CONDICIONALES

CONDICIONALES

COMPLEJIDAD CICLOMÁTICA

CONDICIONALES

TUTTI-FRUTI

CONDICIONALES

AL DESPERTAR

- Escribe una lista de acciones que realizas al despertar hasta que te vas de casa. ¿Hay algunas condicionales?

CONDICIONALES

ALGORITMO

Una máquina está construida para suelos necesita ser programada para delimitar un terreno de 20 metros de largo y 15 de ancho. Escriba un algoritmo que realice esta tarea.

La máquina acepta las siguientes instrucciones:

- Subir brocha Sube la brocha para que ésta no pinte el suelo.
- Bajar brocha. Baja la brocha para que ésta pinte el suelo.
- Avanzar <número de metros> Mueve la máquina la cantidad de metros indicada.
- Girar <ángulo> Gira la dirección de la máquina.

La máquina se encuentra inicialmente con la brocha subida y se encuentra en una de las esquinas del terreno a marcar.

CONDICIONALES

if … else

```
if (expression)
    statement
```

```
if (expression)
    statement1
else
    statement2
```

```
if (!address) {
    address = "";
    message = "Please specify a mailing address.";
}
```

```
if (error.name == "TypeError")
{
    window.alert("Please connect to internet");
}
else
{
    throw(error);
}
```

CONDICIONALES

if … else if … else

```
if (n === 1) {
    // Execute code block #1
} else if (n === 2) {
    // Execute code block #2
} else if (n === 3) {
    // Execute code block #3
} else {
    // If all else fails, execute block #4
}
```

CONDICIONALES

switch

```
switch(n) {
case 1:                          // Start here if n === 1
    // Execute code block #1.
    break;                       // Stop here
case 2:                          // Start here if n === 2
    // Execute code block #2.
    break;                       // Stop here
case 3:                          // Start here if n === 3
    // Execute code block #3.
    break;                       // Stop here
default:                         // If all else fails...
    // Execute code block #4.
    break;                       // Stop here
}
```

COMPLEJIDAD CICLOMÁTICA

- **Métrica** de software sobre la **complejidad lógica** de un programa
- **Ventajas** de reducirla:
  - Facilita mantenimiento del Código
  - Simplifica refactorización
  - Aumenta calidad de la aplicación
  - Facilita escalabilidad
- **Cómo reducirla**
  - No abusar de anidación
  - Evitar sentencias switch
  - Desarrollar métodos más pequeños

BIBLIOGRAFÍA

1. JavaScript: The Definitive Guide, 7th Edition
2. Eloquent JavaScript, 3th edition, Marijn Haverbeke
3. Clean Code: A Handbook of Agile Software Craftsmanship (Robert C. Martin Series)
4. Modern c++ Programming with Test-Driven Development, Jeff Langr
5. Refactoring: Improve the design of existing Code, Martin Fowler
6. Game programming patterns, Robert Nystrom