



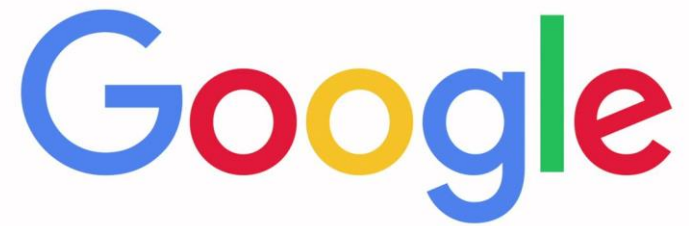
Módulo 4

Algoritmos y Diagramas de Flujo








¿Qué es un algoritmo?



Algoritmo

Un algoritmo es una serie de pasos que definen las instrucciones para realizar algún proceso. Un algoritmo debe ser:

- **Preciso:** Debe seguir un orden establecido. 
- **Definido:** Los resultados son predecibles, dos ejecuciones del mismo algoritmo deben generar el mismo resultado. 
- **Finito:** Debe tener un número determinado de pasos, y terminar en un tiempo finito. 

FULL WINDSOR KNOT

How to Tie a Necktie - 3 of 17



Algoritmo para preparar hot cakes



- 1. Mezclar harina, huevos, leche en un recipiente.**
- 2. Batir hasta generar una mezcla uniforme**
- 3. Calentar sartén**
- 4. Verter mezcla en sartén**
- 5. Voltear el hot cake**
- 6. Servir**

Al seguir un algoritmo al pie de la letra, el ejecutor no necesita conocer el por qué algo funciona.



Diseño de algoritmos

La computadora no debe entender el algoritmo, mientras pueda ejecutarlo.

Un buen **programador** debe dominar todos los aspectos y detalles del algoritmo para poderlo programar.

El trabajo de un programador es **convertir las instrucciones de un algoritmo en código.**

