De la clase anterior …

Desarrollamos el algoritmo de línea de barrido, cuya estructura general se basa en las siguientes funciones

01

def ENCUENTRAINTERSECCIONES(S):

Inicializa una cola de eventos Q vacía. Inserta los puntos finales de los segmentos en Q con el orden definido (mayor y primero). I
Inicializa una estructura de estado T vacía.
while Q no es vacía:
    Extrae p de Q, siguiente evento.
MANEJAREVENTO(p)

02

```
def MANEJAREVENTO(p):
    Si p es un inicio de segmento:
        Introducir el segmento correspondiente en T
        INTERSECCIONSEGMENTO(p,r)
        INTERSECCIONSEGMENTO(p,s) donde r y s son los nuevos vecinos de p en T
    Si p es un fin de segmento:
        Extraer r, s, los vecinos de p en  T
        eliminar el segmento correspondiente en T
        INTERSECCIONSEGMENTO(r,s)
    Si p es una interseccion:
        Intercambiar T el orden de r, s los segmentos que se intersectan en p
        Para q y t los nuevos vecinos de s y r, respectivamente
        INTERSECCIONSEGMENTO(q,s)
        INTERSECCIONSEGMENTO(t,r)
        Reportar interseccion
```

03

```
def INTERSECCIONSEGMENTO(r,s):
        if r y s no se encuentran en la lista L de intersecciones calculadas and if r y s se intersectan:
            Crear un evento p, la intersección de r,s, introducir p en Q
```

Estructuras necesarias

Segmento	Inicio, Fin, id identificador del evento
Evento	Coordenadas, tipo de evento (inicio, fin,..), ids de segmento(s),
Q	Cola de eventos, debe mantener en todo el orden en que se insertan los eventos
T	El estado, que representa donde se encuentra la línea en cada punto del barrido, mantiene los segmentos en orden (x ascendente)
L	Lista de los pares de id de segmentos cuya intersección ya se verificó

Complejidad (en tiempo)

01

def ENCUENTRAINTERSECCIONES(S):

Inicializa una cola de eventos Q vacía. Inserta los puntos finales de los segmentos en Q con el orden definido (mayor y primero). I

Inicializa una estructura de estado T vacía.

O(nlog n)

while Q no es vacía:Extrae p de Q, siguiente evento.

MANEJAREVENTO(p)

O(2n + n_i)

02

```
def MANEJAREVENTO(p):
    Si p es un inicio de segmento:        O(log n)
        Introducir el segmento correspondiente en T
        INTERSECCIONSEGMENTO(p,r)
        INTERSECCIONSEGMENTO(p,s) donde r y s son los nuevos vecinos de p en T
    Si p es un fin de segmento:
        Extraer r, s, los vecinos de p en  T
        eliminar el segmento correspondiente en T    O(log n)
        INTERSECCIONSEGMENTO(r,s)
    Si p es una interseccion:
        Intercambiar T el orden de r, s los segmentos que se intersectan en p
        Para q y t los nuevos vecinos de s y r, respectivamente
        INTERSECCIONSEGMENTO(q,s)
        INTERSECCIONSEGMENTO(t,r)
        Reportar interseccion    O(constante)
```

03

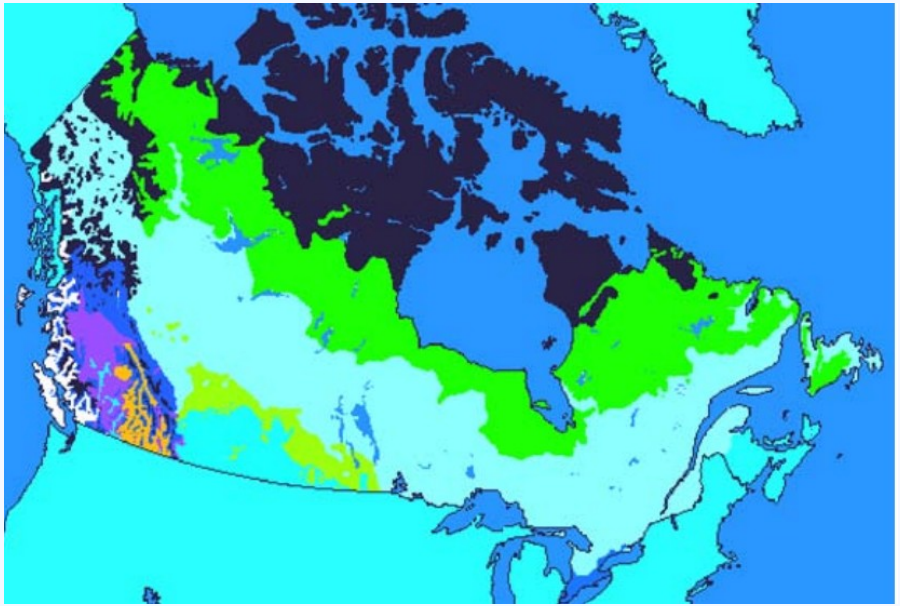
```
def INTERSECCIONSEGMENTO(r,s):
        if r y s no se encuentran en la lista L de intersecciones calculadas and if r y s se intersectan:
            Crear un evento p, la interseccion de r,s, introducir p en Q
```

O(log n)

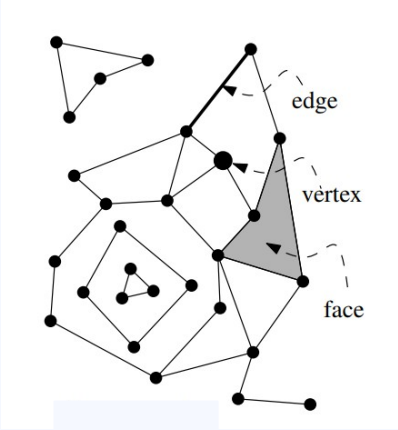
La complejidad es O(nlog n + mlogn), con m el numero de intersecciones encontradas.

Aristas doblemente conexas (doubly connected edges)

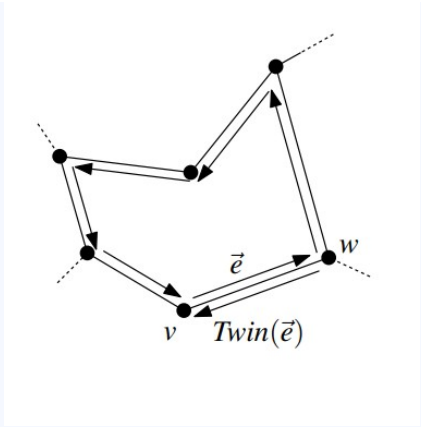
Las subdivisiones del plano, que dividen una región en áreas etiquetadas, son fundamentales en la representación de información geoespacial, como mapas temáticos.



Consideramos mapas como subdivisiones planares inducidas por el mapa.. Buscamos una representación que permita operaciones locales, como recorrer la frontera de una cara o acceder a una cara vecina mediante un borde común. La representación que discutiremos se llama "doubly-connected edge".



Un edge delimita dos faces (por lo general), le tenemos que dar direccion. Medio-edge delimita un solo face. Twins son dos half-edges adyacentes. Definimos una orientación antihoraria para definir el next de cada half-edge. La face correspondiente a un half-edge siempre se encuentra a su izquierda

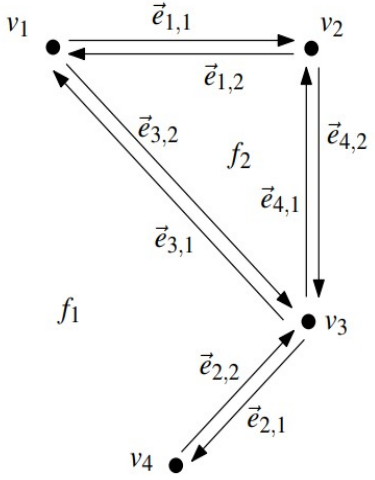


En resumen esta estructura, doubly connected edge list, contiene tres tipos de registros: half-edges, vectices , faces.
Para cada vertice tenemos Coordenadas(v)tambien EdgeIncidencte(v) a un half-edge que tiene v como su origen.

Para un face f ComponenteExterior(f) es un half-edge en su frontera exterior. Para una cara no acotada esto es null. ComponenteInterior(f), contiene por cada hueco un half-edge en la frontera de este hueco.

Para un half-edge e tenemos Origen(e) que apunta a un vertice, Twin(e) a su half-edge opuesto, FaceIncidente(e) al face adyacente, tambien tenemos Next(a) y Prev(e) a el siguiente y anterior half-edge.

Por ejemplo...



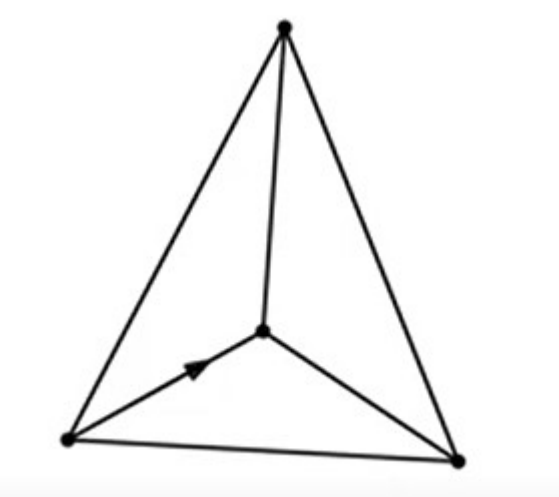
Vertex	Coordinates	IncidentEdge
v_1	$(0,4)$	$\vec{e}_{1,1}$
v_2	$(2,4)$	$\vec{e}_{4,2}$
v_3	$(2,2)$	$\vec{e}_{2,1}$
v_4	$(1,1)$	$\vec{e}_{2,2}$

Face	OuterComponent	InnerComponents
f_1	**nil**	$\vec{e}_{1,1}$
f_2	$\vec{e}_{4,1}$	**nil**

Half-edge	Origin	Twin	IncidentFace	Next	Prev
$\vec{e}_{1,1}$	v_1	$\vec{e}_{1,2}$	f_1	$\vec{e}_{4,2}$	$\vec{e}_{3,1}$
$\vec{e}_{1,2}$	v_2	$\vec{e}_{1,1}$	f_2	$\vec{e}_{3,2}$	$\vec{e}_{4,1}$
$\vec{e}_{2,1}$	v_3	$\vec{e}_{2,2}$	f_1	$\vec{e}_{2,2}$	$\vec{e}_{4,2}$
$\vec{e}_{2,2}$	v_4	$\vec{e}_{2,1}$	f_1	$\vec{e}_{3,1}$	$\vec{e}_{2,1}$
$\vec{e}_{3,1}$	v_3	$\vec{e}_{3,2}$	f_1	$\vec{e}_{1,1}$	$\vec{e}_{2,2}$
$\vec{e}_{3,2}$	v_1	$\vec{e}_{3,1}$	f_2	$\vec{e}_{4,1}$	$\vec{e}_{1,2}$
$\vec{e}_{4,1}$	v_3	$\vec{e}_{4,2}$	f_2	$\vec{e}_{1,2}$	$\vec{e}_{3,2}$
$\vec{e}_{4,2}$	v_2	$\vec{e}_{4,1}$	f_1	$\vec{e}_{2,1}$	$\vec{e}_{1,1}$

