



# Introducción al Pensamiento Computacional -

## Semana 3



© Todos los derechos reservados Universidad Rafael Landívar URL.

---

### DESEMPEÑOS ESPERADOS

---

- ≡ Desempeños esperados

---

### DESARROLLO DE CONOCIMIENTOS

---

- ≡ Antecedentes
- ≡ Definición del problema
- ≡ Identificar una solución: algo para tener siempre en mente
- ≡ Descomposición
- ≡ Otras estrategias
- ≡ Patrones y generalización

---

## ☰ Recursos complementarios

### APLICANDO LO APRENDIDO

---

#### ☰ Actividad 1

#### ☰ Actividad 2

#### ☰ Actividad 3

#### ☰ Actividad 4

#### ☰ Actividad 5

### DE LA TEORÍA A LA PRÁCTICA Y REFLEXIÓN

---

#### ☰ Recursos

### CRITERIOS DE EVALUACIÓN

---

#### ☰ Rúbrica de evaluación

#### ☰ Diario de experiencias de laboratorio

### FUENTES DE REFERENCIAS

---

#### ☰ Referencias

### CRÉDITOS

---

#### ☰ Créditos

# Desempeños esperados

---



## El estudiante:

- 1 Explica cómo aplicar un enfoque sistemático para la resolución de problemas.
- 2 Discute sobre cómo crear una definición de un problema.
- 3 Identificar estrategias para divisar soluciones.
- 4 Explicar la descomposición como una estrategia de resolución de problemas.

5

Mostrar los beneficios de la generalización de patrones en problemas y de las técnicas para crearlos.

## Antecedentes

---



Si se tiene un problema, se empieza a analizar y surge una solución. Pero no solo una solución, una que pueda ser llevada a cabo por una computadora. La primera pregunta típica es, ¿por dónde empiezo? Los problemas de la vida real tienden a ser grandes y complejos. Examinar un problema implica identificar los detalles ocultos, complejidades y facetas.

Al enfrentarse a la resolución de problemas complejos, sería útil tener un proceso que nos indique paso a paso cómo resolverlo, así como las instrucciones para ensamblar un mueble. Sin embargo, la solución de problemas es un proceso **creativo**; similar a escribir una obra o

pintar un cuadro. Pero, con las estrategias y prácticas adecuadas, se puede ayudar al proceso creativo.

## Un enfoque sistemático

George Pólya, un matemático húngaro, sugirió las etapas para resolver problemas, inspirados en las Matemáticas y las Ciencias Naturales.

### Los métodos científicos sugieren estos pasos:

- Formular una hipótesis.
- Planear un experimento.
- Ejecutar un experimento.
- Evaluar los resultados.

### Pólya sugiere:

- Comprender el problema.
- Identificar un plan.
- Ejecutar el plan.

- Revisar y extender.

El pensamiento computacional es compatible con estas formas de resolver problemas. En este módulo se profundizará en la comprensión del problema y en la identificación de un plan.

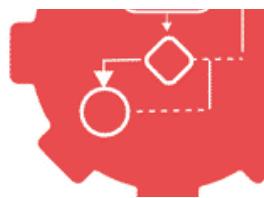
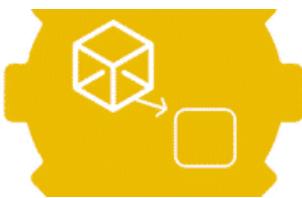
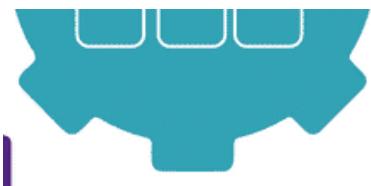
## **¡No hay que asustarse!**

Generalmente los grandes problemas parecen impenetrables, pero también un gran problema se encuentra compuesto por grupos de problemas más pequeños que están interrelacionados. El reto es reducir un problema grande en problemas pequeños que sean más simples y manejables.

## **¡Evite la tentación de tratar de encontrar una solución de inmediato!**

Si se hace, lo más probable es que se solo se resuelva una versión del problema o resolver el problema incorrecto.





## Definición del problema

---



"La parte más difícil de resolver problemas es la caracterización del problema", Michaelson, 2015.

La resolución de problemas implica la transformación de un estado no deseado de una situación (punto inicial) a un estado deseado de la situación (meta). Ambos están íntimamente relacionados. Al analizar el punto inicial se identifica qué es lo que no es deseable y por qué. Esto revela mucho de cómo debería ser la meta.

Las causas por las cuales una situación no es deseable son muchas, algunos ejemplos:

- Un proceso es muy lento, en cuyo caso la meta deberá involucrar mejoras medibles de la rapidez del proceso.
- Se deben tomar decisiones, pero no se tienen datos de respaldo. Esto implica que la meta debe automatizar el proceso de toma de decisiones o filtrar la información para analizarla de la mejor manera.
- Hace falta información sobre cómo se comporta un aspecto particular. Esto sugiere producir un modelo o simulación.

Al tratar de comprender el problema, es **ingenuo** el tratar de responder una respuesta que no se comprende, ante lo cual Pólya recomienda:

- Si alguien le ha trasladado un problema, **reinicie el análisis del problema por sí mismo**. No herede los análisis de alguien más.
- Represente el problema con **figuras y diagramas**. Las personas se desempeñan de mejor manera con representaciones visuales.
- Identificar lo que se sabe y lo que no se sabe desde el inicio. Debe asegurarse de tener la información suficiente para desarrollar una solución. Si no se tiene, se deben explicitar **lo que no se sabe y se debe aprender o investigar**.

No importa el problema con el que se esté tratando, se debe recordar que la meta define lo **que** se necesita y **no cómo** se debe realizar. Pensar en los detalles, como diseños y algoritmos en este punto es muy prematuro. Enfóquese en definir la meta.

Cuando considere cuál es la meta, pregúntese: ¿cómo sabré que el problema ha sido resuelto? Por ejemplo: si su problema es planear una conferencia (alojar espacio para los presentadores, calendarizar recesos y comidas, organizar los auditorios, etc.), entonces, el producto final debe ser un plan en el cual cada evento tenga su horario, ubicación y personal asignado.

Cuando establezca la meta, asegúrese que utilice un vocabulario claro y específico. Por ejemplo, si desea mejorar el tiempo de respuesta de un sistema específico, no defina su meta como "debe ser más rápido", sino indique una unidad de medida precisa.

**Siempre hay que escribir la definición del problema y la meta como si será analizada por alguien más. Esto le obligará a ser más objetivo y tener una vista más considerada.**



## Identificar una solución: algo para tener siempre en mente

---



**Identificar una solución:  
algo para tener siempre en mente**

Una vez se haya finalizado la definición del problema y la meta, es el momento de considerar la estrategia para encontrar una solución. En este módulo se dará prioridad a la **descomposición**, ya que es una estrategia clave para el pensamiento computacional.