***Diseñar un
algoritmo es
trabajo altamente
creativo***

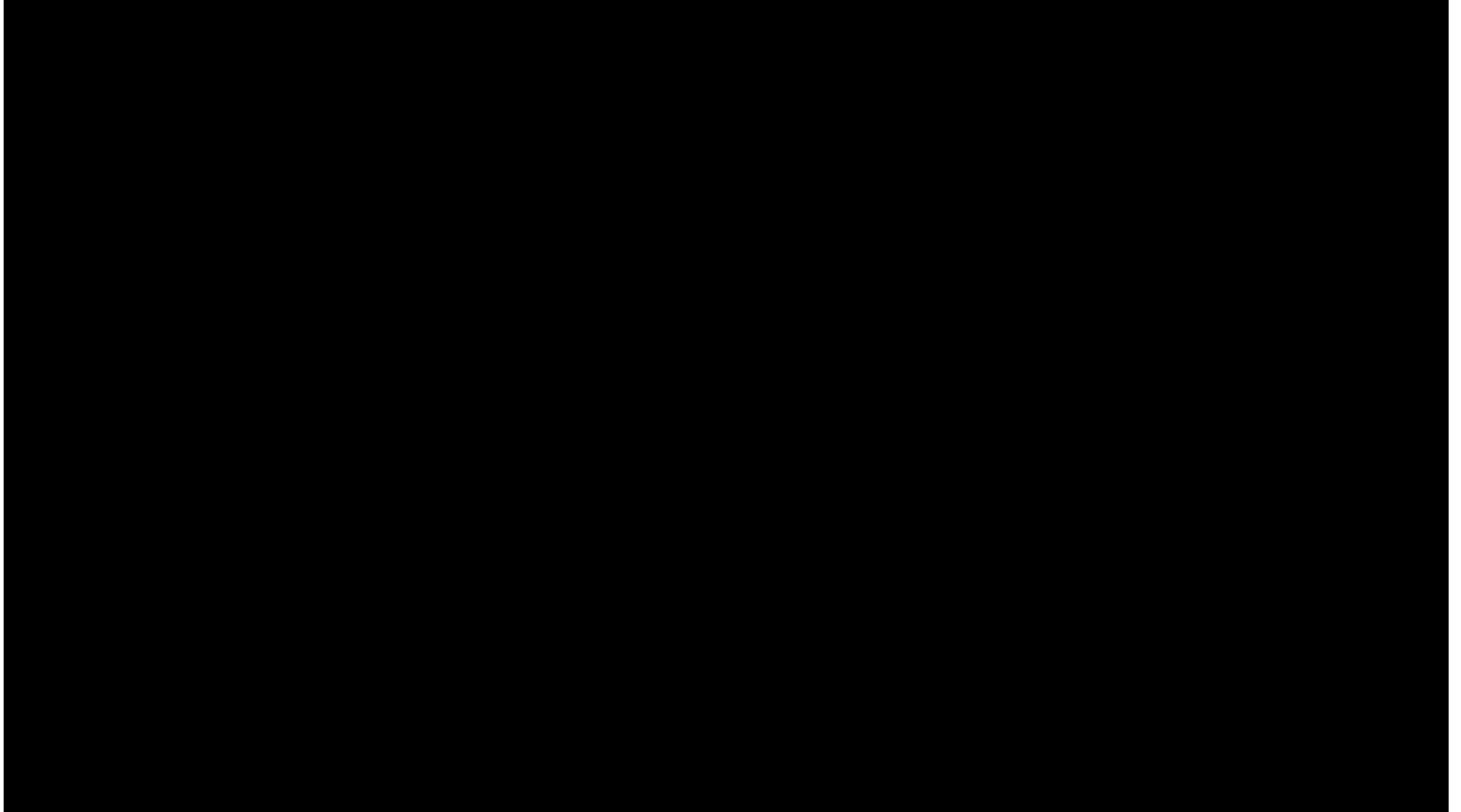


***¡Es detallar algo
complicado de una
forma lógica y
sencilla!***

- ¿Cuánta harina?
- ¿Cuántos huevos?
- ¿Cuánta leche?
- ¿En qué recipiente?
- ¿Cuánto tiempo hay que batir la mezcla?
¿Cómo? ¿En qué sentido?
- ¿Cómo prendo el sartén?
- ¿A qué temperatura caliento el sartén?
- ¿Dónde está el sartén?
- ¿Cómo vierto la mezcla? ¿Por cuánto tiempo?
- ¿De qué tamaño es el hot cake?
- ¿En dónde lo sirvo?



Robot makes The Perfect Burger?



¿Cómo podemos representar un algoritmo computacional?