Ejemplo, necesidad double-connected edge list

Queremos dar una orientación consistente a los edges de la siguiente figura



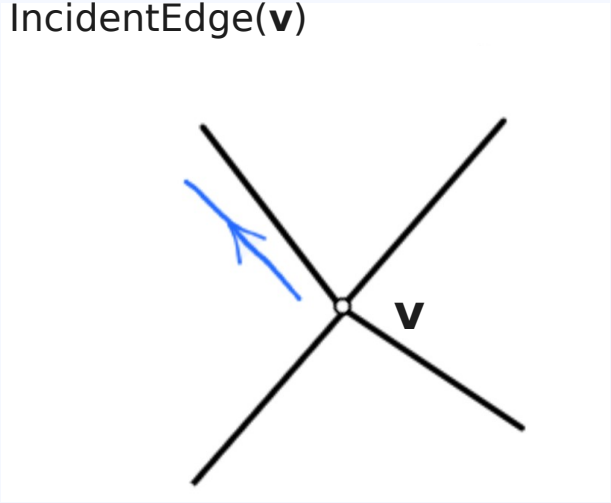
Solucion, half edges

Para esta estructura, doubly connected-edge necesitamos guardar lo siguiente:

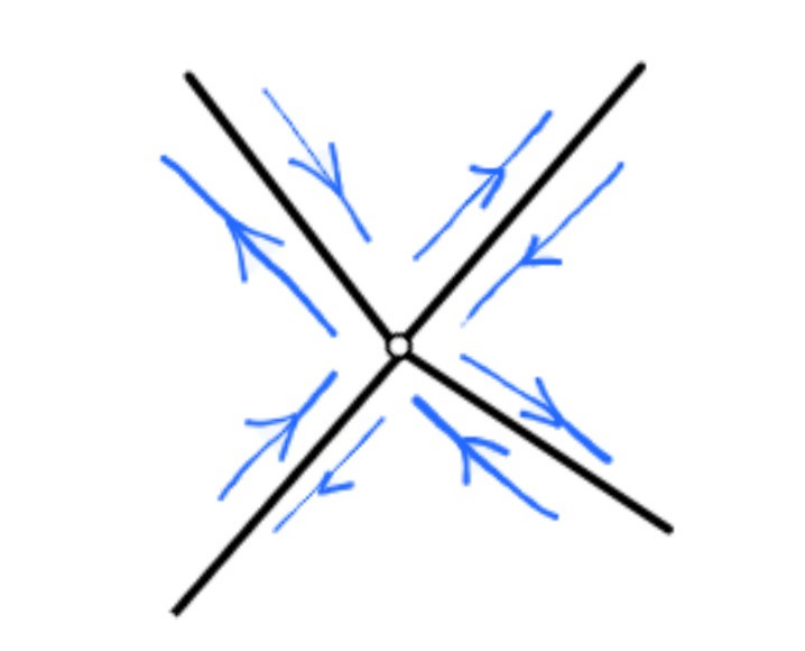
Vertice, v	Coordenadas(**v**), IncidentEdge(**v**) un half-edge con **v** como su origen
Half-edge, e	Origin(**e**) [vertice], Twin(**e**) [half-edge], Next(**e**) [half-edge], Prev(**e**) [half-edge], IncidentFace(**e**) [face] face que se encuentra hacia la izquierda de **e**
Face, f	OuterComponent(**f**)[half-edge] un half-edge con el que se recorra su frontera exterior, OuterComponent(**f**)[half-edge] half-edge sobre su frontera interior

Es esta informacion es suficiente?
Supongamos en el siguente case, para v, tener todos los vertices incidentes

IncidentEdge(**v**)

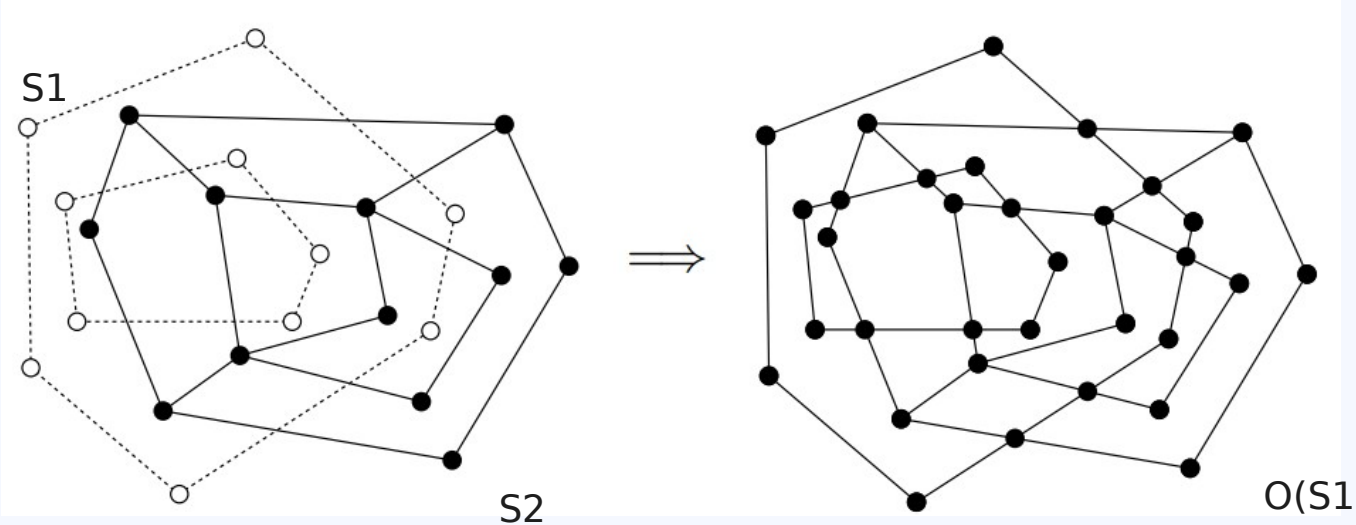


Usando Next(Twin(IncidentEdge(**v**)))

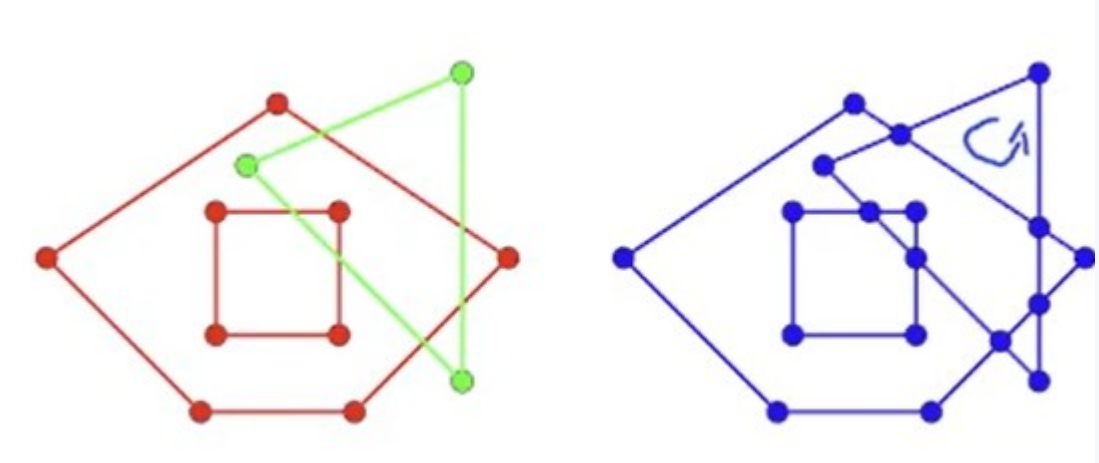


Superposición de 2 subdivisiones

Con esta estructura de dato podemos calcular la superposicion de 2 subdivisiones. Se define la superposicion de subdivisiones S1 y S2 como subdivision O(S1,S2) tal que encontramos un face f en O(S1,S2) si solo son faces f1 en S1 y f2 en S2 tal que f es un conjunto maximal conexo en f1 ∩ f2.



Para la estructura resultante debemos orientar los half-edges de manera que las nuevas faces mantienen la orientación usual, por ejemplo



Queremos que cada face nueva de O(S1, S2) tenga la información correpondiente a las faces en S1 y S2.

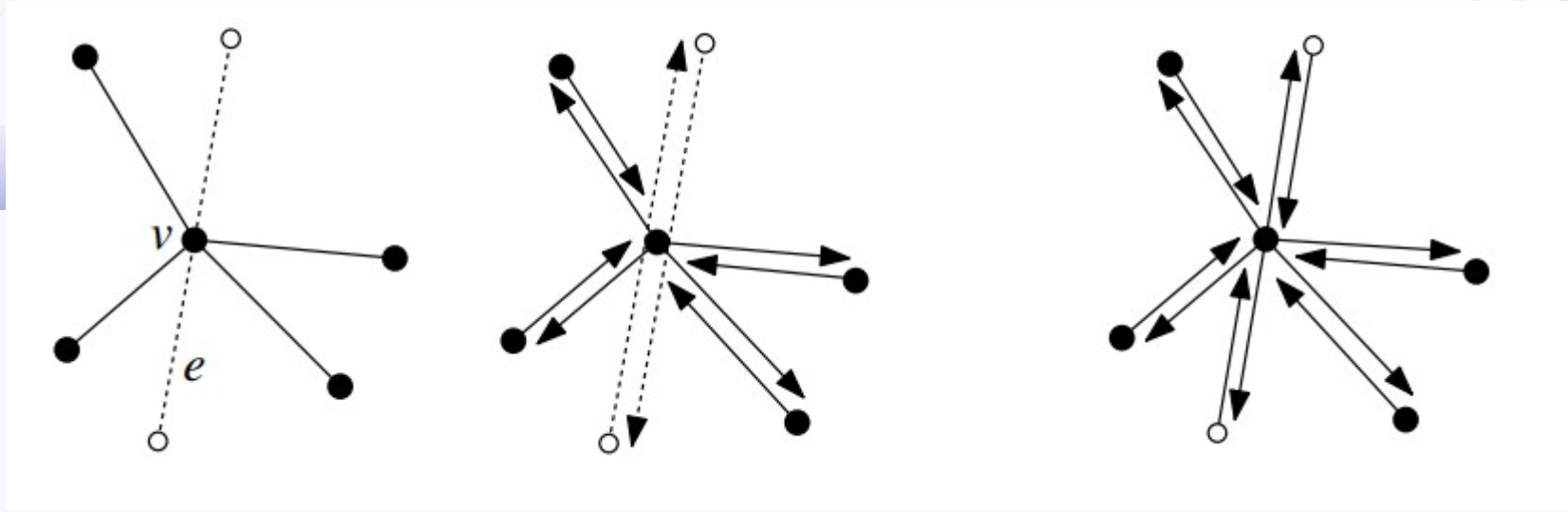
Gran parte de la lista de doubly connected-edges en S1 y S2 se puede reusar para la lista de doubly connected-edges en S, se tiene que cambiar los half-edges de S1 que son cortados por half-edges de S2, y viceversa. Podemos mantener registro de half-edges que no han sido cortados por edges del otro mapa. Ya que las correspondientes caras estarán siempre a la izquierda de un half-edge y esto no cambia, aunque el face incidente puede cambiar.

Para resolver el problema de la superposición primero copiamos la lista de doubly connected-edges de S1 y la de S2 para formar una nueva lista, su union no es una lista de doubly connected-edges valida, esta es la tarea del algoritmo, encontrar las intersecciones y enlazando de manera apropiada la información en cada una de las listas de doubly connected edges.

