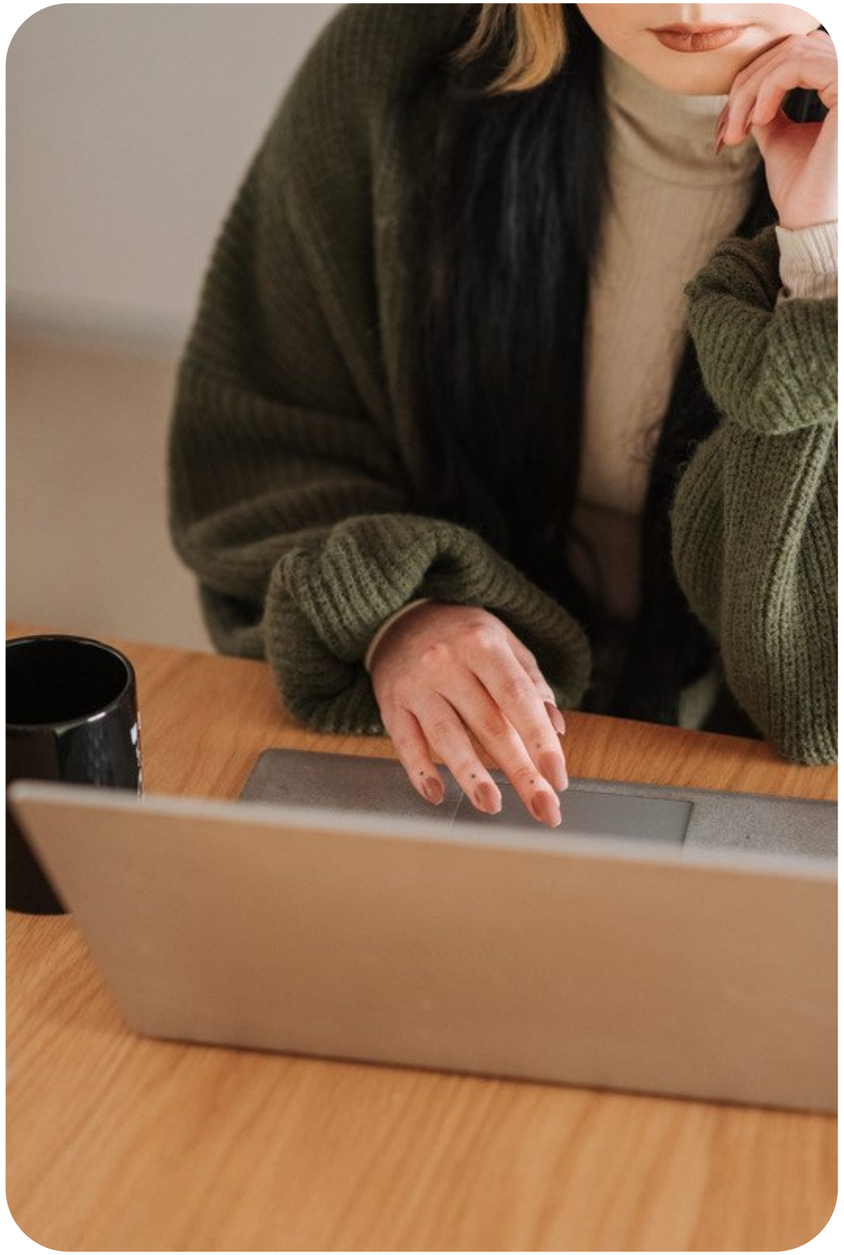
Clase 1

Angel Marroquín

Introduccion a la programacion y computación 1



Fundamentos de programación

Agenda

01

Introducción a Algoritmos

02

Diagramas de Flujo

03

Introducción a la programación

04

Preguntas

Introducción

El curso de introducción a la programación y computación 1, es el acercamiento del estudiante a la programación, en donde él entenderá y empleará mediante diversos usos de disciplinas y metodologías para resolver problemas.
Su principal objetivo es que el estudiante adquiera la habilidad y los conocimientos para poner en practica el uso de técnicas mediante algoritmos.