Antes de iniciar con este tema, es importante recordar algunos aspectos.



1

CALIDAD



2

COLABORACIÓN



3

ITERACIÓN



1

CALIDAD



COLABORACIÓN



ITERACIÓN

## Calidad

Tome nota que se menciona **una** solución, no **la** solución. Para cualquier problema existen múltiples soluciones, unas buenas, unas terribles y otras en el medio. Se debe enfocar en encontrar la mejor solución. Para el problema completo puede ser que no exista una solución perfecta, pero sí hacer el mejor esfuerzo porque las soluciones a los problemas que lo componen sean lo óptimas posibles.



CALIDAD



COLABORACIÓN



ITERACIÓN

2

## Colaboración

Resolver problemas de forma colaborativa es muy útil. A veces, solo con explicar el trabajo actual en voz alta ayuda a identificar errores y potenciales mejoras. Se debe buscar la visión de otras personas. Una perspectiva fresca puede permitir corregir el camino. En este sentido las sesiones de lluvias de ideas son muy utilizadas, éstas pueden ser grabadas a fin de no perder notas importantes.



CALIDAD



COLABORACIÓN

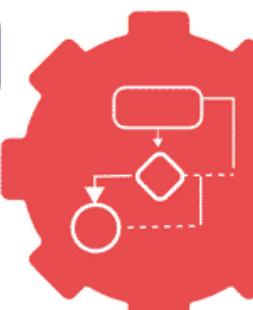
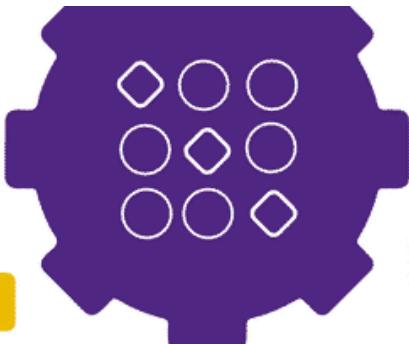
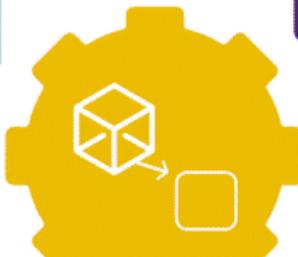
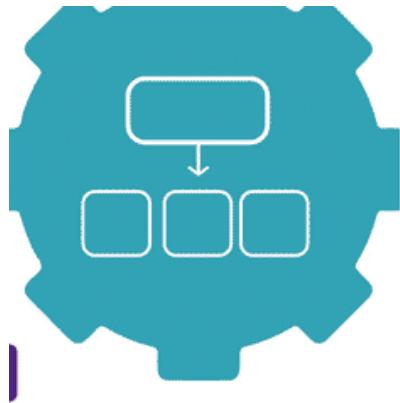


3

ITERACIÓN

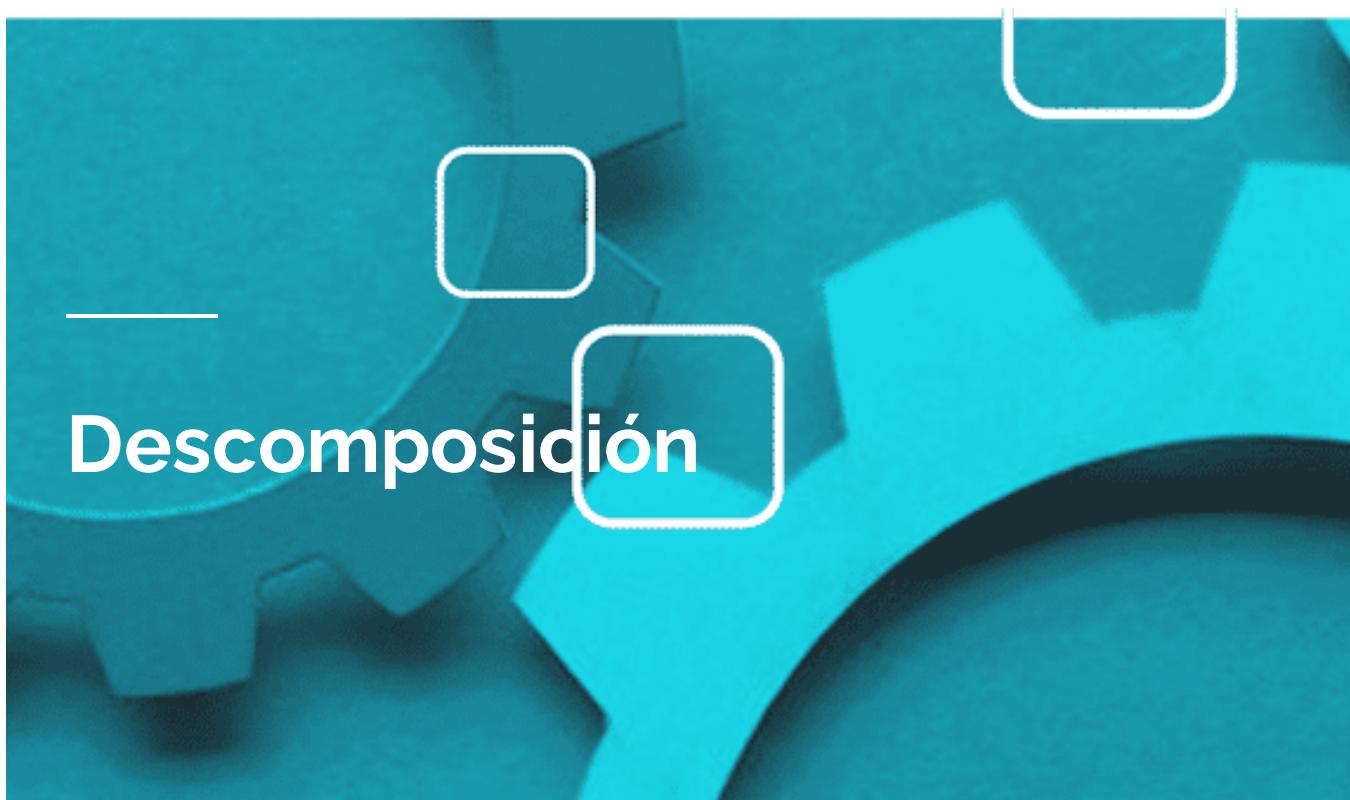
## Iteración

Se debe aceptar el hecho que la primera versión de la solución es raramente la mejor. El enfoque iterativo permite ir afinando la solución. La cantidad de iteraciones dependerá de los objetivos y el tiempo disponible.



# Descomposición

---



Pólya explica diferentes estrategias para resolución de problemas, a estas estrategias se les llama heurísticos. Un **heurístico** es una estrategia específica que conduce a una respuesta o solución lo suficientemente buena, en lugar de una no tan buena. Algunos ejemplos incluyen: prueba y error, diagramación y analogías.