Para corregir la lista de doubly connected edges de S usamos el algoritmo de linea de barrido, sobre la lista de segmentos generados por los edges de la union de las subdivisiones. Usamos la cola de eventos, Q, el estado de la linea, T, que mantiene en orden x-ascendente los segmentos localizados sobre la línea y adicionalmente llevamos D, la lista doubly connected edges. Inicialmente, D contiene una copia de las respectivas listas de S1 y S2 y a medida que progrese el algoritmo vamos corrigiendo. La informacion de las faces se corregirá al final, y el estado, T, mantendrá los respectivos registros descritos de los segmentos dentro de la línea de barrido para corregirlos en el evento de encontrar intersección.

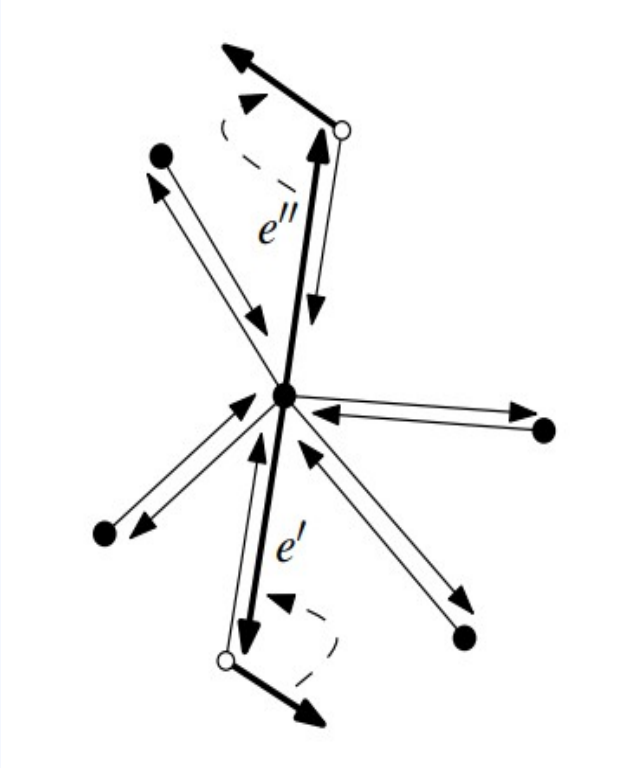
Usamos T, Q las estructuras como anteriormente. Encontramos todas las posibles intersecciones. Si al encontrar un evento, este evento involucra segmentos de S1 y S2 debemos hacer cambios locales a D para contener los registros adecuados de la superposicion.

Cuando un edge de S1 se encuentra con un vertice de S2

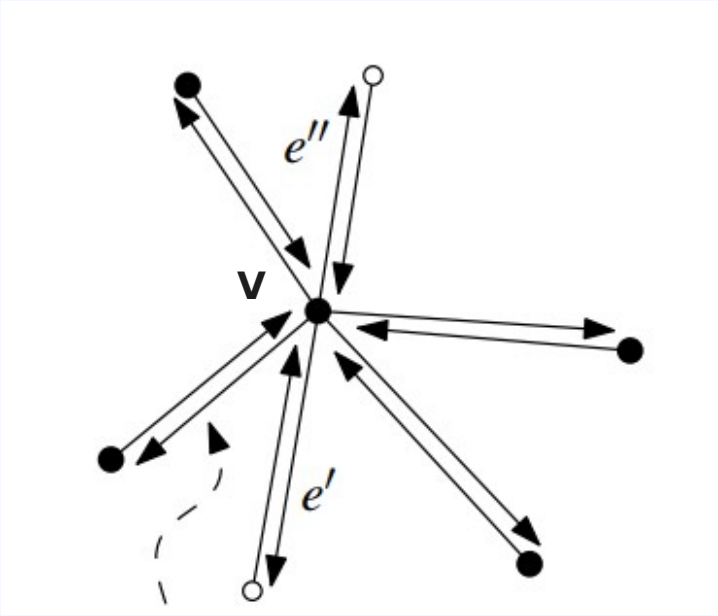


De los dos half-edges de e debemos tener 4, mantenemos dos con origen los extremos del segmento de e y creamos registros de los edges adicionales con sus respectivos Twin(.).

Ahora debemos cambiar Prev(.) y Next(.), primero para los nuevos half edges e’, e”

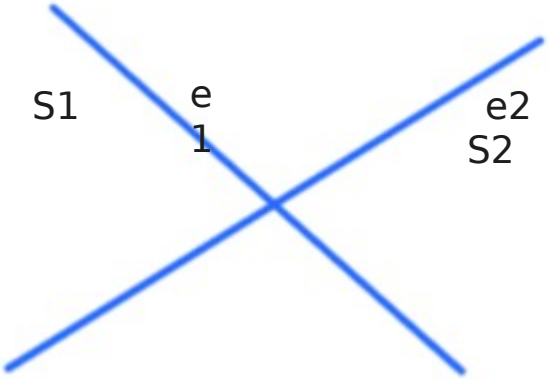


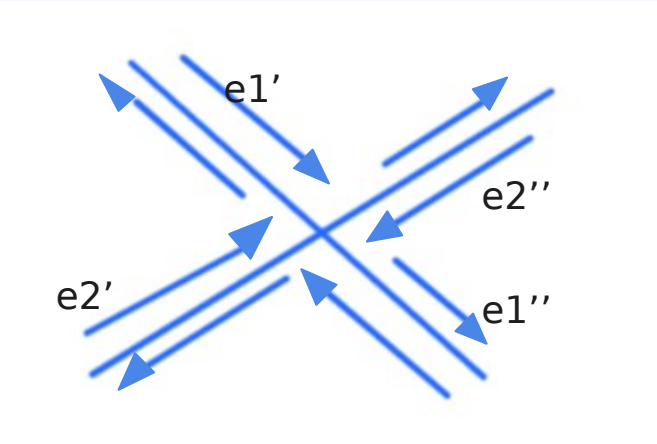
Next() es el Next() del half-edge anterior, el Prev() de estos half-edges es e’, e’’



Debemos cambiar Prev() y Next() alrededor de v para poder hacer el recorrido en el sentido del reloj

En el caso de encontrar intersección de edges de S1 y S2



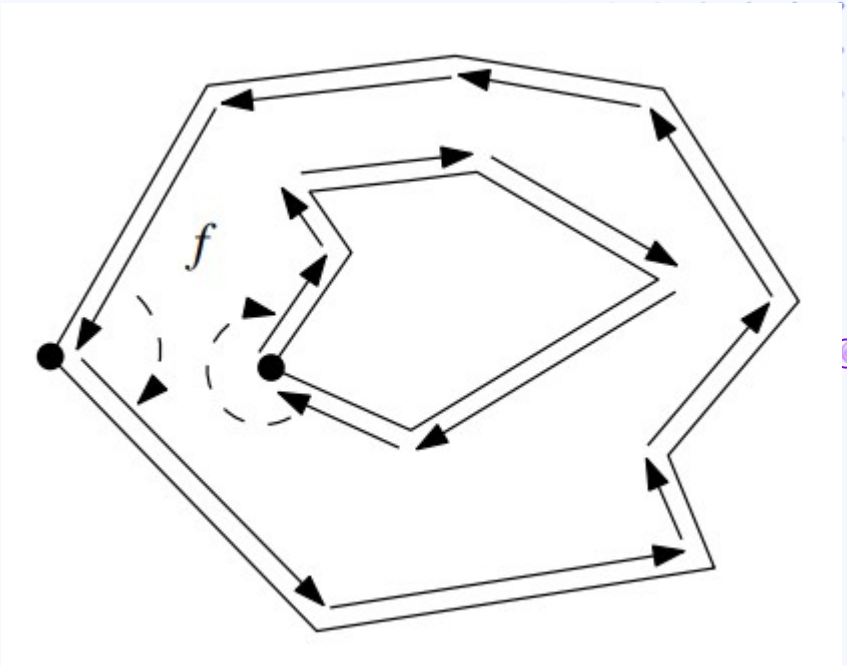


Se deben intercambiar e1 y e2, con sus half-edges por los respectivos half-edges e1’, e1”, e2’, e2” y Prev(.) y Next(.).

En el caso de encontrar intersección de vertices, v, se deben reasignar en el orden los Next() y Prev() que tengan como destino, origen v.

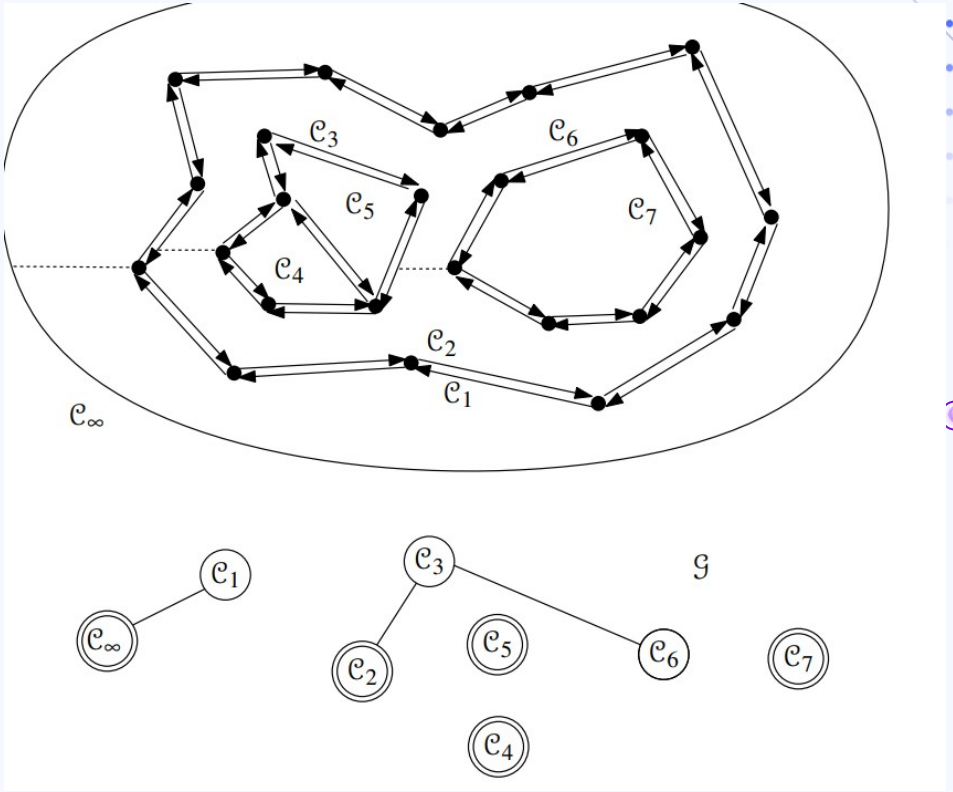
Para el caso de faces, el número de faces serán las diferentes componentes conexas de los bordes exteriores + 1 (cada face tiene una componente conexa excepto el unbounded face exterior).