2

Analisis del problema

ANÁLISIS Y COMPRENSIÓN DE PROBLEMAS

- Identificar el problema
- Identificar la necesidad derivada del problema
- Identificar la necesidad de un sistema de información

Juan es propietario una tienda, Juan ha tenido problemas últimamente, ha percibido una baja de ventas de productos diarios y pocos ingresos. Aunque en sus intentos de verificar que está pasando el ha implementado un sistema de inventario en papel, aun así Juan no halla su solución. El desea llevar el control de su tienda.

- Identificar factores directos e indirectos que intervienen en el problema
- Identificar y analizar los requisitos e implicaciones del proyecto

2

Introducción a algoritmos

¿QUÉ ES UN ALGORITMO?

- Sucesión finita de pasos ordenados que resuelven un problema
- Acción previa a escribir código
- Ejemplos en realidad: hacer un pastel, cargar el celular
- Ejemplos en matemáticas: sumar dos números, calcular el factorial de un número

¿CÓMO SE COMPONE UN ALGORITMO?

- Declaración del algoritmo con nombre
- Input o datos de entrada que el algoritmo necesita
- Proceso
- Output o salida
- Finalización del algoritmo

¿PARA QUÉ SIRVE UN ALGORITMO?

- sirve para resolver paso a paso un problema

2

Introducción a algoritmos

TIPOS DE ALGORITMOS

- **Algoritmos computacionales:** Un algoritmo cuya resolución depende del cálculo
- **Algoritmos no computacionales:** Aquellos que no requieren de los procesos de un computador para resolverse
- **Algoritmos cualitativos:** Se trata de un algoritmo en cuya resolución no intervienen cálculos numéricos
- **Algoritmos cuantitativos:** es un algoritmo que depende de cálculos matemáticos para dar con su resolución

CARACTERISTICAS

- Secuenciales
- Precisos
- Ordenados
- Finitos
- Concretos
- Definidos

2

Introducción a algoritmos

REQUISITOS

Para dar pie en marcha a la creación de un algoritmo es necesario cumplir con ciertos requisitos.

- Entender hacia dónde se dirige la solución
- Estructura del algoritmo
- Datos de Entrada
- Datos de Salida
- Procesos o cálculos relacionados con el problema y solución
- Relaciones entre datos de entrada y salida

Ejemplos de algoritmos

Calcular el área de un triangulo

INICIO

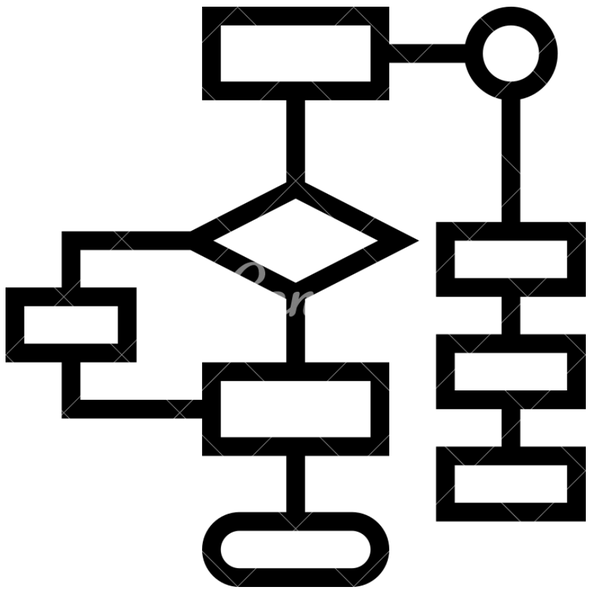
1. Proporcione la base (b) y altura (h)
2. Multiplicar: base por altura la base b por la altura h
3. Dividir entre 2 el resultado anterior y asignarlo al área (A)
4. Mostrar el resultado en pantalla de A
5. FIN