El pensamiento computacional promueve el uso de la **descomposición**, que busca descomponer un problema complejo en partes más sencillas y fáciles de manejar.

## **La descomposición se basa en la estrategia divide y vencerás, referida en otras áreas fuera de la computación, tales como:**

- Campos de batalla, cuando el enemigo es más numeroso. Al comprometer solo una parte de las fuerzas enemigas, se neutraliza al oponente, atacando a un grupo a la vez.
- En la política se utiliza para desintegrar a la oposición en pequeños partidos, que resultan más débiles.
- Cuando se enfrenta a una audiencia grande y diversa, los mercadólogos segmentan a sus consumidores potenciales en diferentes grupos y los atienden de manera diferenciada.

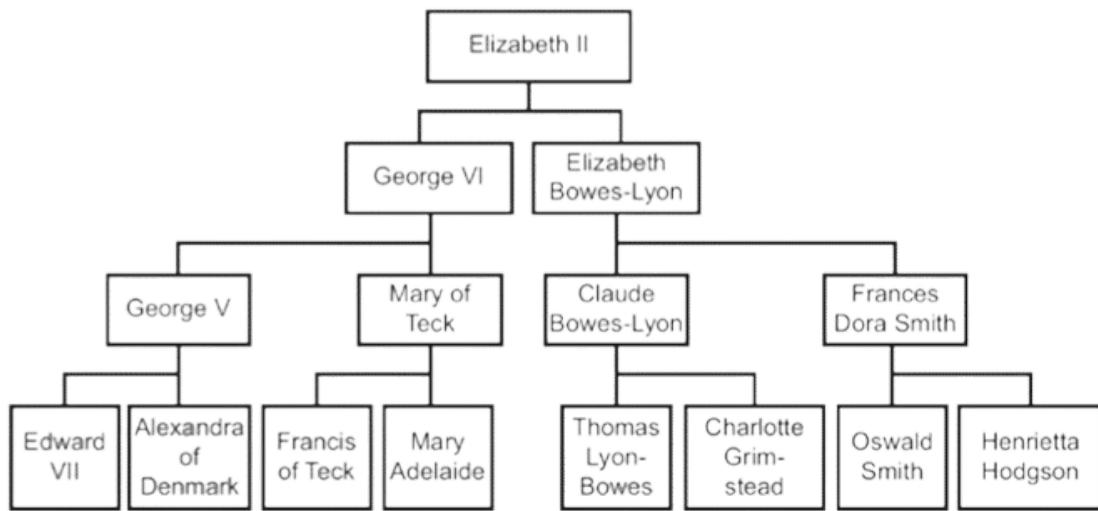
En pensamiento computacional, se utiliza divide y vencerás cuando se enfrenta a un problema grande y complejo. Por ejemplo: un problema puede contener muchas partes interrelacionadas o un proceso puede tener muchos pasos que necesitan mejora. Aplicando la descomposición a un problema implica recoger todas esas partes.

---

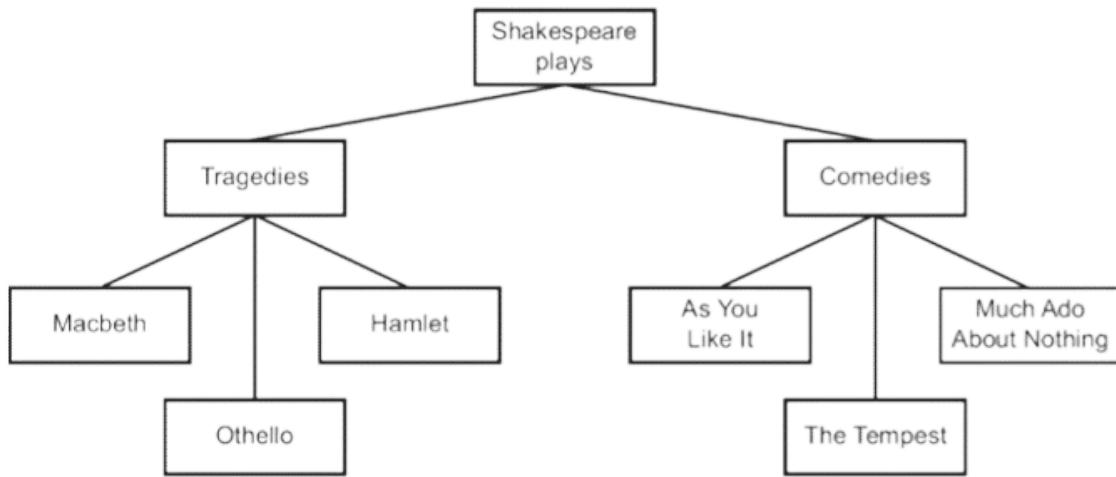
**Recursión:** una técnica para simplificar un problema. Define la solución a un problema grande y complejo en términos de problemas más pequeños, simples y de la misma naturaleza del problema original.

Al aplicar la descomposición, se pretende terminar con subproblemas que pueden ser comprendidos y resueltos individualmente. Esto requiere aplicar un proceso **recursivo**. Visualmente, se le da al problema una estructura de árbol.

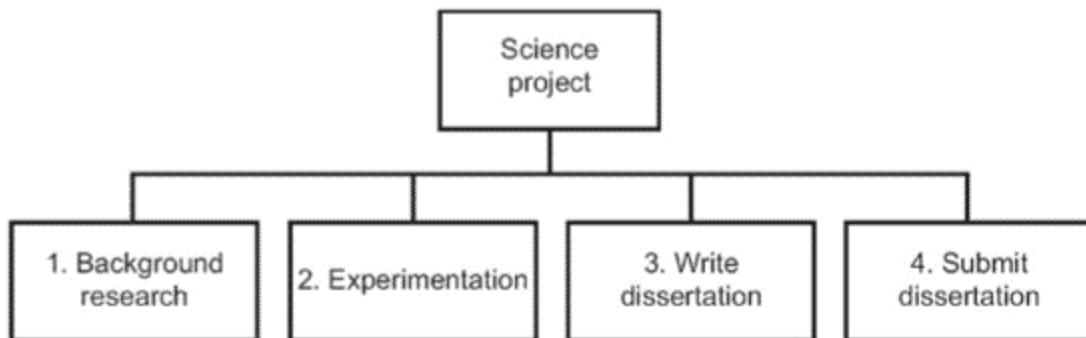
La idea conceptual de un árbol es importante en las ciencias de la computación y va de la mano con la recursión. Un árbol representa un conjunto de entidades organizadas jerárquicamente. Cada entidad padre puede o no tener entidades hijas. La entidad principal se llama raíz. En la siguiente figura se representa el árbol de la familia real británica partiendo de la Reina Isabel como raíz, luego sus padres, los padres de sus padres y el resto de los ancestros. En este árbol, observamos cuatro niveles.