**Diagrama de
flujo**

Pseudocódigo



Diagramas de Flujo

THE FRIENDSHIP ALGORITHM

DR. SHELDON COOPER, Ph.D

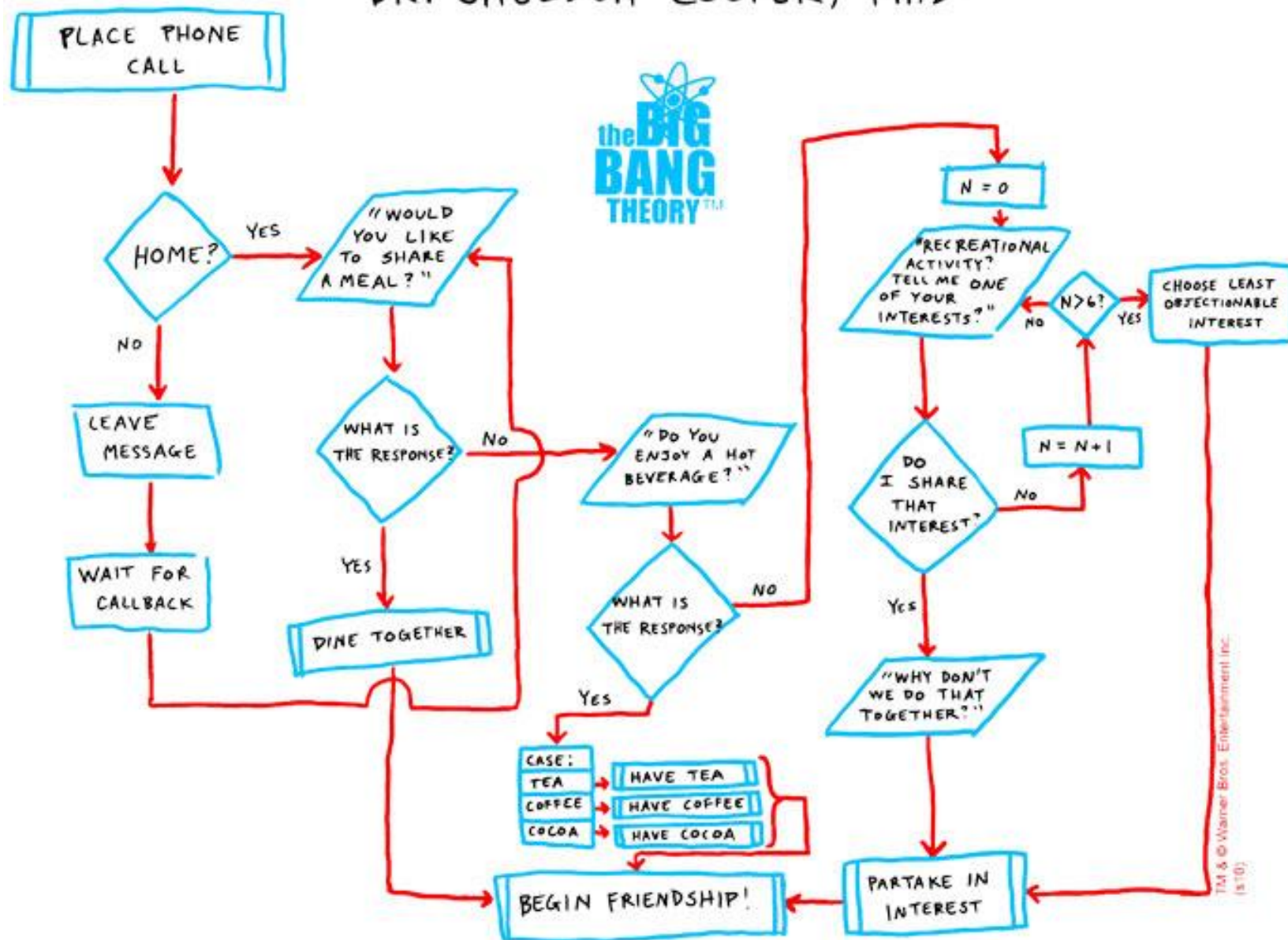


Diagrama de Flujo

Es una representación gráfica de un algoritmo. Utiliza figuras para representar acciones, decisiones y caminos. Todos los diagramas deben tener:

1. Un punto de inicio
2. Un punto fin
3. Sencillo de entender

La dirección en la que se ejecuta el diagrama del flujo está indicado por el sentido de los conectores, a través de una flecha →.

Bloque Inicio / Fin

Señalizan el principio y final de un diagrama de flujo.

Sólo puede existir uno de cada uno de estos bloques por diagrama.

Se utiliza una óvalo o círculo para representar esta acción



Inicio



Fin

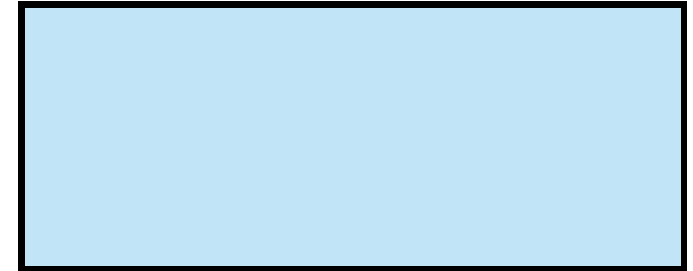
Bloque Proceso

Indica algún cálculo, operación, o procedimiento. Esto puede ser expresado en una expresión matemática o frase.

Se utiliza un rectángulo para indicar esta acción.

Por ejemplo:

- **Días = años * 365**
- **Metros = centímetros / 100**
- **Calcular total a pagar**



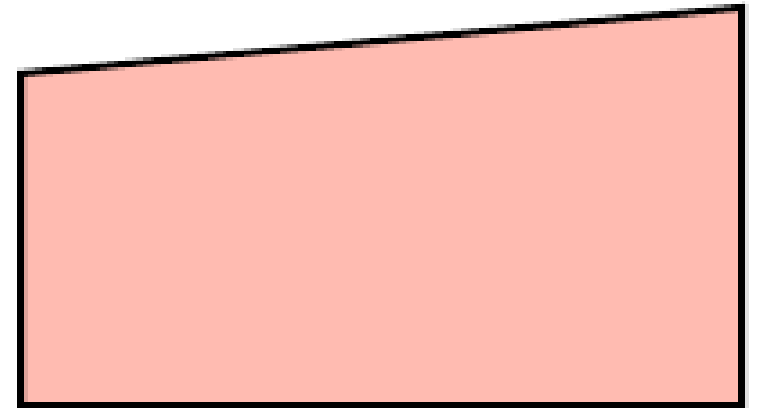
Bloque Entrada de Datos

Indica que se recibe alguna información de alguna fuente externa. Esto puede ser un teclado, un mouse, algún sensor, reloj, etc. Debe mencionarse el dato que será leído.

Se utiliza una figura combinación entre rectángulo y rombo.