Comprar zapatos de fiesta

INICIO

1. Entrar a la tienda y buscar la sección de zapatos de caballero.
2. Tomar un par de zapatos.
3. ¿Son zapatos de fiesta?
4. SI: (ir al paso 5) – NO: (volver al paso 2)
5. ¿Hay de la talla adecuada?
6. SI: (ir al paso 7) – NO: (volver al paso 2)
7. ¿El precio es pagable?
8. SI: (ir al paso 9) – NO: (volver al paso 2)
9. Comprar el par de zapatos elegido.
10. FIN

Diagrama de flujo



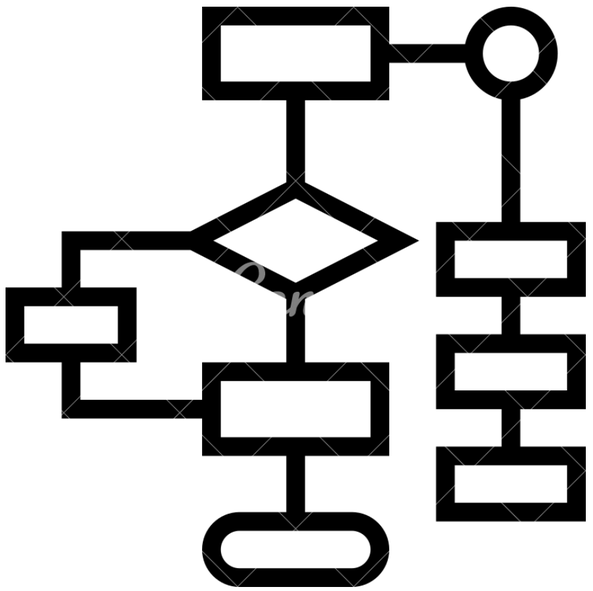
Es una representación grafica o esquemática de un algoritmo, el cual esta compuesto de símbolos y líneas de relación.

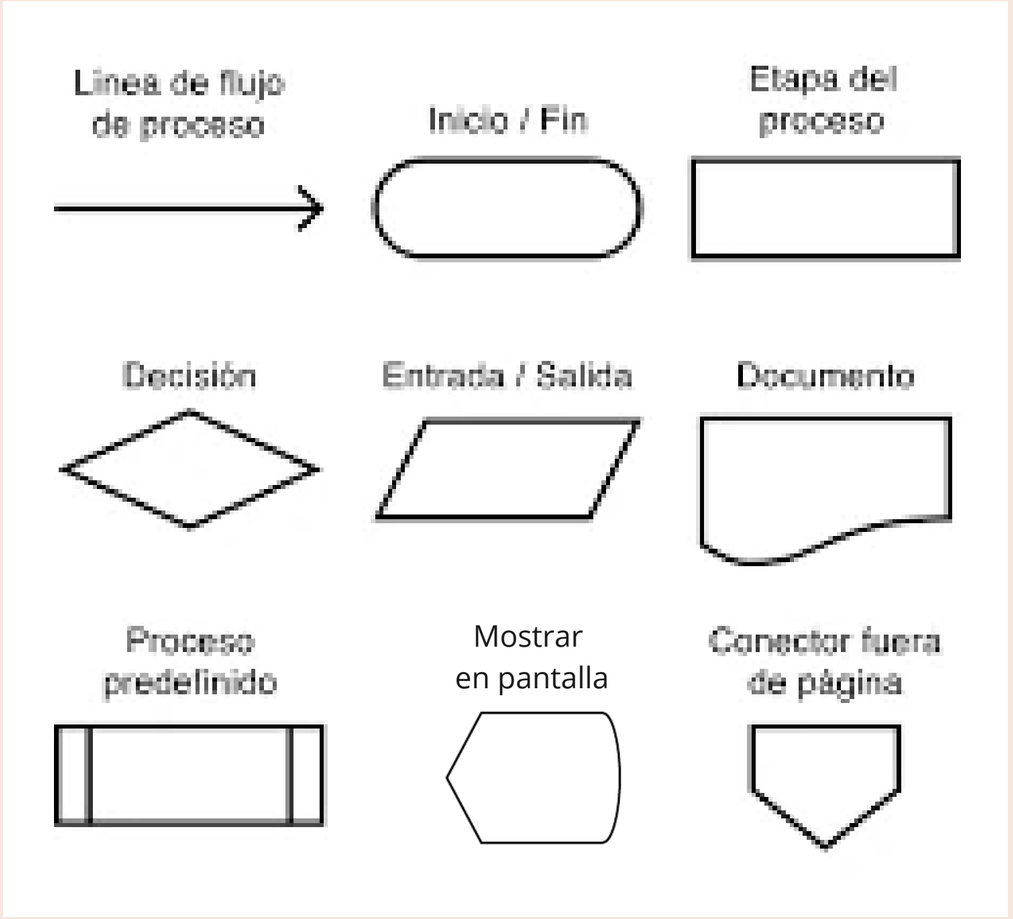
El diagrama de flujo siempre posee un inicio y un final
Las líneas de relación deben ir solo en vertical y horizontal, ya que indican el orden de los pasos
Es independiente del lenguaje de programación