Otro ejemplo de árbol son las obras de Shakespeare por género (tragedias y comedias).

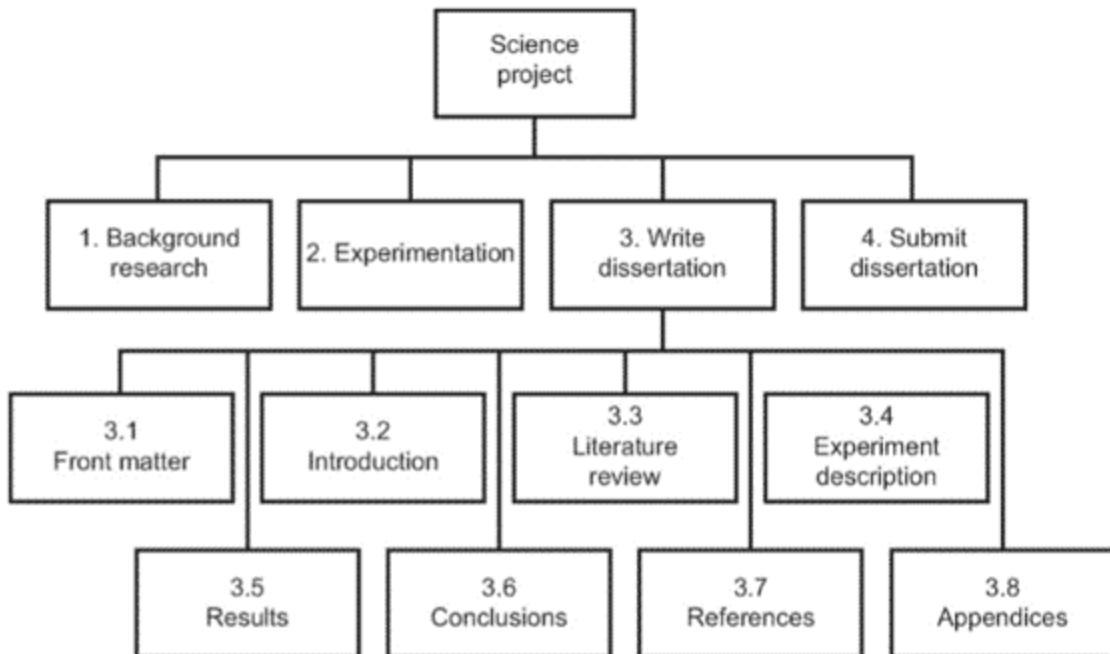


Veamos un ejemplo del ámbito académico. Escribir una disertación científica de tesis es un problema complejo, pero al descomponerlo en partes, se puede prestar atención en cada una para enfrentarlo de manera más efectiva.



Cada una de las tareas puede tener subtareas, aplicando la descomposición de manera

recursiva. En la figura se observa que la tarea de escribir la disertación se ha descompuesto en 8 subareas. En este punto siempre se puede regresar al árbol y encontrar otra tarea que requiera descomposición, de manera que las **hojas** sean problemas que se puedan resolver individualmente.



**Recuerde: en este punto seguimos analizando qué hay que resolver, no cómo resolverlo.**

La descomposición se apoya en la colaboración. Si se descompone un problema adecuadamente, es decir, que los subproblemas pueden ser resueltos de manera

independiente, entonces se puede recurrir a diferentes personas para que atiendan diferentes tareas, permitiendo el **parallelismo**.



## Otras estrategias

---



## Otras estrategias

Si bien le hemos dado protagonismo a la descomposición, no es la única estrategia para resolver problemas. A continuación, una breve descripción de otras estrategias útiles que complementan a la descomposición. No todas pueden aplicarse en cada problema, pero es bueno tenerlas presentes.

**Pensar críticamente**

El propósito del pensamiento crítico es cuestionar las ideas y ver si se sostienen ante el escrutinio. Examinar la información y las decisiones de manera escéptica, justificándose a sí mismo sobre su validez. Cada decisión debe tener una buena justificación que la respalde.



Por ejemplo: si se diseña una portada para una librería, debería cuestionarse: ¿he elegido la estructura correcta? Pero esta pregunta aún está muy ambigua, debería ser más específica: ¿podrán los clientes buscar los libros? ¿La estructura permite a los clientes ubicar un libro específico de manera eficiente?

Además de validar las ideas, se deben exponer las razones y supuestos en el proceso. Es muy importante, porque los supuestos implícitos pueden ser falsos.

Finalmente, se debe asumir el hecho que las soluciones pueden tener errores. Así que siempre pregúntese, ¿qué puede salir mal? En el ejemplo de la librería: ¿qué pasa si los libros no se colocan bien? Si las respuestas varían, será necesario mejorar la solución para que acepte estas nuevas posibilidades.

## Resolver una instancia en concreto

Al resolver problemas generalmente se trata abstracciones o falta de detalles. Si bien la abstracción es muy útil en pensamiento computacional, puede ser retador resolver problemas que no están detallados.

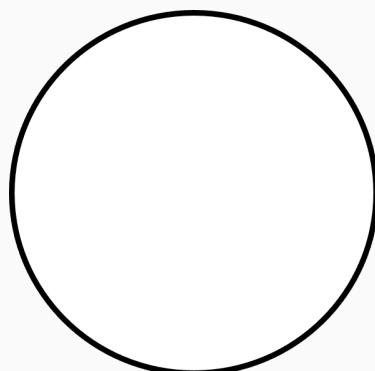
Para Dromey, es mejor lidiar con problemas concretos porque los detalles pueden ser manipulados y comprendidos.



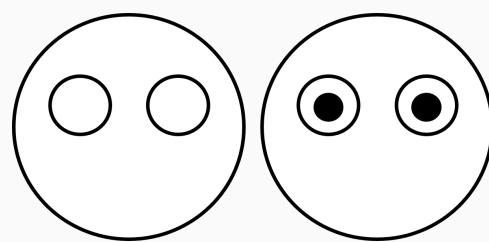
Por ejemplo: considere automatizar el dibujo de formas e imágenes. Suena abstracto, ¿verdad? ¡Cuántos patrones existen! Pero si se enfoca en el problema concreto de una carita feliz, la solución es más comprensible.

Combinando esta estrategia con la descomposición, se puede identificar que podemos dibujar una carita feliz por medio del dibujo de otras formas:

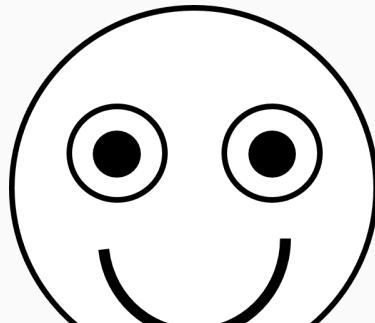
Un círculo que sirva de cara.