Por ejemplo:

- Leer temperatura
- Leer cantidad de años



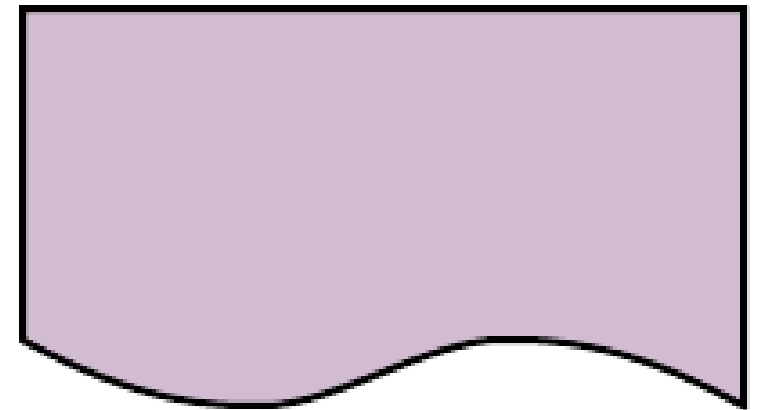
Bloque Salida de Datos

Indica que se enviará o desplegará alguna información a algún dispositivo de salida. Esto puede ser una pantalla, alguna consola, algún foco, etc. Debe mencionarse el dato al que se le dará salida.

Se utiliza un rectángulo con ondas en la base inferior.

Por ejemplo:

- **Print hora**
- **“La temperatura es: ” + temperatura**
- **Output 4 * 20.**



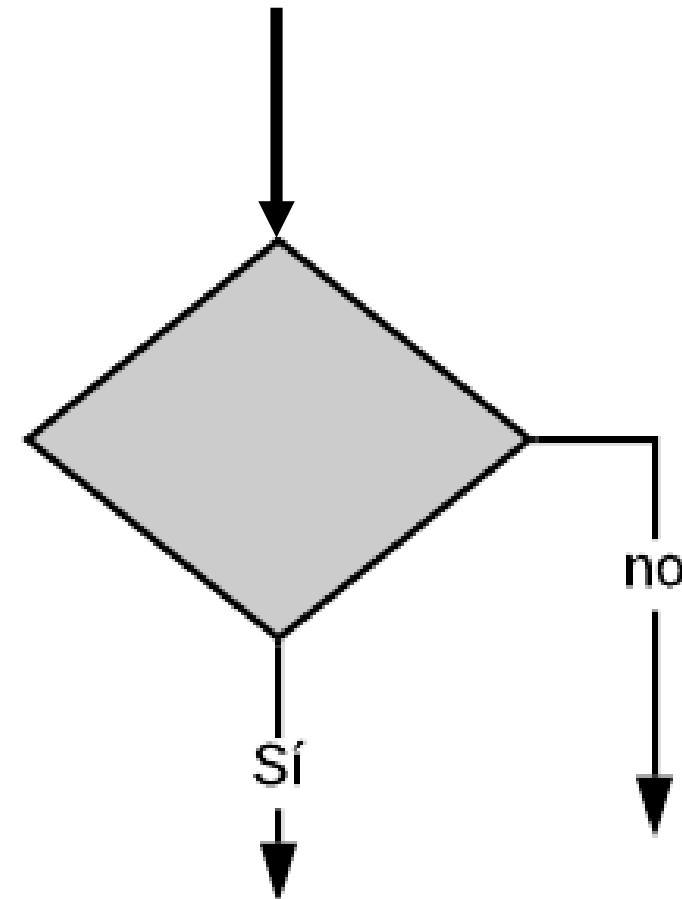
Bloque Decisión

Este bloque indica la bifurcación de un camino en dos dependiendo de una condición booleana. Sirve para ejecutar caminos condicionalmente. Dentro del bloque, deberá haber alguna expresión verdadera o falsa.

Se representa a través de un rombo.

Por ejemplo:

- **IF dia == lunes**
- **Saldo en cuenta bancaria mayor que cero**
- **Luz está encendida**



Ejemplo

Calcular la nómina de un trabajador que trabaja por horas.

Entradas:

→ **Sueldo por hora**

→ **Horas trabajadas**

Salidas:

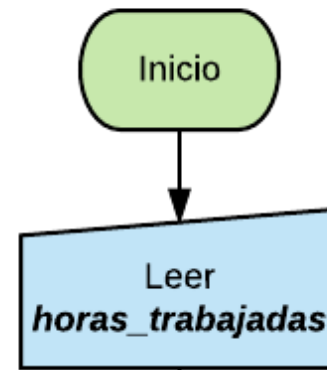
← **Monto a pagar**





Inicio

Comenzamos con el
bloque de inicio.



Leemos la cantidad de horas trabajadas.