Herramientas que ayudan a realizar este tipo de diagramas:

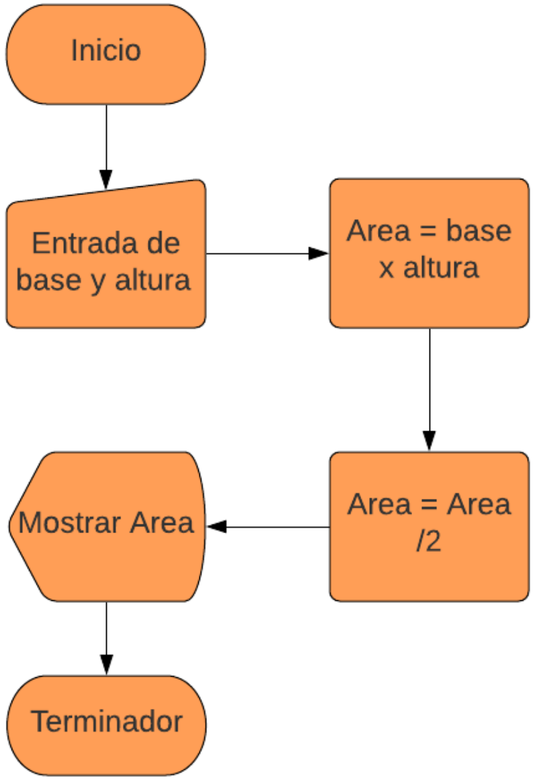
- Microsoft Visio
- Dia
- lucidchart

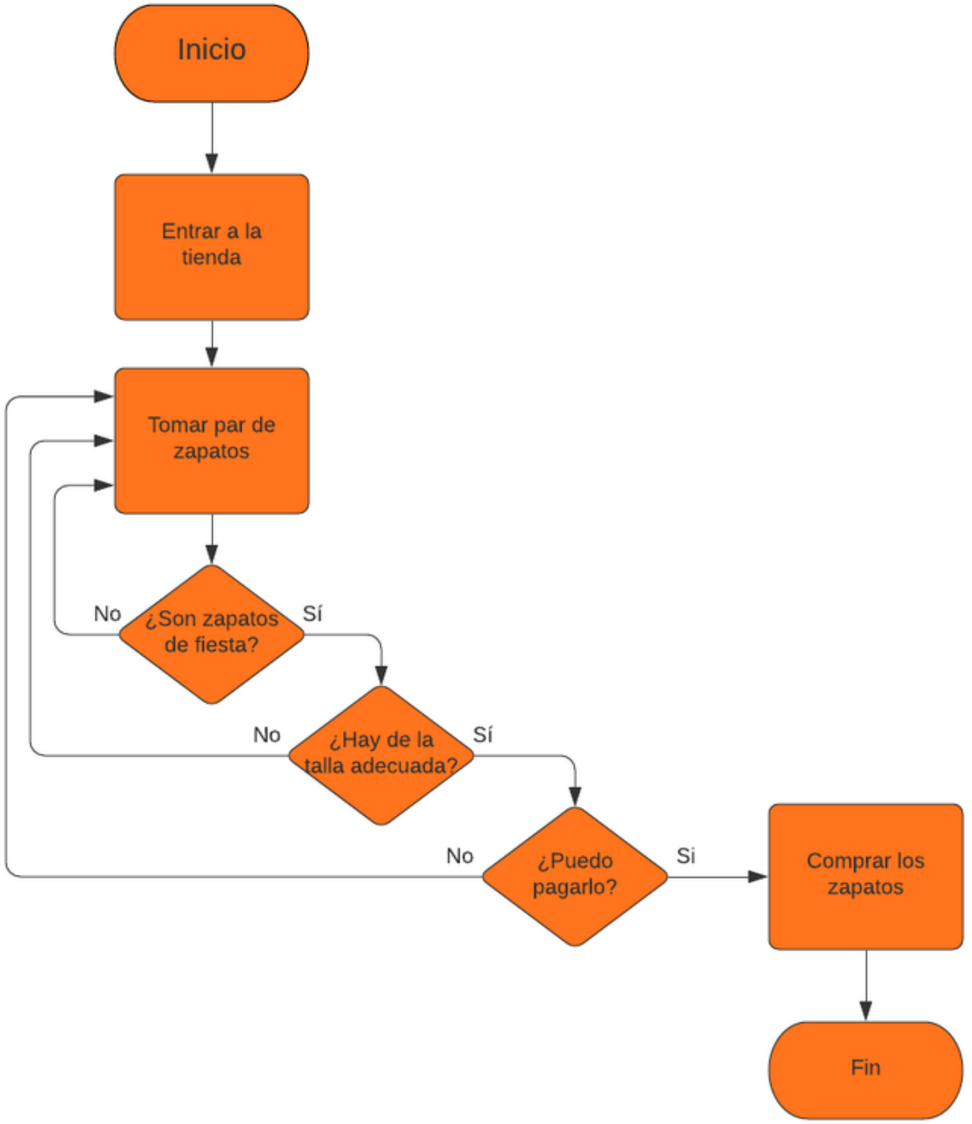
Diagrama de flujo





Ejemplos Diagrama de flujo





Conclusiones o dudas

Conclusiones o dudas

Introducción a la programación

01

Introducción

02

Elementos del lenguaje

03

Expresiones

04

Estructuras de Control

3

Elementos de la programación

IDENTIFICIDORES Y COMENTARIOS

Los identificadores son los nombres con los que se identifican valores en un lenguaje de programación, estos en el lenguaje Java tienen ciertas reglas:

- Debe iniciar con una letra o un guión bajo
- Pueden contener números, letras y guiones
- No deben llamarse igual que ciertas palabras reservadas que serán vistas mas adelante.

Tengo un valor numérico 5
Quiero ponerle de nombre "patito"
ahora diré, "patito" tiene valor 5
ahora quiero sumar 3 unidades a "patito"
ahora diré, "patito" tiene valor 8

Los comentarios en Java inician con doble barra (//) para comentarios de una sola línea y barra asterisco (/*) para iniciar un comentario multilínea terminando con asterisco barra (*/)

```
// éste es un comentario
/* éste es un comentario de varias líneas
   el comentario sigue aquí
*/
```

3

Elementos de la programación

VARIABLES, CONSTANTES Y TIPOS DE DATOS

Variables:
Un valor es una de las cosas básicas que utiliza un programa, como una letra o un número. Esos valores pertenecen a tipos diferentes. Una variable puede visualizarse como un espacio dentro del programa que adquiere un valor que puede cambiar (variar) durante la ejecución del programa.
En java existen tipos de datos que restringen los datos que puede almacenar las variables.

varible numérica patito es igual a cinco
Int patito = 5;

variable cadena de texto miCadena es igual a "Hola Usac"
String miCadena = "Hola Usac";

Int patito = "Hola Mundo"; ERROR. NO SE PUEDE.