Para saber si un ciclo de half-edges bordea una face en el interior o exterior se puede usar el siguiente criterio, se toma el vertice de menor componente x, en caso de empate el de menor componente y, luego se mira el angulo de los half edges alrederor de v, si es menor que 180 es un face exterior, de los contrario interior

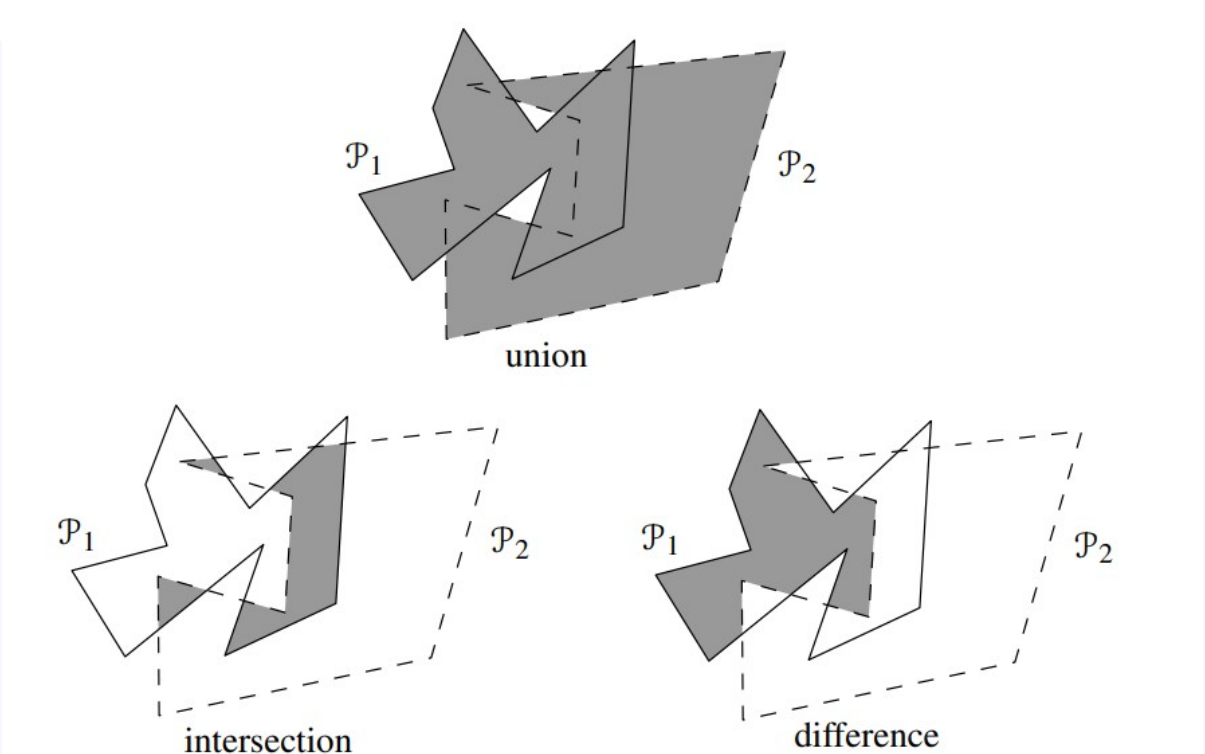


Para decidir que fronteras bordean la misma face construimos un grafo G. para cada ciclo frontera —inner y outer—hay un nodo en G. Hay un nodo por la frontera de la unbounded face. Existe arco entre los ciclos si solo si uno de los ciclos es una frontera de un hueco y el otro tiene un half-edge inmediatamente a la izquierda del vértice más a la izquierda de la frontera del hueco.

Las componentes conexas del grafo son las distintas faces.



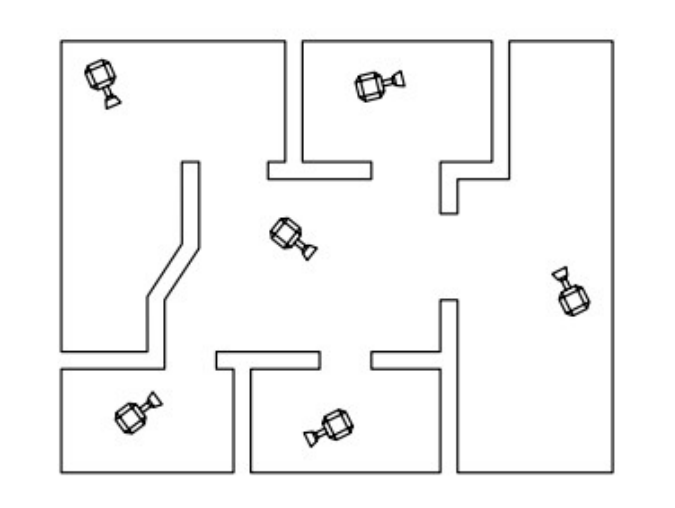
La solución al problema de superposición de mapas puede usarse para realizar operaciones Booleanas sobre polígonos, si se guarda información de a que estructura pertenece cada cara se puede mostrar cada cara según la operación



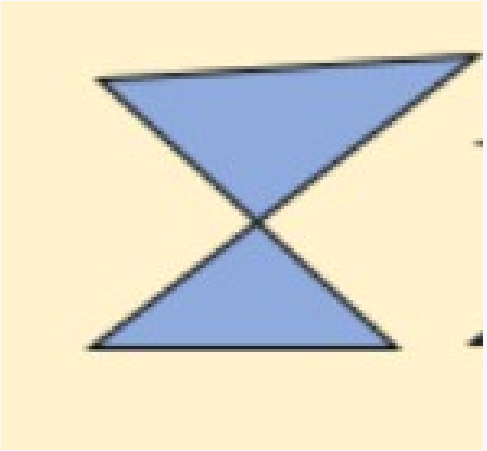
Triangulación de Polígonos

Motivamos esta sección con la formulación del siguiente problema, tenemos una galería que queremos custodiar por unas cameras, cómo minimizar las cámaras y dónde localizarlas para visualizar todo punto en la galería.

Escogemos poligonos simples, polígonos que no se auto intersectan y no tienen huecos..



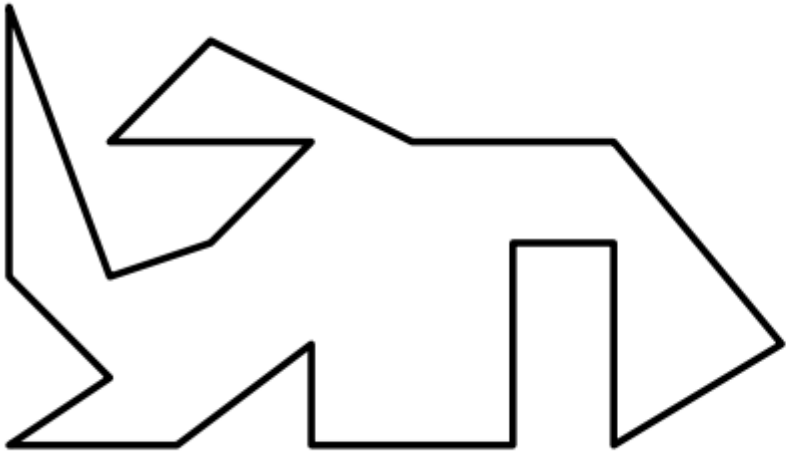
Cuántas cámaras necesitemos dependerá de la complejidad del polígono entre más complejo mayor número de cámaras. En una primera instancia el número de cámaras máximo dependerá de la complejidad del polígono, De n el numero de vertices?

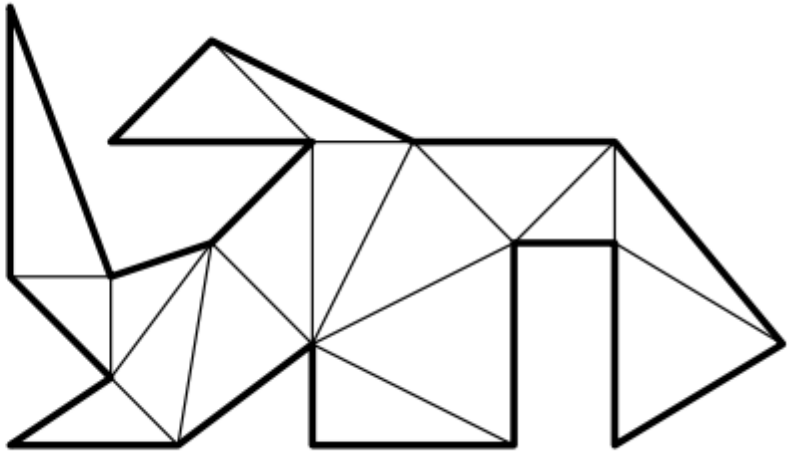


Encontrar el mínimo número de cámaras es un problema NP hard.
https://kottke.org/16/05/p-vs-np-and-the-computational-complexity-zoo

Supongamos que el poligono P tiene n vertices, primero lo descomponemos entre piezas fáciles de custodiar, dibujando diagonales, que son lineas contenidas dentro del polígono. Una triangulacion es la descomposición de P en un conjunto maximal de diagonales sin intersecciones (se impone la maximalidad para hacer que cada triangulo formado no contenga puntas)

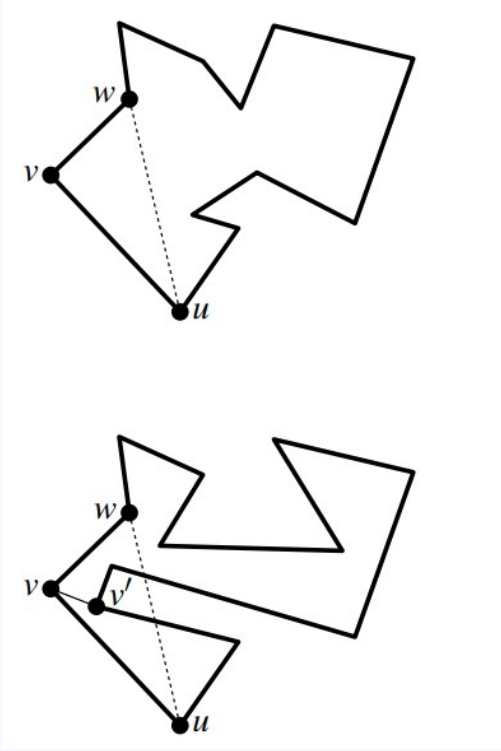
Una triangulacion





Teorema *cada polígono admite una triangulación, y tiene exactamente n -2 triangulos, con n el numero de vertices.*

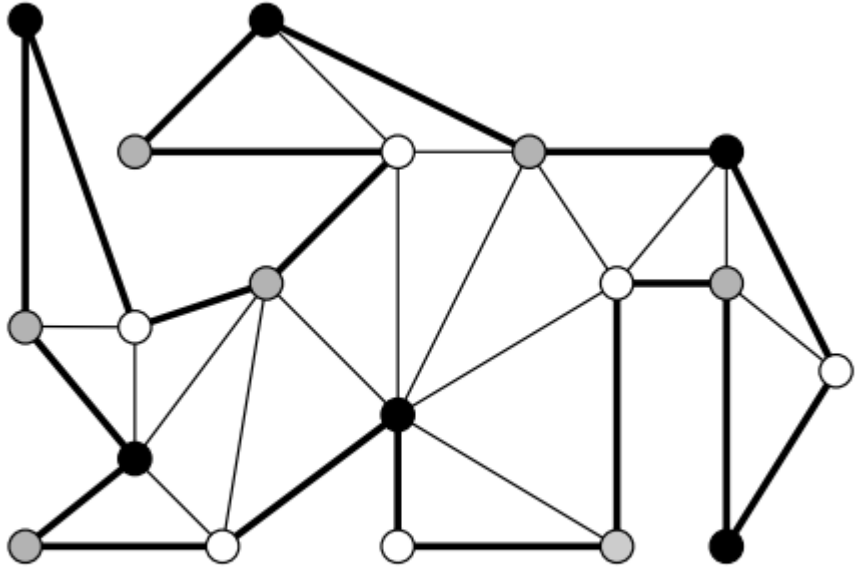
Por induccion fuerte en n, para la existencia triangulacion. El caso de 3 vertices es trivial. Si tomamos el vértice más hacia la izquierda y hacia abajo, v, tomamos sus vecinos, w , u,
si wu es no es una diagonal existen vertices mas hacia la izquierda dentro del poligono, tomamos v' el punto mas ajelado hacia la izquierda de esta linea. vv' tiene que ser una diagonal de lo contrario escogiendo la escogencia de v'. Esta linea divide el poligono en dos donde se puede aplicar la hipotesis de induccion y concluir.