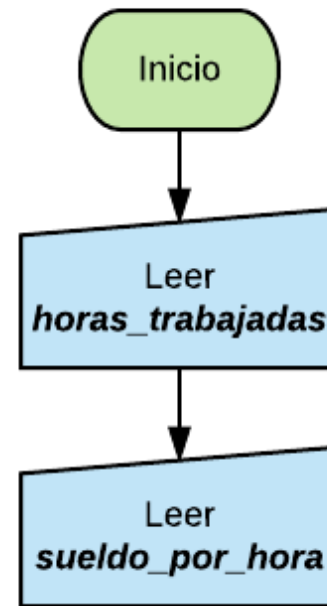
Estas pueden ser:

8, 10, 20, 40.

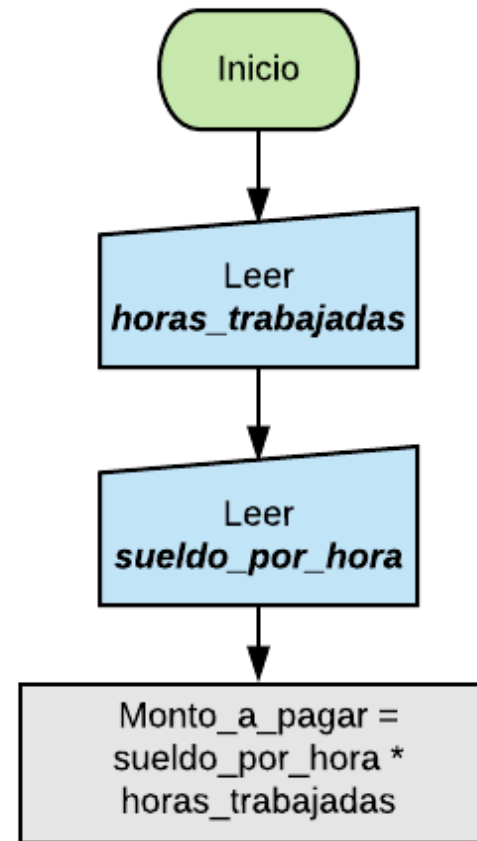
Leemos el sueldo por cada hora laborada.

Este puede ser:

