aprobados y reprobados y cuales calificaciones son buenas y cuáles no.

Este caso, pide mostrar las calificaciones aprobatorias y reprobatorias y la cantidad de cada una.

Se comienza iniciando un contador de aprobado y otra de reprobados, se pide la calificación, si es mayor o igual de 7 y menor o igual de 100 es aprobatoria y aumenta el contador de aprobados, si no, es reprobatoria y aumenta el contador de reprobados, se debe mostrar que tipo de calificación es ingresada y mostrar la cantidad de los contadores y termina el proceso.

6. Haz un programa que simule un cajero automático, permitiendo varias transacciones con switch y repitiendo el proceso hasta que el usuario decida salir.

- Problema: Simular un cajero automático que no pare hasta que el usuario decida salir.
- Datos de entrada: Monto de ingresado, monto retirado.
- Solución:
    1. Se declara, una variable de saldo y otra de verdadero/falso, para el continuar.
    2. Se inicia ciclo do-while: para que se repita el proceso.
       Dentro de do, se inicia un switch para poder elegir la opción deseada.
       Si se elige depositar, se pide cuanto se ingresa, se suma el monto a saldo y se muestra el saldo actual.
       Si se pide retirar, se aplica algo similar porque es restando del saldo
       Y si se pide ver saldo, solo se muestra el saldo disponible.
    3. En while, la condición es que el proceso se repite mientras el usuario ingrese el salir.
- Salida: Poder ingresar o retirar y ver el saldo disponible.

Este fue un caso, en que busque ayuda externa. Para solucionarlo, primero se declara una variable para el saldo y otro para poder continuar. Se abre un ciclo do-while para que le proceso se repita a menos que el usuario solicite salir. En do se inicia un switch, las opciones están el ingresar, retirar o ver saldo, si se elige salir, se rompe el ciclo y termina el proceso.

7. Implementa un programa que imprima los múltiplos de 3, 5 y 7, usando un for. Si un número es múltiplo de varios, indícalo.

- Problema: Imprimir los múltiplos de 3, 5 y 7, usando un for. Si un número es múltiplo de varios, indícalo.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:
    1. Leer número: Ingresado por el usuario.
    2. Declarar una variable para el texto.
    3. Iniciar ciclo for: Inicio = 1; inicio <= numero ingresado y que inicio ++.
       Dentro, la primera sentencia if: inicio % 3===0, se añade al texto: 3;
       la segunda sentencia if: inicio % 5 ===0, se añade al texto: 5;
       y la tecera sentencia if: inicio % 7 ===0, se añade al texto: 7.
    4. La cuarta sentencia if se encarga de mandar el mensaje si un numero es múltiplo de 3, 5 y 7.
- Salida: Mostrar los múltiplos de 3, 5 y 7 y también si comparten alguno.

Aquí me di cuenta que no solo se pueden “sumar” números, también se puede concatenar un texto directo a una variable como si fueran números.

La solución es iniciar el ciclo for con un inicio en 1, que el mismo no se pase del numero dado y se incremente. Dentro se declara una variable booleana (también supe que así se denominan a este tipo), y la que contendrá el texto, se inician tres sentencias if (puse tres para más seguridad que pase el número por cada una) y suman el numero al texto de las sentencias que se cumple, al final se imprime si el numero ingresado es múltiplo de alguno o comparte y termina el proceso.

8. Realiza un programa que pida 5 números y los ordene de mayor a menor usando un ciclo for y if.

- Problema: Dado cinco números, ordenarlos de mayor a menor utilizando un ciclo for y condicionales if.
- Datos de entrada: Cinco números enteros dados por el usuario.
- Solución:
    1. Se abre arreglo para almacenar los números: números [];
    2. Ciclo for: pedir números ya almacenarlos en el arreglo;
    3. Se usan dos ciclos for:
       El primero: para controlar el paso de los números ingresados.
       El segundo: para comparar pares de números.
       Se agrega una sentencia if para decidir si un número es mayor que otro.
- Salida: Mostrar los números ordenados de mayor a menor.

Este planteamiento es uno de los que más me ha costado. Se nos pide que lo resolvamos con un ciclo for y con if.

Se inicia pidiendo los números y almacenado los en un arreglo. Después, se inicia un ciclo for que controla el paso de los números ingresados, anidado, se abre otro ciclo for, que comprara el número resultante de la sentencia for anterior y le suma uno, para dar el número siguiente; con esos dos números, una sentencia if compara el número inicial y el que le sigue, si el inicial resulta ser mayor, se intercambian valores para ordenarse e imprime el resultado, cuándo se pasan todos los números es que hay en el arreglo, termina el proceso.

9. Crea un programa que solicite una contraseña al usuario y permita tres intentos. Si falla, muestra un mensaje de bloqueo, usando do-while.

- Problema: Crear un programa que pida una contraseña, si se intenta más de tres veces, bloquear las oportunidades.
- Datos de entrada: Una contraseña dada por el usuario.
- Solución:
    1. Leer contraseña: dada por el usuario;
    2. Iniciar contador en 0
    3. Iniciar variable booleana en false.
    4. Iniciar do:
       Dentro, se pide que se ingrese la contraseña
       Y se compara con al originas con sentencia if, si es igual, se rompe el ciclo y se muestra “Bienvenido”.
       Si no, el contador aumentara 1 y se mostrara que la contraseña es incorrecta y cuantos intentos faltan.
       Si los intentos llegan a 3, se
    5. En while, la condición el bucle se repita hasta 3 veces.
- Salida: Mostrar bloqueado si fueron 3 intentos fallidos.

La solución a esta cuestión es: comenzar declarando la variable de la contraseña originas, el contador de intentos y una booleana para saber cuando se bloquean los intentos. Se inicia el ciclo do-while que pide la contraseña, con sentencia if se compara si lo ingresado es igual a la contraseña original, si se cumple Se muera “Bienvenido” y se corta el ciclo, si no se muestra que la contraseña es incorrecta y cuantos intentos le quedan, si pasa de los tres intentos, se imprime bloqueado y no deja volver a ingresar la contraseña, y termina el proceso.

10. Desarrolla un programa que determine si un número ingresado es perfecto (un número es perfecto si es igual a la suma de sus divisores propios) usando un ciclo for.

- Problema: Determinar con ciclo for si un número es perfecto.
- Datos de entrada: Un número ingresado por el usuario.
- Solución:
    1. Leer número: ingresado por el usuario.
    2. Declarar variable booleana en true: esPerfecto y otra para almacenar resultado.
    3. Iniciar for: inicio en 1, que llegue hasta la mistad del número, pues los números que se suman no pasan de la mitad e incrementa el inicio.
       Dentro, una sentencia if realiza numero % inicio === 0, si es verdadera, guarda ese número en suma y los suma con el valor que tiene.
    4. Sentencia if-else: Si suma es igual al número, se imprime que el número es perfecto, de lo contrario se muestra que no es perfecto.
- Salida: Mostrar si el número es perfecto.

Se empieza pidiendo un número, declarando una variable booleana en true y una variable que almacena el resultado, el ciclo for se inicializa en 1 y se limita a llegar a la mitad del número ya que el valor sus divisores nos pasan de la mitad y que incremente el inicio, si el modulo del número entre el inicio es 0, el valor de inicio se almacena en la variable suma y los suma con el valor que tenga. Para mostrar el resultado, una sentencia if comprueba que la suma sea igual que el numero ingresado y termina el proceso.