Para la prueba del numero de triangulos usamos la misma idea teniendo en cuenta que si partimos P de n vertices en dos pologinos $m_1 + m_2 = n + 2$ ya que hay dos vertices repetidos.

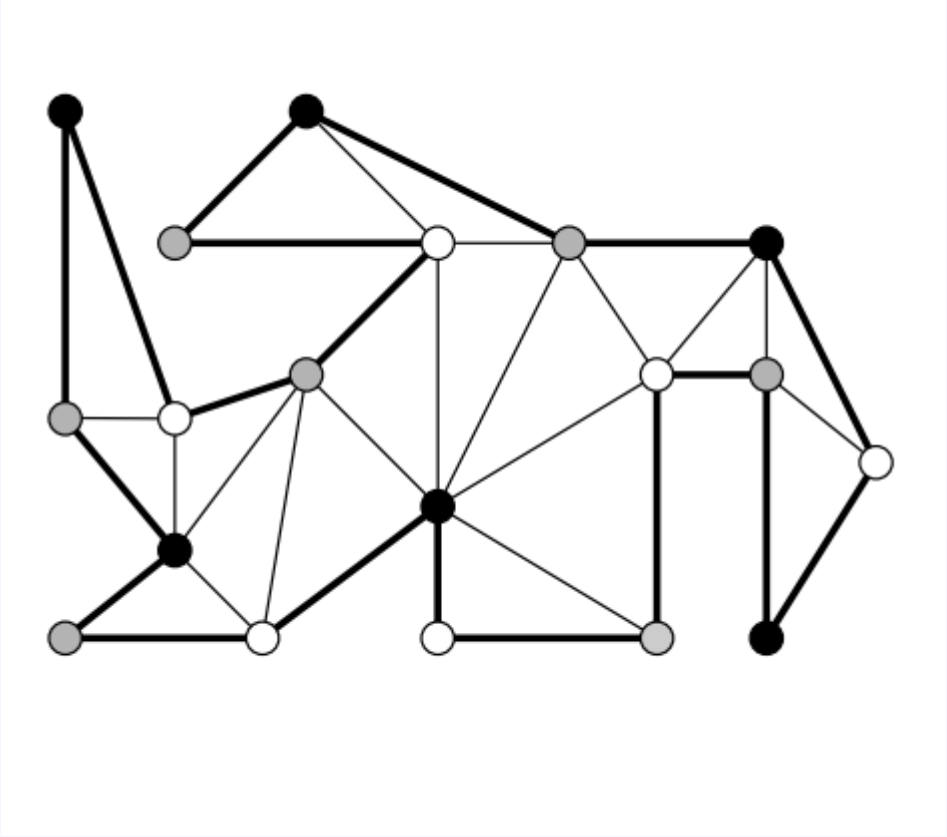
Usando una camara por vertice es demasiado...
Tomando T_P seleccionamos un subconjunto de los vértices de P tal que cada triángulo tiene un vértice, localizamos las camaras alli. Para encontrar estos vertices usamos 3 colores, blanco, gris y negro. Es tal que dos vértices conectados por edge o diagonal comparten color.

Cada polígono simple se puede colorear de esta manera, (3-coloring) de manera que si localizamos una cámara en cada color tenemos



Teorema *Para un polígono simple de n vertices [n/3] camaras son suficientes y a veces necesarias para localizar todas las áreas en el problema de la galería de arte.*

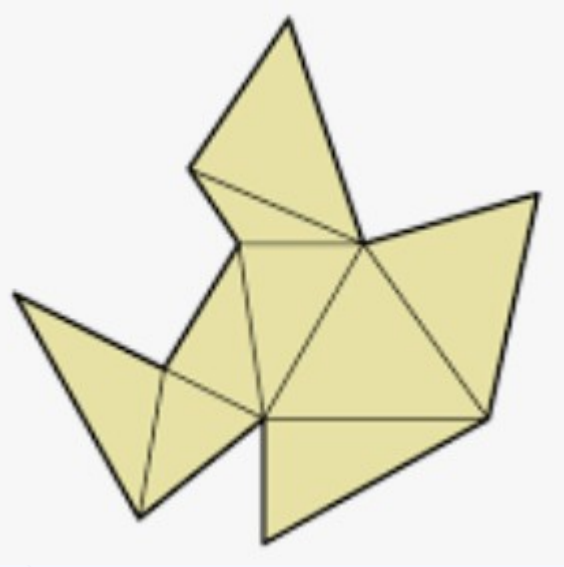
Para ver esto usamos $G(T_P)$, grafo que tiene un nodo por cada triángulo en T_P , Denotamos el triángulo correspondiente a un nodo v como t(v).
Hay un arco entre dos nodos v y m si t(v) y t(m) comparten una diagonal. Los arcos en $G(T_P)$,corresponden a diagonales, como cada diagonal corta la triangulacion en dos, la eliminación de un arco de $G(T_P)$ divide el grafo en dos.$G(T_P)$ es un árbol.

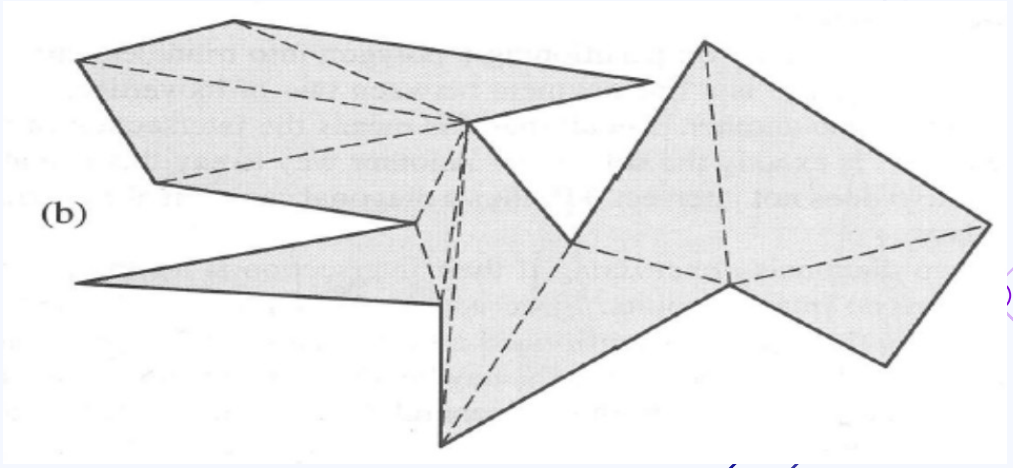


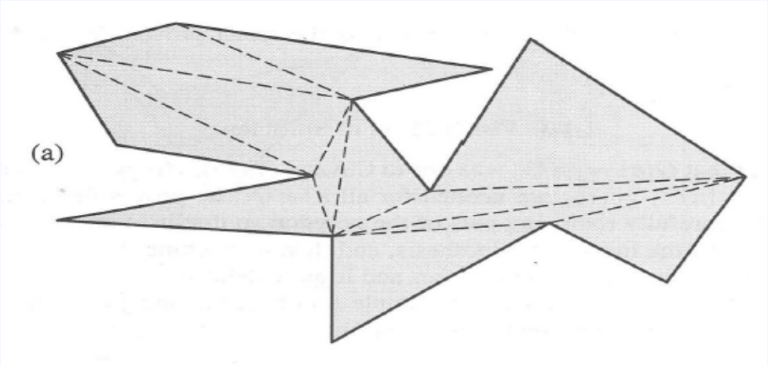
Con un recorrido de grafo podemos encontrar los colores, DFS. En DFS mantenemos el siguiente invariante, los vértices de los triángulos encontrados hasta ya han sido coloreados en blanco, gris o negro, siguiendo las reglas. Ahora supongamos que llegamos a un nodo v en G, viniendo del nodo m. Por lo tanto, t(v) y t(m) comparten una diagonal. Dado que los vértices de t(m) ya han sido coloreados, solo queda un vértice de t(v) por colorear. Queda un color para este vértice, es decir, el color que no se usa para los vértices de la diagonal entre t(v) y t(m). Como el grafo es un árbol se puede recorrer de esta forma y completar con 3 colores.

Esto prueba el teorema, pero para encontrar esos colores necesitamos una triangulación de P. Necesitamos un algoritmo (eficiente) para generar una triangulación

Algunas triangulaciones







La prueba de la existencia de las triangulaciones es constructiva.
Tomamos el vertice mas hacia la izquierda, v, y sus dos vecinos w, u si vw no es una diagonal tomamos v’ el vertice dentro del triangulo formado por v, u, w que se encuentre más alejado de la línea uw

Contents of this template

You can delete this slide when you're done editing the presentation

Fonts	To view this template correctly in PowerPoint, download and install the fonts we used
Used and alternative resources	An assortment of graphic resources that are suitable for use in this presentation
Thanks slide	You must keep it so that proper credits for our design are given
Colors	All the colors used in this presentation
Icons and infographic resources	These can be used in the template, and their size and color can be edited
Editable presentation theme	You can edit the master slides easily. For more info, click **here**

For more info:
Slidesgo | Blog | FAQs

You can visit our sister projects:
Freepik | Flaticon | Storyset | Wepik | Videvo

Table of contents

01

Theory lesson

You can enter a subtitle here if you need it

Introduction

Mercury is the closest planet to the Sun and the smallest one in the entire Solar System. **This planet’s name has nothing to do with the liquid metal,** since Mercury was named after the Roman messenger god. Mercury’s surface is filled with craters

Mercury takes a little more than 58 days to complete its rotation, so try to imagine how long days must be there! **Since the temperatures are so extreme, albeit not as extreme** as on Venus, Mercury has been deemed to be non-habitable for humans

“This is a quote, words full of wisdom that someone important said and can make the reader get inspired”

—Someone Famous

Concepts

Mercury

Mercury is the closest planet to the Sun and **the smallest one** in the Solar System—it’s only a bit larger than the Moon

Venus

Venus has a beautiful name and is the **second planet from the Sun.** It’s hot and has a poisonous atmosphere

What is this topic about?