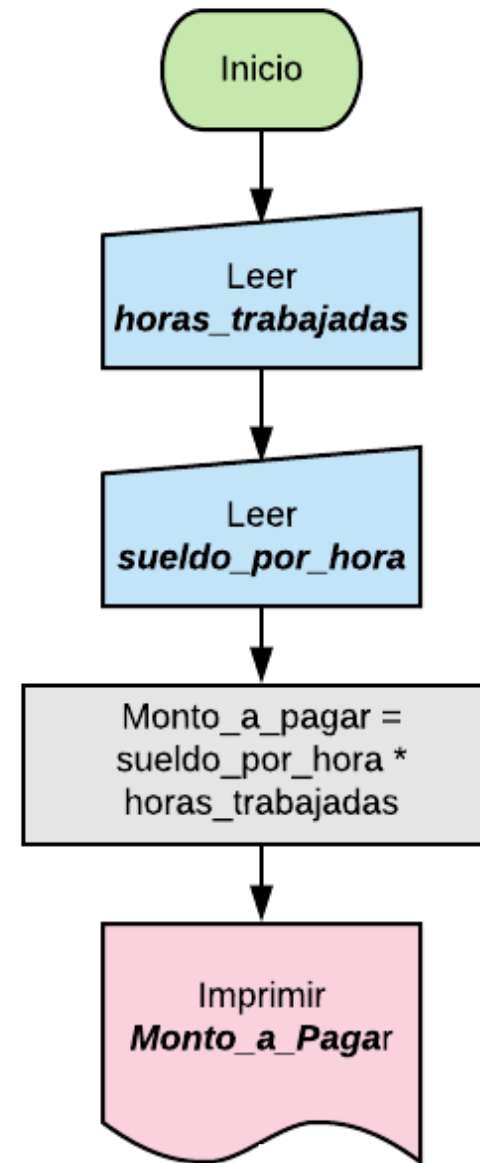
- 50
- 65.50
- 100
- 1000



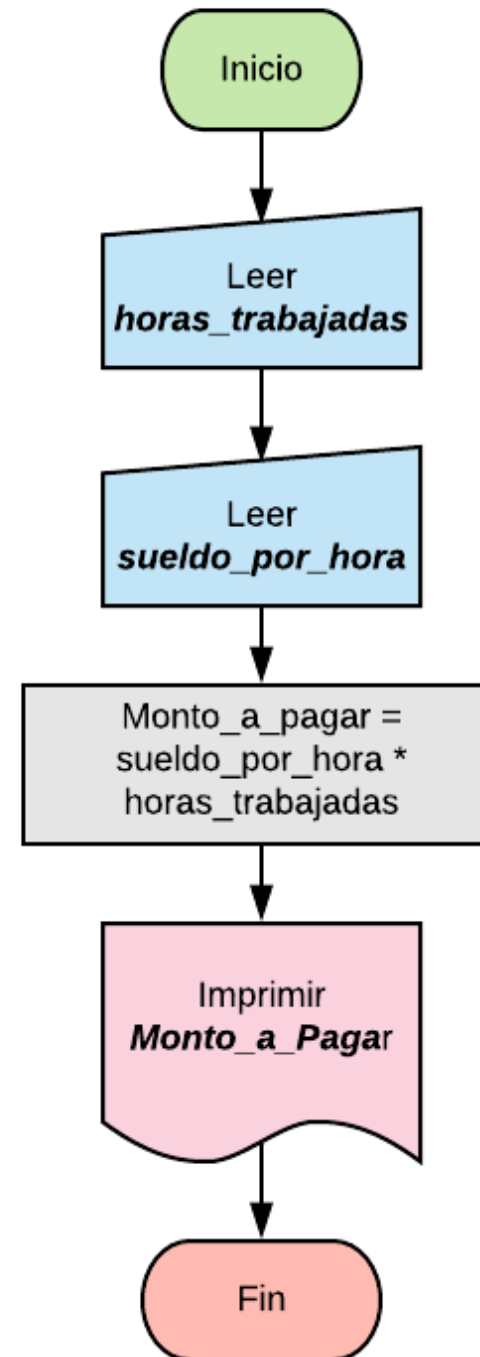
Calculamos el monto a pagar
en un **bloque de proceso**.



Imprimimos el resultado del cálculo con un **bloque de salida**.



Terminamos con un bloque de **Fin**.



Inicio

Ingresamos 40 horas
trabajadas en la semana
utilizando el teclado

Ingresamos un sueldo de 150
utilizando el teclado

$\text{Monto_a_pagar} = 40 \times 150$
 $\text{Monto_a_pagar} = 600$

En consola aparece
600

Ejemplo

Calcular la nómina de un trabajador que trabaja por horas.

Entradas:

→ **Sueldo por hora**

→ **Horas trabajadas**

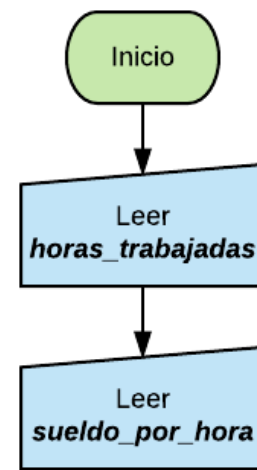
Salidas:

← **Monto a pagar**



¡NUEVA REGLA!

Si la persona trabaja mas de 40 horas, ganará un bono de 100 pesos.



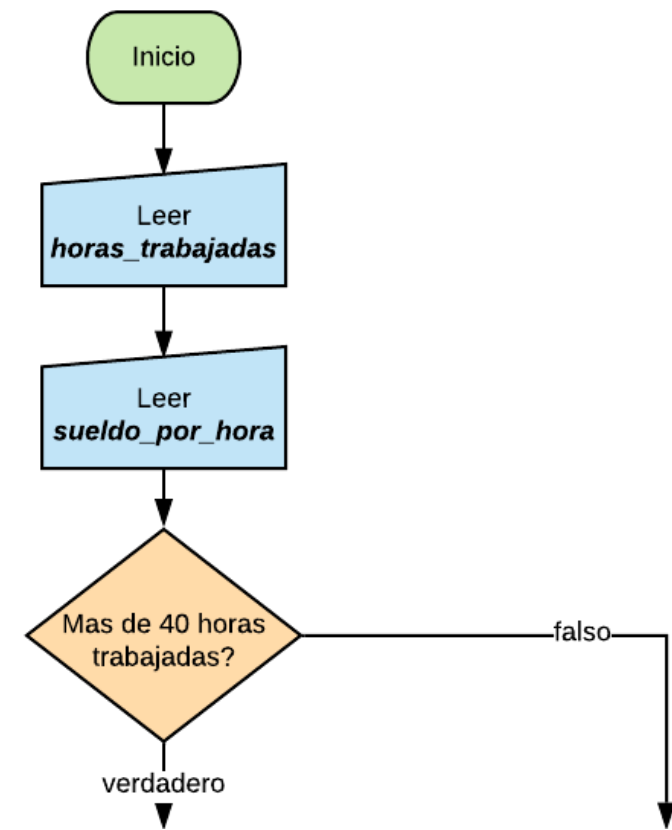
Recibimos de consola
horas_trabajadas y sueldo_por_hora.