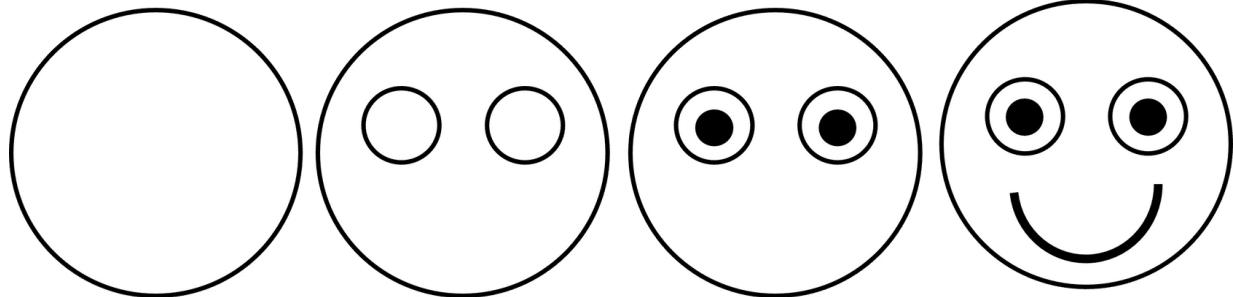


Dos círculos concéntricos  
(el círculo más pequeño es  
sólido) que sirvan de ojos.



Una curva, que haga la  
boca.





Una forma compleja y no familiar, puede ser integrada por otras más simples y familiares.

## Encontrar un problema similar

Esta estrategia invita a examinar un problema desde la solución de otro similar o relacionado, lo cual puede conducir a la solución del propio problema. Esta estrategia debe utilizarse con precaución, ya que permite analizar y descubrir aspectos importantes del propio problema, pero no resolverlo completamente. Requerirá análisis particular, definición de metas y adaptaciones.

## Trabajar en reversa

Esto implica empezar por el estado de la meta y luego ir en reversa etapa por etapa, hasta llegar al estado actual. Esto permitirá visualizar qué hace falta para llegar a la meta.

Esta estrategia es útil cuando la meta está bien definida. Por ejemplo: si el problema es planear la ruta de un lugar para llegar a una hora en particular. La meta está clara y a partir de allí, se puede ir en reversa.

Si salgo de casa en la calle principal y debo llegar a la estación central a las 3:00 p.m. ¿A qué hora debo salir?

3:00 p.m. Llegar a la estación central.

2:55 – 3:00 p.m. Caminar de la estación subterránea 3 al a estación central.

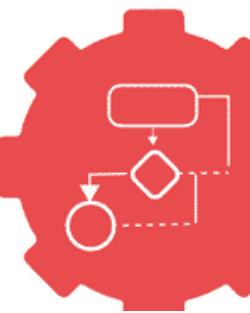
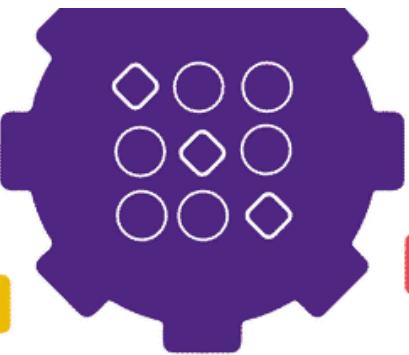
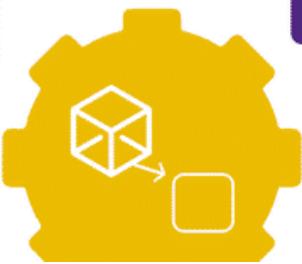
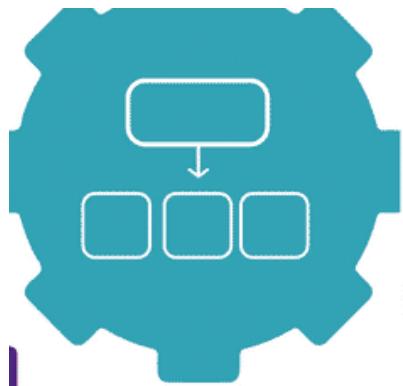
2:45 – 2:55 p.m. Tomar el bus de la ciudad hacia la estación subterránea 3.

2:30 – 2:45 p.m. Tomar el bus en la calle principal hacia la estación de bus de la ciudad.

2:25 – 2:30 p.m. Caminar de la casa en la calle principal a la estación de bus.

2:25 p.m. Hora de salida.





# Patrones y generalización

---



**Patrones y generalización**

El resolver problemas efectivamente involucra más que solo encontrar una solución. Una vez se encuentre la solución, se debe invertir esfuerzo en su mejora y optimización, para ello el reconocimiento y exploración de patrones son claves.

**¿Por qué buscar patrones?**

Cuando se analiza un problema y se empieza a trabajar en la solución, probablemente se notará que algunos elementos se repiten o son muy similares a otros. Si no se identifican patrones en una solución, se recomienda realizar un esfuerzo para buscarlos, porque son el primer paso para una solución más manejable y poderosa.

En el ejemplo de la carita feliz, un algoritmo que dibuje la imagen se vería así:

- 1 Dibujar un círculo de radio 30 en la posición 50,50 con la línea de grosor 2
- 2 Dibujar un círculo de radio 6 en la posición 40,40 con la línea de grosor 1
- 3 Dibujar un círculo de radio 3 en la posición 40,40 relleno de negro
- 4 Dibujar un círculo de radio 6 en la posición 60,40 con la línea de grosor 1
- 5 Dibujar un círculo de radio 3 en la posición 60,40 relleno de negro
- 6 Dibujar una línea de la posición 30,70 a la posición 70,70 con línea de grosor 1

En este ejemplo, los patrones son fáciles de identificar. Cuando los patrones emergen, se posibilita la agrupación de solución con un enfoque común. Es decir, se pueden tomar conceptos separados, pero similares y **generalizarlos** en un mismo concepto.

## Reconocimiento de patrones simples

## **Reconocimiento de patrones simples**

Para reconocer patrones simples se pueden considerar algunas recomendaciones:

## Patrones simples 1

**Retomando el ejemplo del dibujo de la carita feliz, los patrones encontrados son:**

- Buscar sustantivos que se repitan constantemente. Pueden corresponder a objetos de la solución.
- Buscar verbos que aparezcan repetidamente. Pueden ser operaciones que lleve a cabo la solución.
- Buscar descripciones concretas, como adjetivos (rojo, grande, cuadrado) indican propiedades de las cosas que pueden ser reemplazados (color, tamaño, forma). También se pueden identificar números, que pueden ser reemplazados por variables.

## Patrones simples 2

**Retomando el ejemplo del dibujo de la carita feliz, los patrones encontrados son:**

- Sustantivos como círculo, radio, posición, línea, grosor.
- Verbos como dibujar.
- Adjetivos como negro, relleno.
- Números en muchos lugares.