3

Elementos de la programación

TIPOS DE DATOS

1. **Datos Primitivos**
    a. Son tipos de datos elementales u originales, estos solo almacenan datos atómicos y son nativos del lenguaje
        i. **int** -> valores numéricos enteros
        ii. **float**, **double** -> valores numéricos con decimales
        iii. **char** -> un único caracter
        iv. **boolean** -> verdadero o falso
2. Datos no primitivos
    a. Estos son datos que son mas complejos poseen sus propios atributos y otras funcionalidades.

varible numérica patito es igual a cinco
Int patito = 5;

variable cadena de texto miCadena es igual a "Hola Usac"
String miCadena = "Hola Usac";
int tamaño = cadena.lenght()

3

Elementos de la programación

OPERADORES, PRIORIDAD DE OPERADORES

Prior.	Operador	Tipo de operador	Operación
1	++	Aritmético	Incremento previo o posterior (unario)
	--	Aritmético	Incremento previo o posterior (unario)
	+, -	Aritmético	Suma unaria, Resta unaria
	~	Integral	Cambio de bits (unario)
	¡	Booleano	Negación (unario)
2	(tipo)	Cualquiera	
3	*, /, %	Aritmético	Multiplicación, división, resto
4	+, -	Aritmético	Suma, resta
	+	Cadena	Concatenación de cadenas
6	<, <=	Aritmético	Menor que, Menor o igual que
	>, >=	Aritmético	Mayor que, Mayor o igual que
	instanceof	Objeto, tipo	Comparación de tipos
7	==	Primitivo	Igual (valores idénticos)
	¡=	Primitivo	Desigual (valores diferentes)
	==	Objeto	Igual (referencia al mismo objeto)
	¡=	Objeto	Desigual (referencia a distintos objetos)

3

Elementos de la programación

OPERADORES, PRIORIDAD DE OPERADORES

8	& &	Integral Booleano	Cambio de bits AND Producto booleano
9	^ ^	Integral Booleano	Cambio de bits XOR Suma exclusiva booleana
10	| |	Integral Booleano	Cambio de bits OR Suma booleana
11	&&	Booleano	AND condicional
12	||	Booleano	OR condicional
13	? :	Booleano, cualquiera, cualquiera	Operador condicional (ternario)
14	= *=, /=, %= +=, -= <<=, >>= >>>= &=, ^=, |=	Variable, cualquiera	Asignación Asignación con operación

3

Elementos de la programación

PALABRAS RESERVADAS

Son palabras propias del lenguaje
Ejemplos:

- Class
- Public
- Private
- Static
- Protected
- int
- string
- void
- etc

4

Estructuras de Control

IF - Si

Si numero1 es igual que numero2 entonces
    mostrar "son iguales"
si no
    mostrar "son diferentes"

```
if( condición ) {
    ...instrucciones
} else if ( condición ) {
    ...instrucciones
} else {
    ...instrucciones
}
```

4

Estructuras de Control

IF - Si

Si numero1 es igual que numero2 entonces
    mostrar "son iguales"
si no
    mostrar "son diferentes"

```
if( condición ) {
    ...instrucciones
} else if ( condición ) {
    ...instrucciones
} else {
    ...instrucciones
}
```

4

Estructuras de Control

Switch - Seleccionar o En caso

variable = 5

seleccionar variable
en caso 2
    mostrar "el valor es dos"
en caso 5
    mostrar "el valor es cinco"
en caso 7
    mostrar "el valor es siete"

```
switch(variable)
case 1:
    ...instrucciones
    break;
case 2:
    ...instrucciones
    break;
case 3:
    ...instrucciones
    break;
default:
    ...instrucciones
}
```

4

Estructuras de Control

For - De Mientras

desde variable = 0 mientras variable sea menor que 7
    mostrar "La variable es " + variable
    variable = variable + 1

```
for (int i = 0; i < 7; i++) {
    ...instrucciones
}
```

4

Estructuras de Control

Do while - hacer mientras
variable = 8
hacer
    mostrar "La variable tiene valor " + variable
    variable = variable / 2
mientras variable sea diferente que 2

```
do {
    ...instrucciones
} while (condición)
```

4

Estructuras de Control

while - mientras
variable = 8
mientras variable sea diferente que 2 hacer
    mostrar "La variable tiene valor " + variable
    variable = variable / 2
mostrar "ya es dos!"

```
while (condición) {
    ...instrucciones
}
```

Conclusiones o dudas