Mercury

It’s the closest planet to the Sun and the **smallest** in the Solar System

Venus

Venus has a beautiful name and is the second planet from the Sun

Mars

Despite being red, Mars is actually a **cold place.** It’s full of iron oxide dust

Features of the topic

Mars

Despite **being red**, Mars is very cold

Neptune

It’s the farthest planet from the Sun

Jupiter

Jupiter is the biggest planet of them all

Saturn

Saturn is a **gas giant** and has several rings

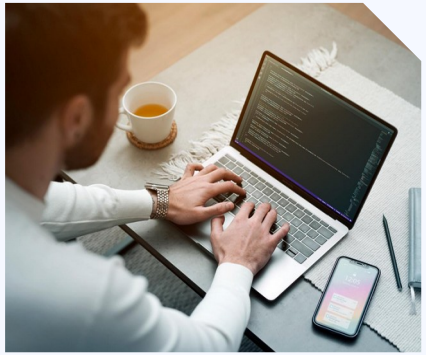
Examples

Mercury

It’s the closest planet to the Sun and the **smallest** in the Solar System

Venus

Venus has a beautiful name and is the second planet from the Sun



Mars

Despite being red, Mars is actually a **cold place.** It’s full of iron oxide dust

Recommendations

Mars

Despite being red, Mars is very cold

Venus

Venus is the second planet from the Sun

Neptune

Neptune is the farthest planet from the Sun

Mercury

Mercury is the closest planet to the Sun

Saturn

Saturn is a gas giant and has several rings

Jupiter

Jupiter is the biggest planet of them all

Image always reinforce the concept

You can give a brief description of the topic you want to talk about here. For example, if you want to talk about Mercury, you can say that it’s the smallest planet in the entire Solar System

4,498,300,000

Big numbers catch your audience’s attention

9h 55m 23s

Jupiter’s rotation period

333,000

The Sun’s mass compared to Earth’s

386,000 km

Distance between the Earth and the Moon

Awesome words

A picture is worth a thousand words

Practical exercise - calculator

Objective:

Introduce participants to basic coding concepts by building a **simple calculator**

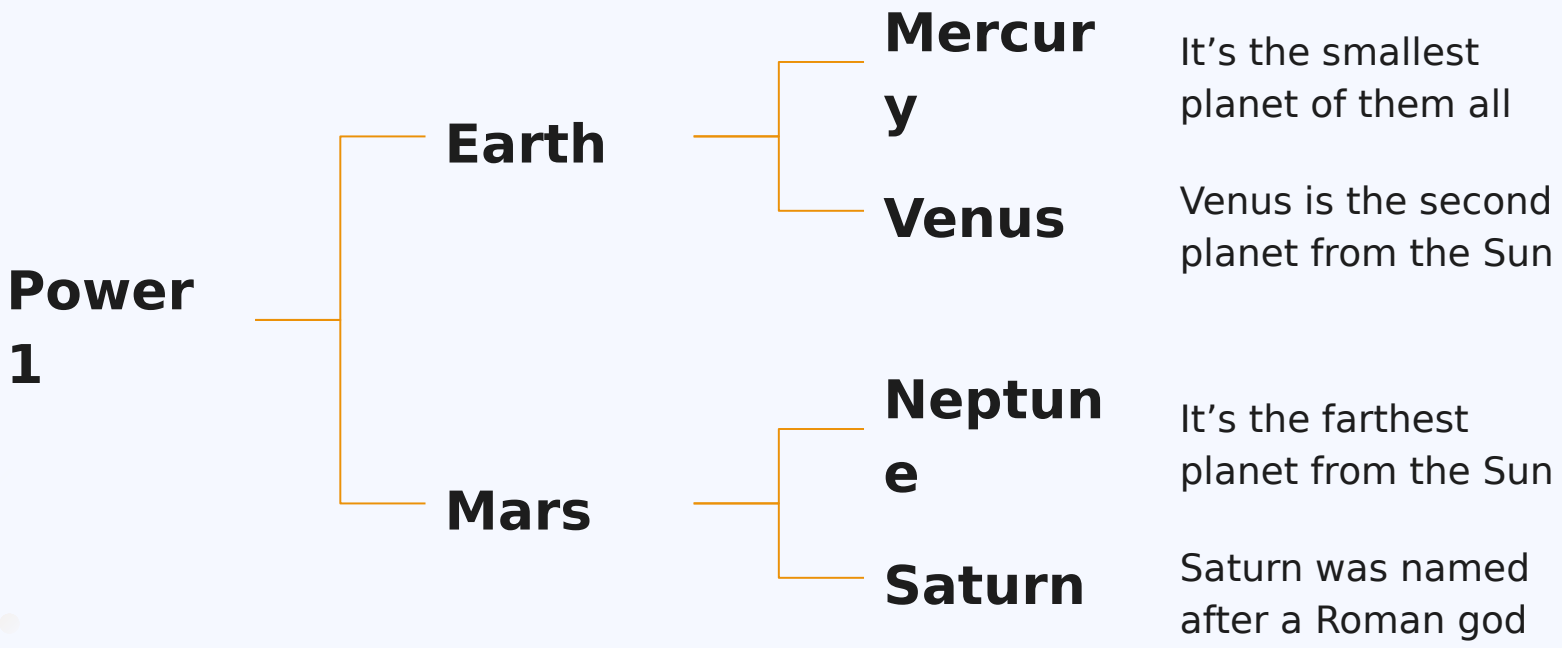
Instructions:

1. Open a Python development environment and write the following code:

```
# Simple Calculator
num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))
print("Sum:", num1 + num2)
print("Difference:", num1 - num2)
print("Product:", num1 * num2)
print("Quotient:", num1 / num2)
```

2. Run the program and experiment with different numbers
3. Observe the output

Brainstorm and idea generation



Main topic and details

Mars

Despite being red, Mars is **very cold**

Neptune

It’s the farthest planet from the Sun

Jupiter

Jupiter is the biggest planet of them all

Saturn

It’s a gas giant and has **several rings**

Popular programming languages

01 — **Neptune**

Mercury is the closest planet to the Sun and the **smallest** of them all

02 — **Venus**

Venus has a beautiful name and is the **second planet from the Sun**

03 — **Earth**

Earth is the third planet from the Sun and the only one that harbors life in the Solar System

04 — **Saturn**

Saturn is a gas giant and has several rings. It’s composed mostly of hydrogen and helium

Sequences

Saturn is composed of **hydrogen and helium**

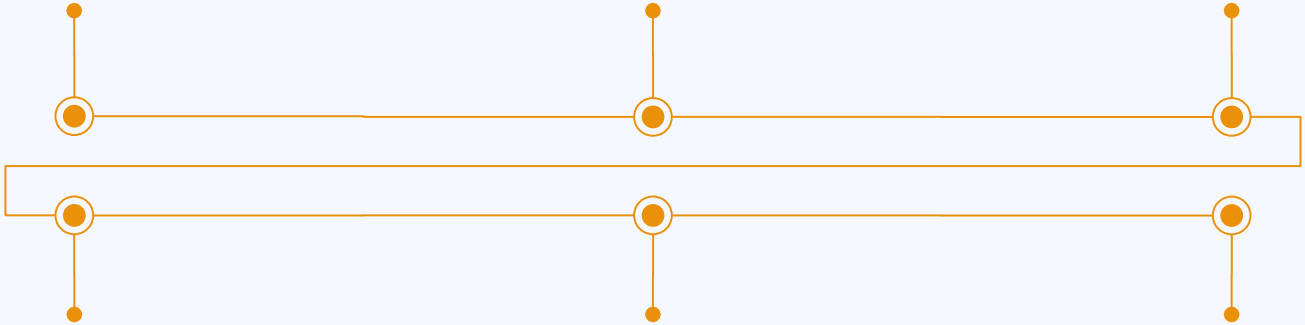
First

Mercury is the **closest** planet to the Sun

Next

Jupiter was named after a Roman god

Next



Despite being red, Mars is **very cold**

Next

Earth is the third planet from the Sun

Next

Venus has extremely high temperatures

Last

Classification

Mars	Venus	Mercury	Jupiter
● Small ● Red ● Cold ● Rocky	● Small ● Hot ● Dry ● Volcanic	● Small ● Hot ● Rocky ● Cratered	● Large ● Cold ● Gassy ● Striped
Mars is full of iron oxide dust	Venus has high temperatures	Mercury is quite a small planet	Jupiter is a huge gas giant

Cause and effect

Problem

Mars

Despite being red, Mars is very cold

Venus

Venus is the second planet from the Sun

Solution

Mercury

Mercury is the closest planet to the Sun

Saturn

Saturn is a gas giant and has several rings

Question and answer

Question

Is Mercury the closest planet to the Sun and the smallest one in the Solar System? **Note that it’s a bit larger than the Moon**

Answer

Venus has a beautiful name and is **the second planet from the Sun.** It’s hot and has a poisonous atmosphere

Step-by-step coding

01

Earth

It’s the only planet known to **harbor life**

02

Mercury

Mercury is the closest planet to the Sun

03

Jupiter

Jupiter is the **biggest** planet of them all

04

Saturn

Saturn was named after a Roman god

Parts and whole

The whole objective

Mercury is the closest planet to the Sun and the smallest one in the entire Solar System

Parts of the object

- Mercury
- Jupiter
- Venus
- Mars
- Earth
- Saturn
- Mercury

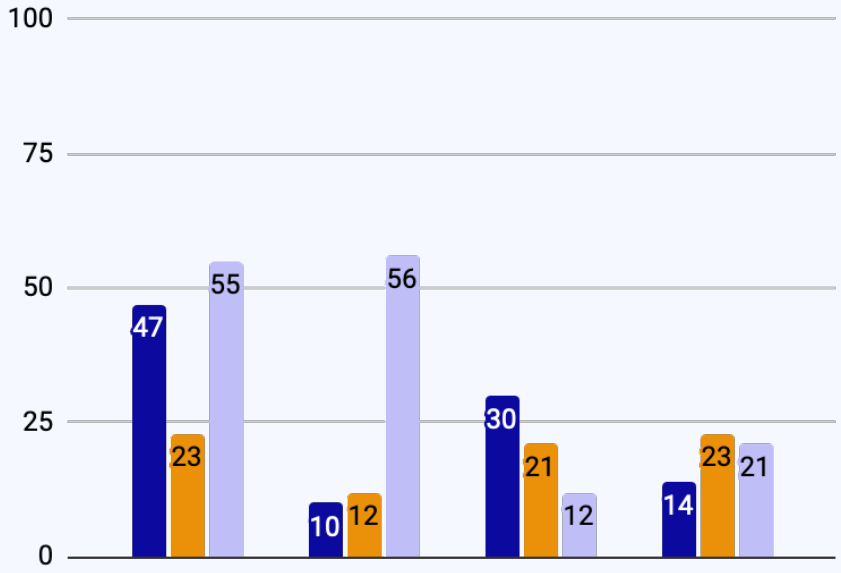
What happens if the parts are missing?

Earth is the third planet from the Sun and the **only one that harbors life in the Solar System**

What’s the function of the parts?