## Ahora trasladamos a lenguaje computacional:

- Sustantivos -> se traducen a **objetos**.
- El resto (color, forma) -> traducen como **propiedades** de los objetos.
- Línea y círculo -> son **instancias** de formas.
- Formas -> es una **generalización** de línea y círculo.
- Los verbos -> son **operaciones** que al invocarse llevan datos o **parámetros**.
- Los datos de adjetivos y números -> son **valores** que pueden tomar las propiedades de los objetos.

## **Entonces tendríamos:**

- Dibujar (círculo): esta operación realiza la forma de círculo.
- Llenar (forma, color): esta operación coloca el interior de una forma de un color específico.
- Forma: figura con un contorno. Los tipos de formas son los siguientes.
  - Círculo: figura con un radio, una posición y un grosor de línea.
  - Línea: figura con dos posiciones y un grosor de línea.

En resumen, se tomó una solución concreta, se buscaron **patrones** y se **simplificaron**. La solución original estaba especializada para dibujar ciertas formas. La **generalización**, obtuvo sus conceptos esenciales y los permite aplicar a esa solución y a otras. Esta solución puede ser mejorada introduciendo nuevos conceptos como el lapicero.

## Patrones complejos

Para identificar patrones más complejos se requiere ampliar el alcance, considerando grupos de instrucciones a la vez. Requiere experiencia, pero vale la pena.

Algunas recomendaciones para identificar patrones complejos son:

- Patrones en secuencias de instrucciones u operaciones pueden ser generalizados en **ciclos**.
- Patrones entre grupos separados de instrucciones pueden ser generalizados en **subrutinas**.
- Patrones dentro de condicionales o ecuaciones pueden ser generalizados en **reglas**.

## Ciclos

---

Los ciclos se utilizan cuando se ejecutan las mismas series de pasos repetidamente. Es un tipo de generalización. Se deben observar las formas específicas y luego derivar a una más general. Un bloque de instrucciones muy repetitivo puede facilitar la identificación.

Por ejemplo, considere este bloque.

```
Dibujar un círculo de radio 6 en la posición 40,40  
Dibujar un círculo de radio 3 en la posición 40,40 relleno de negro  
Dibujar un círculo de radio 6 en la posición 60,40  
Dibujar un círculo de radio 3 en la posición 60,40 relleno de negro
```

Vemos un patrón. Al menos dos pasos se repiten igual que los primeros dos. Se podría colocar en un ciclo de esta manera.

Establecer coordenadas = (40,40), (60,40)

Para cada coordinada x,y hacer lo siguiente:

```
Dibujar un círculo de radio 6 en la posición x,y  
Dibujar un círculo de radio 3 en la posición x,y relleno de negro
```

Recordemos del módulo anterior que un ciclo es controlado por una variable, que cambia de iteración a iteración. Esta generalización hace la solución más flexible. Si se quieren dibujar ojos en otros lugares, simplemente se envían las coordenadas. ¿Cómo quedaría el dibujo de un cíclope?

## Subrutinas

---

Los siguientes patrones se encuentran entre bloques separados de instrucciones. En el ejemplo de la carita feliz, imagine dibujar diferentes caras en diferentes lugares; pero pudiendo alterar el radio. En este caso se puede aplicar una generalización, de la siguiente manera. Recordando las instrucciones de dibujo de ojos.

```
Dibujar un círculo de radio 6 en la posición 40,40  
Dibujar un círculo de radio 3 en la posición 40,40 relleno de negro
```

Al generalizarlo, se vería así.

```
Dibujar un círculo de radio r1 en la posición x,y  
Dibujar un círculo de radio r2 en la posición x,y relleno de negro
```

Se pueden declarar estos pasos en una **subrutina** que pueda ser invocada desde diferentes lugares y esté **empaquetada** en una unidad.

Subrutina Dibujar Ojo (r1, r2, x, y)

Dibujar un círculo de radio  $r_1$  en la posición  $x,y$   
Dibujar un círculo de radio  $r_2$  en la posición  $x,y$  relleno de negro

Recordemos que indentamos o tabulamos las instrucciones dentro de la subrutina para mayor claridad y legibilidad.

La invocación de la subrutina sería de la siguiente manera:

Llamar a "Dibujar Ojo" con los parámetros  $r_1 = 6, r_2 = 3, x = 40, y = 40$   
Llamar a "Dibujar Ojo" con los parámetros  $r_1 = 6, r_2 = 3, x = 60, y = 40$   
Llamar a "Dibujar Ojo" con los parámetros  $r_1 = 4, r_2 = 3, x = 40, y = 40$   
Llamar a "Dibujar Ojo" con los parámetros  $r_1 = 4, r_2 = 3, x = 60, y = 40$

## Reglas

Hasta aquí nuestro ejemplo de generalización está muy claro, pero ¿qué pasaría si el círculo del iris es más grande que el de la esclerótica? Es decir, que el círculo interior, sea más grande que el del exterior. Para ello, se pueden definir unas reglas explícitas, de la siguiente manera.

Subrutina Dibujar Ojo ( $r, x, y$ )

Dibujar un círculo de radio  $r$  en la posición  $x,y$   
Dibujar un círculo de radio  $1/2r$  en la posición  $x,y$  relleno de negro

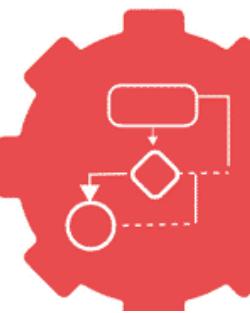
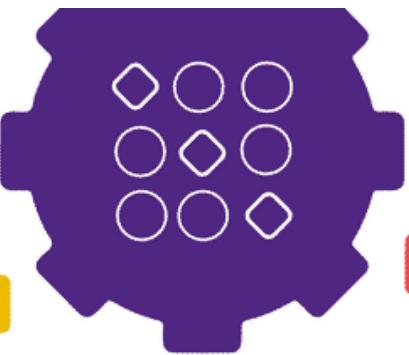
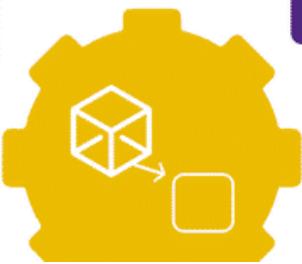
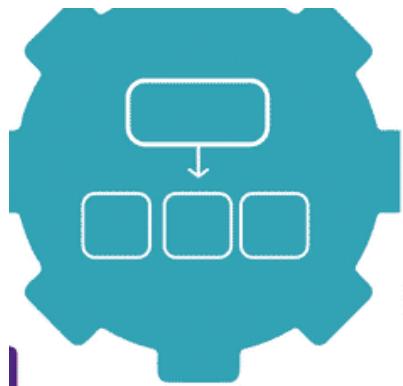
Las ecuaciones y condicionales son buenos identificadores de las reglas. Se utilizan palabras como "cuando", "al menos", así como patrones como "if-then (si-entonces)". Por ejemplo:

Si un círculo no está lleno con un color, entonces que tome el color del fondo.