Haremos dos caminos:

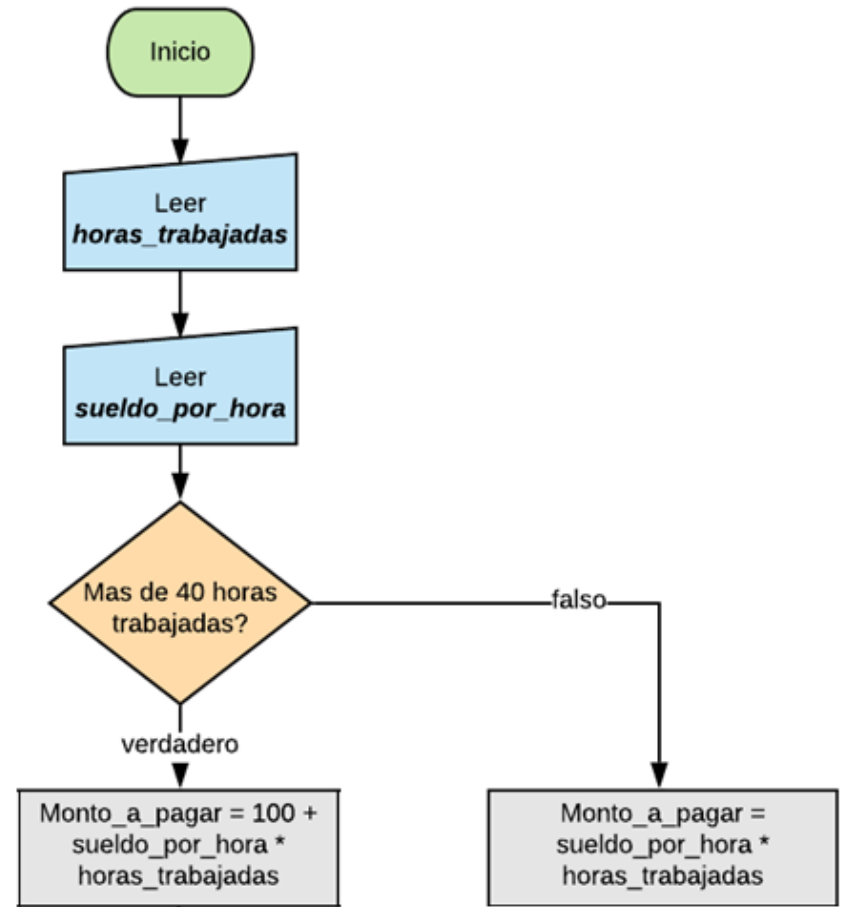
1. Cuando la persona haya trabajado mas de 40 horas
2. Cuando la persona haya trabajado menos de 40 horas