Jupiter is a gas giant and the biggest planet in the Solar System

You can use this graph



Follow the link in the graph to modify its data and then paste the new one here. **For more info, click here**

Mercury

Mercury is the closest planet to the Sun

Jupiter

Jupiter is the biggest planet of them all

Saturn

Saturn was named after a Roman god

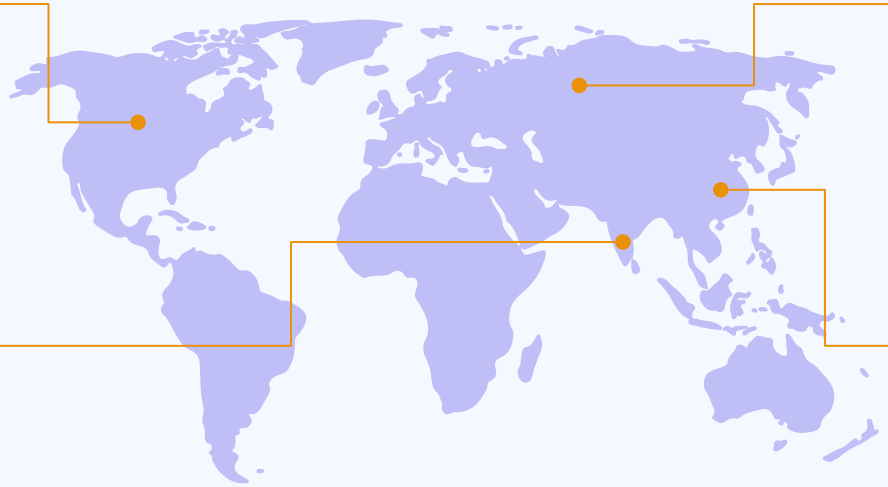
This is a map

USA

Despite being red, Mars is **very cold**

India

Jupiter is the biggest planet of them all



Russia

Neptune is the farthest planet from the Sun

China

Saturn is **a gas giant** and has several rings

Mockups

You can replace the images on the screen with your own work. Just right-click on them and select “Replace image”

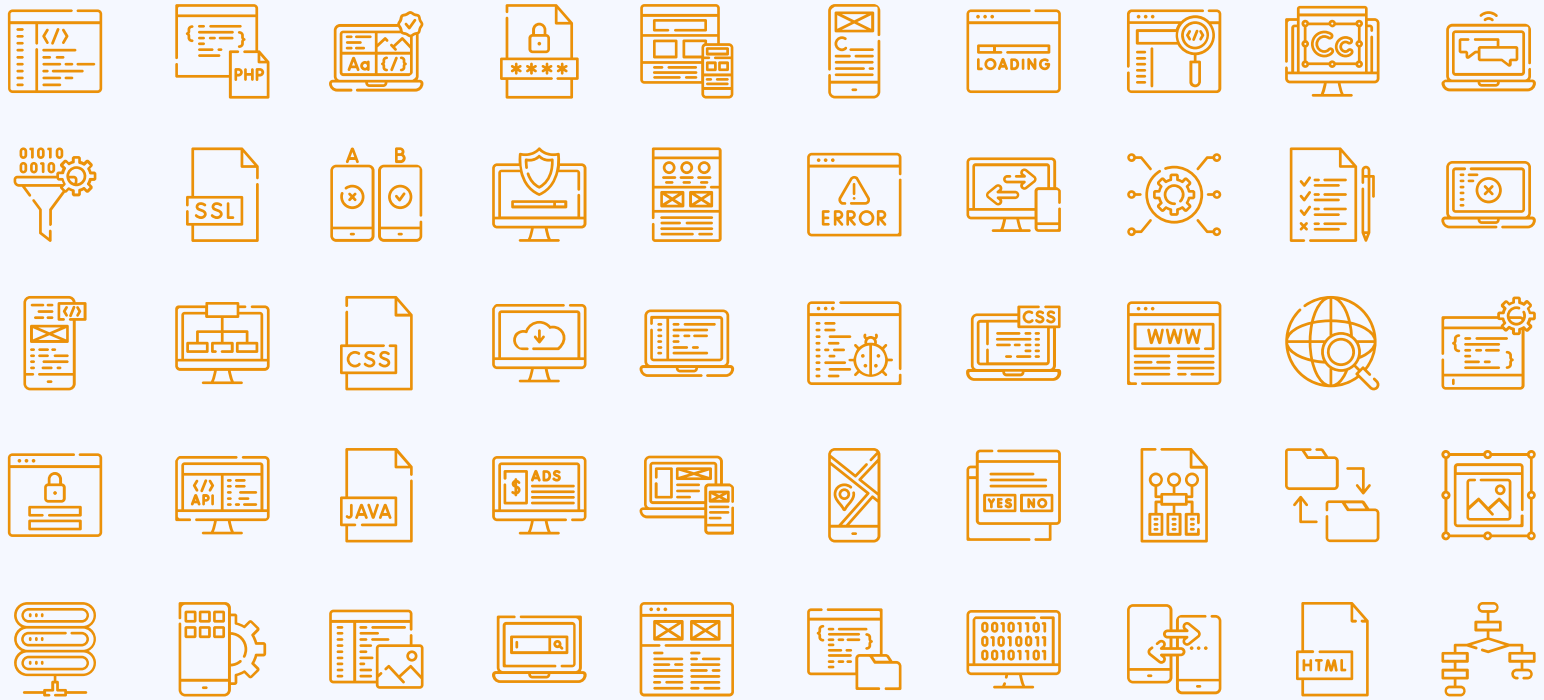
Thanks!

Do you have any questions?

youremail@freepik.com
+34 654 321 432
yourwebsite.com

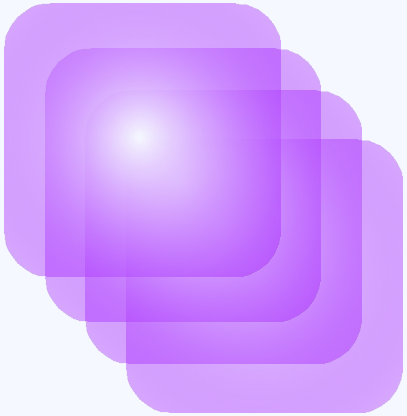


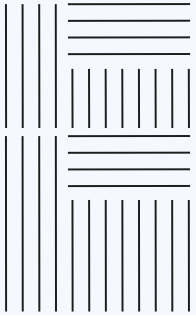
Icon pack

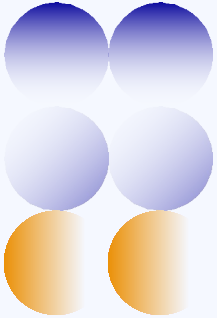

PHP
****
LOADING
01010
0010
SSL
A
B
ERROR
CSS
WWW
ADS
API
JAVA
YES
NO
00101101
01010011
00101101
HTML

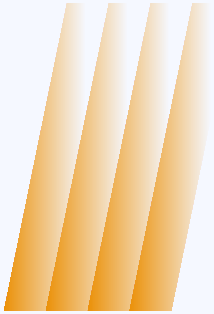
Alternative resources

Here’s an assortment of alternative resources whose style fits that of this template:









Resources

Did you like the resources in this template? Get them for free at our other websites:

Photos:

- Medium shot man working on laptop
- Group of friends planning a trip in a cafe
- Medium shot man working on laptop
- Side view of men working on laptops at the office
- Lifestyle of woman in the office
- Secretary working on laptop

Vectors:

- Abstract gradient circuit board background

Icon pack:

- Icon Pack: Coding | Lineal

Instructions for use

If you have a free account, in order to use this template, you must credit **Slidesgo** by keeping the **Thanks** slide. Please refer to the next slide to read the instructions for premium users.

As a Free user, you are allowed to:

- Modify this template.
- Use it for both personal and commercial projects.

You are not allowed to:

- Sublicense, sell or rent any of Slidesgo Content (or a modified version of Slidesgo Content).
- Distribute Slidesgo Content unless it has been expressly authorized by Slidesgo.
- Include Slidesgo Content in an online or offline database or file.
- Offer Slidesgo templates (or modified versions of Slidesgo templates) for download.
- Acquire the copyright of Slidesgo Content.

For more information about editing slides, please read our FAQs or visit our blog:
https://slidesgo.com/faqs and **https://slidesgo.com/slidesgo-school**

Instructions for use (premium users)

As a Premium user, you can use this template without attributing **Slidesgo** or keeping the **Thanks** slide.

You are allowed to:

- Modify this template.
- Use it for both personal and commercial purposes.
- Hide or delete the “Thanks” slide and the mention to Slidesgo in the credits.
- Share this template in an editable format with people who are not part of your team.

You are not allowed to:

- Sublicense, sell or rent this Slidesgo Template (or a modified version of this Slidesgo Template).
- Distribute this Slidesgo Template (or a modified version of this Slidesgo Template) or include it in a database or in any other product or service that offers downloadable images, icons or presentations that may be subject to distribution or resale.
- Use any of the elements that are part of this Slidesgo Template in an isolated and separated way from this Template.
- Register any of the elements that are part of this template as a trademark or logo, or register it as a work in an intellectual property registry or similar.

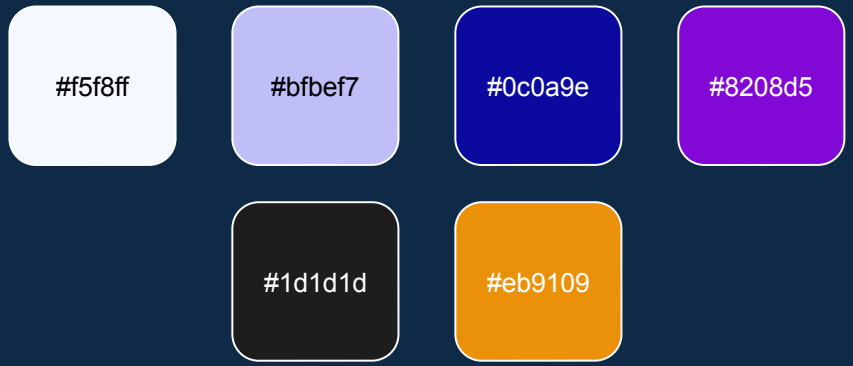
For more information about editing slides, please read our FAQs or visit our blog:
https://slidesgo.com/faqs and https://slidesgo.com/slidesgo-school

Fonts & colors used

This presentation has been made using the following fonts:

IBM Plex Mono Bold
(https://fonts.google.com/specimen/IBM+Plex+Mono)

Poppins Normal & Bold
(https://fonts.google.com/specimen/Poppins)



Storyset

Create your Story with our illustrated concepts. Choose the style you like the most, edit its colors, pick the background and layers you want to show and bring them to life with the animator panel! It will boost your presentation. Check out **how it works**.

Pana

Amico

Bro

Rafiki

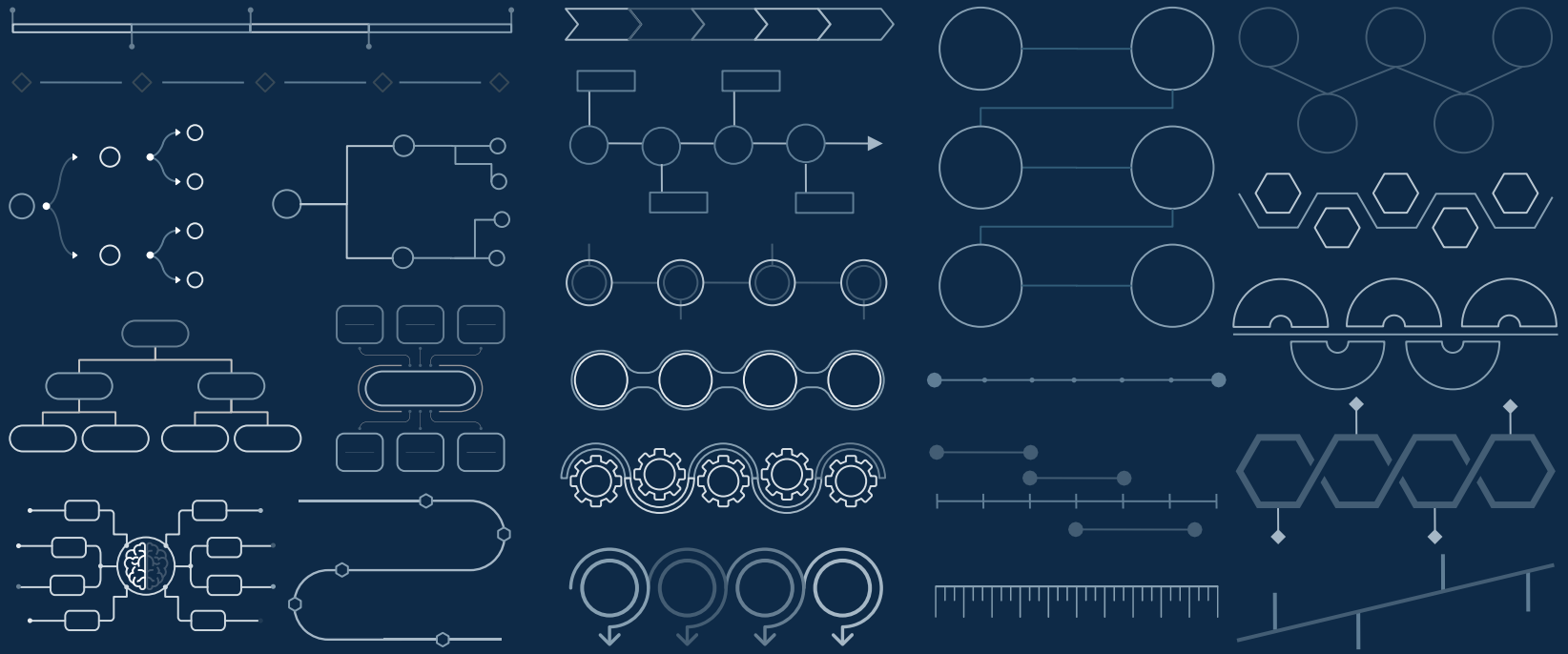
Cuate

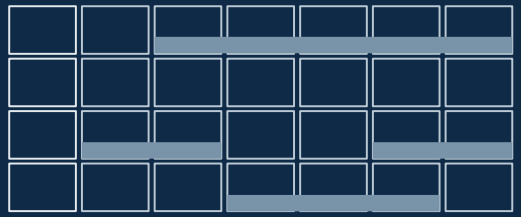
Use our editable graphic resources...