Esta regla puede ser aplicada a cualquier círculo que se dibuje.

La descomposición y la generalización están relacionadas. La descomposición implica tomar un problema en pequeñas partes y la generalización implica combinar esas pequeñas partes. Estas acciones no son contrarias, ya que al generalizar no se está poniendo el problema como estaba al inicio.

La clave de la generalización es mirar a las partes desagregadas y encontrar formas de que la solución sea más simple de manejar y sea aplicable a problemas similares.



## Recursos complementarios

---



Recursos complementarios

---

# Descomposición

 YOUTUBE



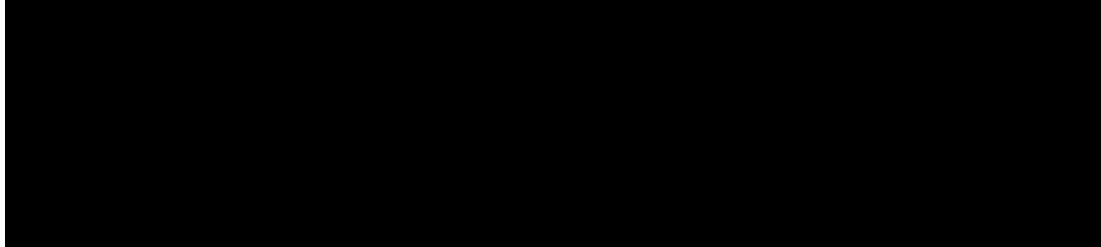
## Introduction to Decomposition

<http://www.education.rec.ri.cmu.edu>

**VER EN YOUTUBE >**

 YOUTUBE

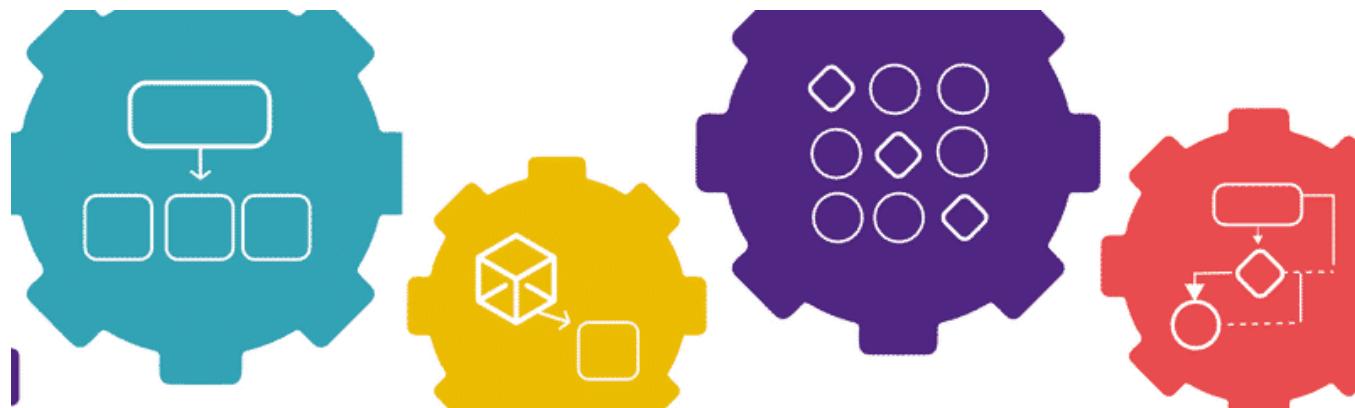




## OCR GCSE 2.1 Decomposition

OCR Specification ReferenceSection 2.1 Why do we disable comments? We want to ensure these videos are always appropriate to use in the classroom. However, we ...

**VER EN YOUTUBE >**



## Actividad 1

---



Actividad 1



## Instrucciones

Indicar si los siguientes enunciados son verdaderos o falsos.

La meta define cómo el problema debe ser resuelto.

Verdadero



Falso

SUBMIT

Es inadmisible empezar a elaborar una solución si no se ha definido la meta.

---



Verdadero



Falso

SUBMIT

Para cada problema complejo solo hay una solución.

Verdadero

Falso

SUBMIT

La descomposición garantiza una solución óptima.

Verdadero

Falso

SUBMIT

Una estructura de árbol es jerárquica.

Verdadero

Falso

SUBMIT

Todos los nodos (entidades) de un árbol tienen uno o más nodos hijos.

---

Verdadero

Falso

SUBMIT

Los patrones entre grupos separados de instrucciones pueden ser generalizados en subrutinas.

---

Verdadero

Falso

**SUBMIT**

## Actividad 2

---



Actividad 2



## Instrucciones

Ud. está planeando una fiesta de cumpleaños para un niño y sus amigos. Separe la preparación en una estructura de árbol, considerando:

1

Se necesita enviar invitaciones a los padres de los otros niños.

2

La comida debe incluir sándwiches (jamón, pollo y queso) y un pastel hecho en casa.

3

Se necesitan comprar todos los ingredientes frescos del día.

- 4 Se requiere contratar cubiertos, manteles y juegos.
- 5 Se debe reservar el lugar para la fiesta de cumpleaños.
- 6 A todos los invitados se les debe dar un obsequio cuando se retiren de la fiesta.

Diríjase a la plataforma para subir su actividad en el espacio correspondiente.

## Actividad 3

---



Actividad 3



Actualice el dibujo de la carita feliz realizado en el módulo, calculando la posición de los ojos y la boca según las medidas de un ser humano promedio.

Diríjase a la plataforma para subir su actividad en el espacio correspondiente.

## Actividad 4

---



Actividad 4



## Instrucciones

Mejore el dibujo de la carita feliz para que:

- Todas las formas tengan color (color de piel, labios, ojos).
- Agregar una cicatriz en la mejilla.
- Agregar una nariz redonda y roja que cubra parcialmente los ojos.

Diríjase a la plataforma para realizar la actividad en el espacio correspondiente.

## Actividad 5

---



Actividad 5



## Instrucciones

Agrupe los animales siguientes en una estructura de árbol, de acuerdo con sus características.  
Tome como ejemplo el árbol de las obras de Shakespeare.

- Murciélagos.
- Cocodrilo.
- Zorro.

- Humanos.
- Pulpo.
- Avestruz.
- Pingüino.
- Tiburón.
- Serpiente.
- Cisne.
- Tortuga.