Ejemplos

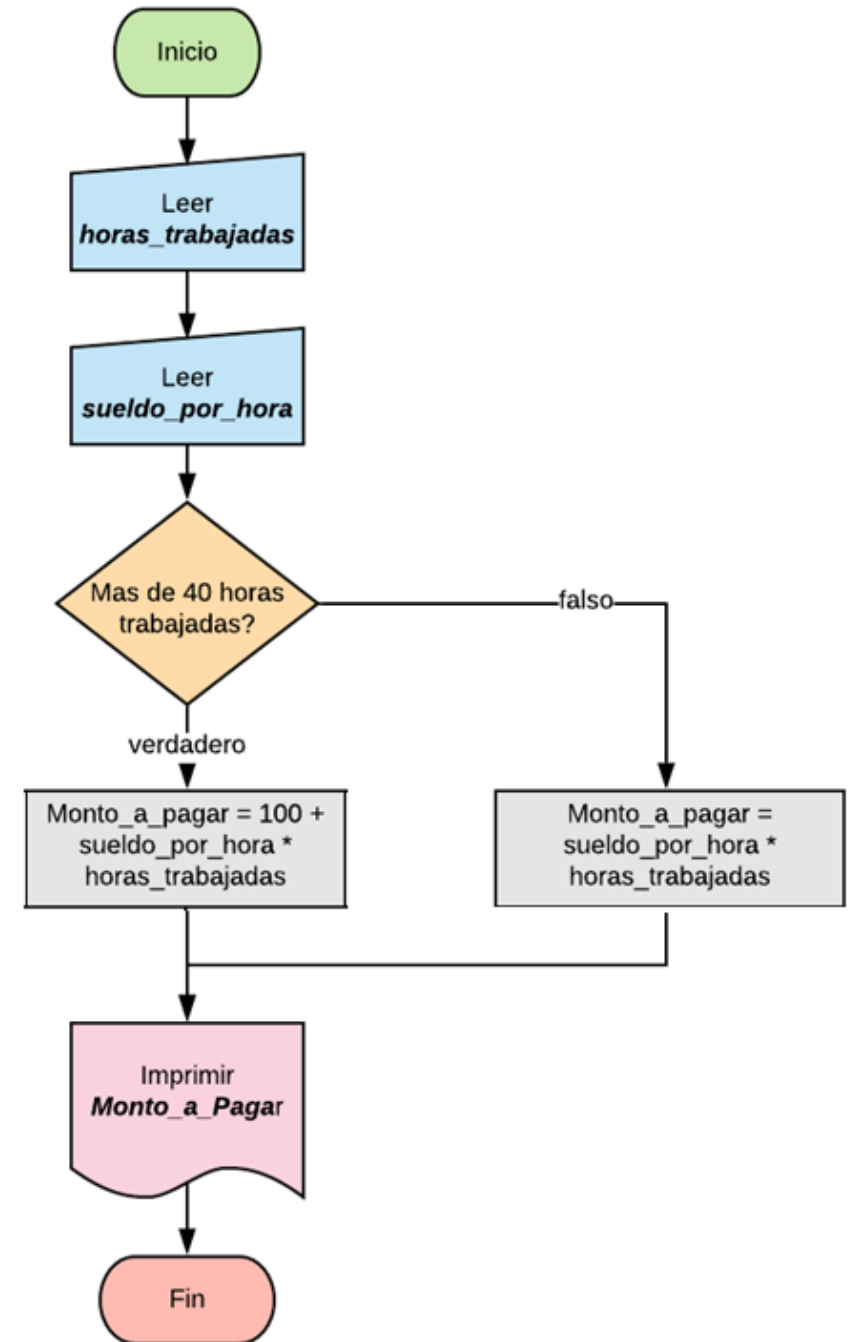
- 30 horas = falso
- 100 horas = verdadero
- 0 horas = falso
- 40 horas = falso



Cada camino realiza el cálculo correspondiente.



Se imprime el valor de
Monto_a_Pagar y terminamos



Tipos de Variables

Tipo	¿Qué puede almacenar?	Ejemplos:
int	Números enteros positivos o negativos.	10 -4 543
double	Números de punto flotante. Cualquier número positivo, negativo, y con algún componente decimal.	10.0 -4.58 0.000056
char	Caracteres individuales en codificación ASCII. Los caracteres deben estar rodeados por comillas sencillas.	'a' 'Z' '!' '9'
String	Un string debe estar encapsulado por comillas dobles (quotes).	"Hola Mundo!" "Hay 126.74 millones de habitantes en México".
boolean	Sirve para guardar valores binarios: verdadero o falso.	true false

Operadores lógicos y relacionales

Operador	¿Para qué sirve?	
==	Igual a	mes == 12 letra == 'c'
!=	Es diferente a	mes != 12 letra != 'c'
>=	Mayor o igual a	Saldo >= 100
>	Mayor a	Saldo > 100
<=	Menor o igual a	Saldo <= 0
<	Menor a	Saldo < 1000
&&	AND	(mes >= 1) && (mes <= 12)
 	OR	(letra == 'c') (letra == 'a')
!	NOT	!(mes == 12)

Ejercicio

Realiza un diagrama de flujo que permita calcular el costo de tu bebida en Starbucks.

	Capuccino	Frappuccino
Costo base	\$50.00	\$70.00
Agrandar la bebida (Venti)?	+ 20%	+ 25%
Con crema batida?	\$10.00	\$10.00
Con popote de galleta?	N/A	\$20.00



Ejercicio en Progranimate



Pseudocódigo

Pseudocódigo

El pseudocódigo es una descripción de alto nivel de un algoritmo. Utiliza ciertas convenciones de la programación, pero la intención es que sea **entendible para los humanos**.

El pseudocódigo generalmente se utiliza para:

- **Describir cómo funciona** un algoritmo.
- Explicar un proceso computacional a **gente no técnica**.
- Diseñar **código en grupo**.



Lenguaje

Utilizamos lenguaje lógico y algunas convenciones de programación, pero omitimos detalles que compliquen el código.

Podemos utilizar frases como:

- Sumar 1 a *variable*
- Multiplicar saldo * 10
- (IF) Si la luz está encendida, entonces...
- (ELSE) De lo contrario...

Vaso de Agua

El contenido de un vaso de agua lo vamos a representar a través de su volumen en mililitros.

- Es decir, un vaso con 0 mililitros está vacío

Diseña un algoritmo que solicite el tamaño del vaso y lo llene de agua.

Represéntalo en pseudocódigo