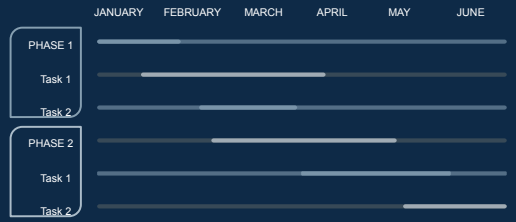
You can easily **resize** these resources without losing quality. To **change the color**, just ungroup the resource and click on the object you want to change. Then, click on the paint bucket and select the color you want. Group the resource again when you're done. You can also look for more **infographics** on Slidesgo.

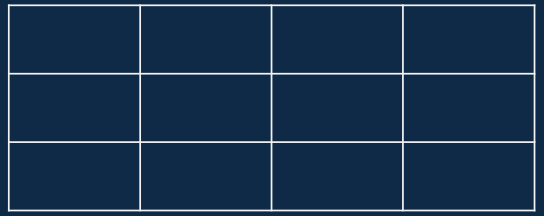


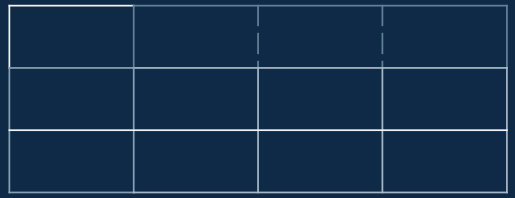


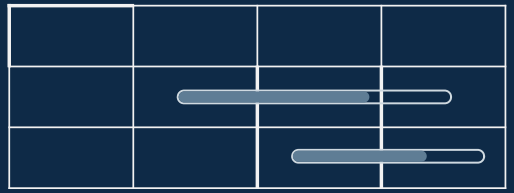



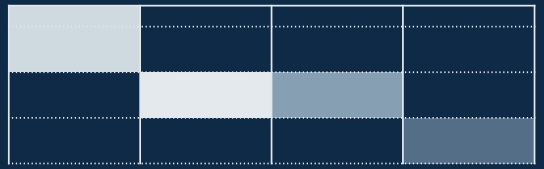


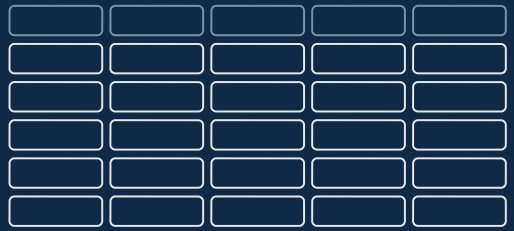

JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
PHASE 1
Task 1
Task 2
PHASE 2
Task 1
Task 2

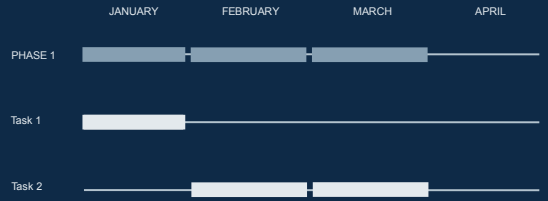


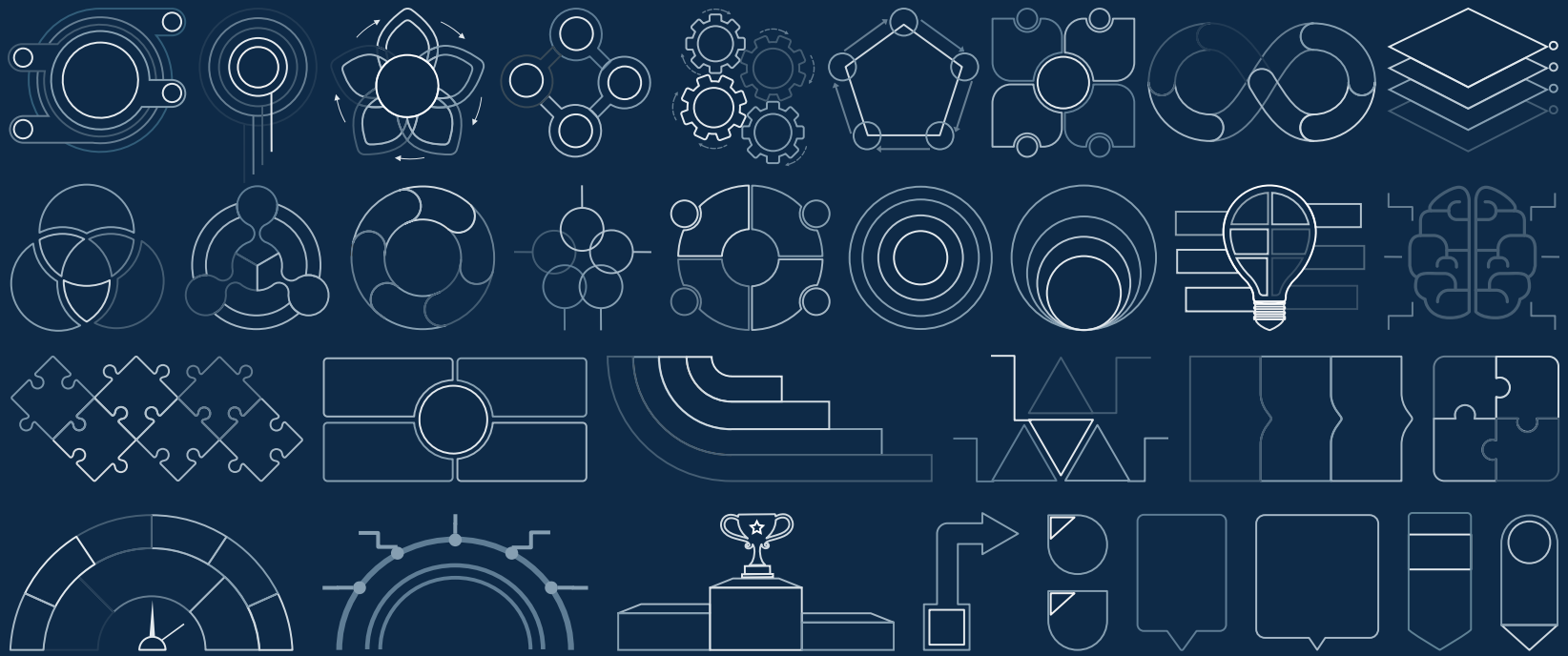


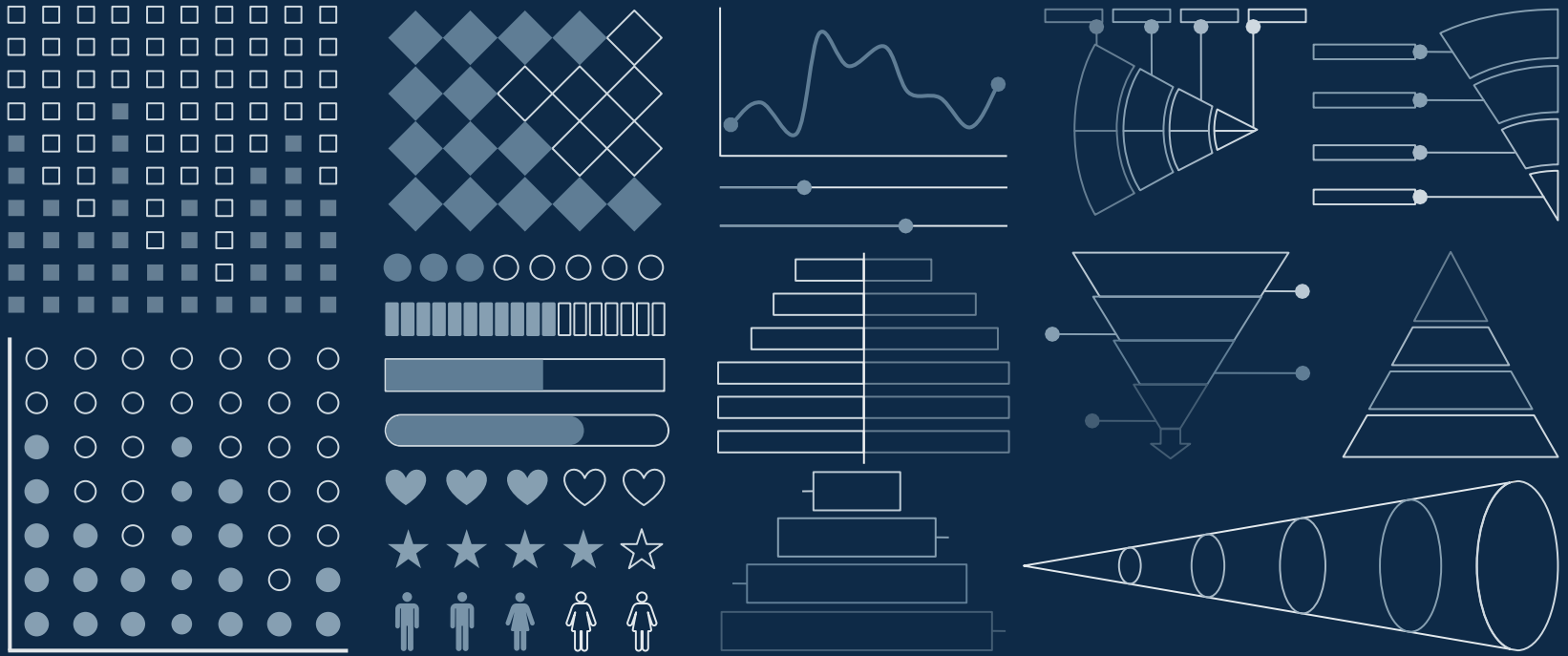







JANUARY
FEBRUARY
MARCH
APRIL
PHASE 1
Task 1
Task 2





...and our sets of editable icons

You can **resize** these icons without losing quality.
You can **change the stroke and fill color**; just select the icon and click on the **paint bucket/pen**.
In Google Slides, you can also use **Flaticon's extension**, allowing you to customize and add even more icons.

Educational Icons

Medical Icons

Business Icons

Teamwork Icons

Help & Support Icons

Avatar Icons

Creative Process Icons

Performing Arts Icons

Nature Icons

SEO & Marketing Icons


slidesgo