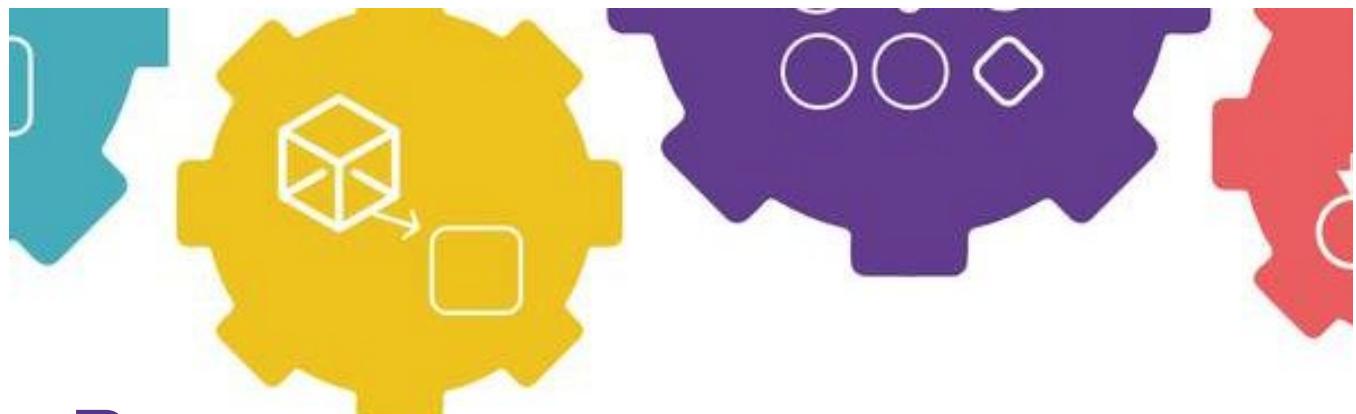
## Agrúpelos en tres estructuras según:

- Número de patas.
- Si pueden volar o no.
- Su clasificación (mamífero, peces, etc.).

Diríjase a la plataforma para subir la actividad en el espacio correspondiente.

## Recursos

---



## Recursos

### Instrucciones:

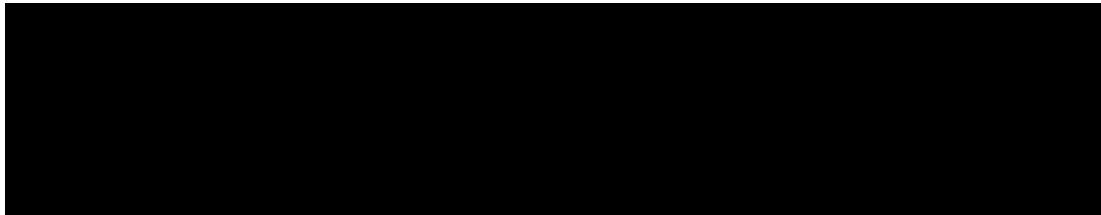
Vea los siguientes videos.



Think like a corder

 YOUTUBE





## **The World Machine | Think Like A Coder, Ep 10**

The adventure ends! Episode 10: Can Ethic make it to the World Machine and stop Hedge from covering the world in a giant maze? --This is episode 10 of our an...

**[VER EN YOUTUBE >](#)**

**La ciencia de la computación es para todos**

 **YOUTUBE**



**Computer science is for everyone | Hadi Partovi | TEDxRainier**

This talk was given at a local TEDx event, produced independently of the TED Conferences. This persuasive talk shows how essential and easy it is to gain a b...

**VER EN YOUTUBE >**

Analizar este video e inspirar a la importancia de pensar computacionalmente.

## Rúbrica de evaluación

---



Rúbrica



Descargue la siguiente rúbrica de evaluación.



**Rúbrica de Evaluación.pdf**

142.3 KB



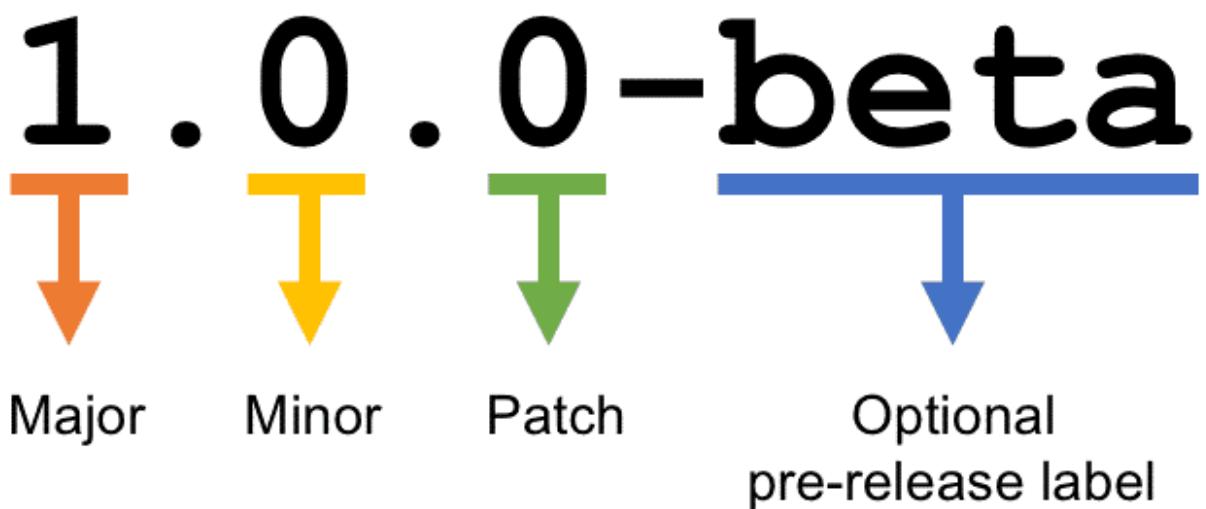
## Diario de experiencias de laboratorio

---



# Diario de experiencias del laboratorio

Cada práctica de laboratorio deberá publicarse en una carpeta en el diario de experiencias de laboratorio, bajo la versión 1.0.0.



# Referencias

---

## Referencias

Bordigón, F. e Iglesias, A. (2020). Introducción al pensamiento computacional. Universidad

Pedagógica Nacional, Educar Sociedad del Estado y el Ministerio de Educación Argentina.

Beecher, K. (2017). Computational Thinking. A beginner's guide to problem-solving and programming.

ISTE (2018). Computational Thinking. Meets Students Learning. International Society for Technology in Education.

MIT (2022). Computational Thinking, a live online Julia/Pluto textbook. Julia: A Fresh Approach to Computing. computationalthinking.mit.edu. Massachusetts Institute of Technology.

Pourbahrami & Tritsy (2018). Computational Thinking: How Computer Science Is Revolutionizing Science and Engineering. ENGenious (15) 8-11.  
<https://resolver.caltech.edu/CaltechCampusPubs:20181025-110029157>.

## Créditos

---

