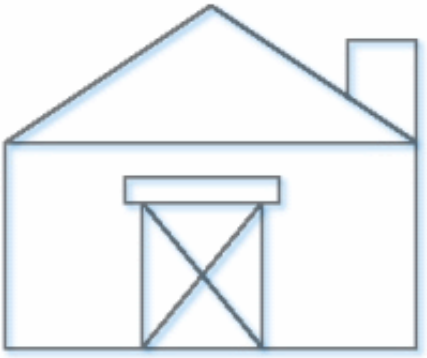
Les graphes eulériens ont de nombreuses applications, certaines ludiques.

Par exemple est-il possible de dessiner cette maison sans lever le crayon, et bien sûr sans repasser par le même trait ?



Ce problème correspond à déterminer si un graphe admet un *chemin eulérien*

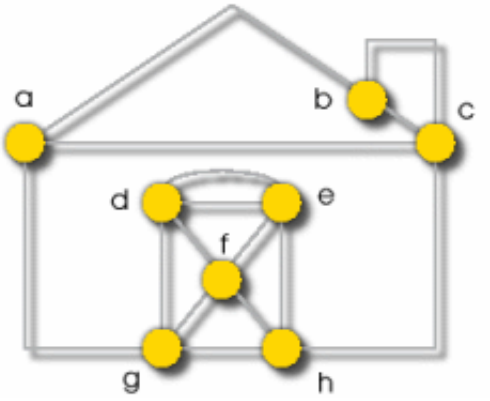
Définition **Chemin Eulérien**

Un **chemin eulérien** est un chemin passant une et une seule fois par chaque arête du graphe.
Un graphe admet un chemin eulérien ssi il est connexe et au plus 2 de ses sommets sont de degré impair.

Les conditions d'existence d'un chemin eulérien sont une conséquence du théorème pour les graphes eulériens. En effet si un graphe ***G*** n'admet pas de sommet de degré impair, il est alors eulérien, et il suffit de prendre comme chemin un cycle eulérien. Sinon le graphe ***G*** admet, de part la propriété des degrés, exactement 2 sommets de degré impair, disons ***x*** et ***y***. En ajoutant l'arête ***(x,y)***, on obtient un graphe eulérien : la suppression de l'arête ***(x,y)*** de son cycle eulérien montre l'existence dans ***G*** d'un chemin eulérien entre ***x*** et ***y***.

Le graphe représentant notre maison comporte 2 sommets de degré impair : ***a*** et ***b***. En ajoutant cette arête le graphe devient eulérien.

Il admet par conséquent un chemin eulérien entre ces 2 sommets. Autrement dit, il est possible dessiner la maison sans lever le trait.



La recherche d'un chemin eulérien dans un graphe revient à la recherche d'un cycle eulérien. Si nous connaissons les conditions d'existence pour un cycle eulérien, nous n'avons pas donner de méthode (un algorithme) permettant de le construire. La preuve du théorème des graphes eulériens nous fournit une idée pour construire un cycle eulérien sur un graphe, en recherchant récursivement des cycles sur des composantes connexes et en les fusionnant au fur et à mesure. L'algorithme récursif prend en entrée un graphe, non nécessairement connexe, dont tous les sommets sont de degré pair, et un sommet ***x***. Il retourne comme résultat un cycle eulérien sur la composante connexe de ***x***. Pour ce faire il commence par construire un cycle ϕ contenant ***x*** en utilisant un algorithme proche de l'algorithme de marquage utilisé dans la preuve de l'existence d'un cycle sur les graphes de degré supérieure à 2. Ce cycle ϕ est ensuite parcouru, et pour chacun de ses sommets x_i l'algorithme est rappelé récursivement pour construire un cycle ϕ_i. Les différents cycles sont finalement concaténés entre eux pour construire un cycle eulérien sur la composante connexe. L'opération de concaténation entre 2 chemins, ***(u,...,v)*** $\circ$ ***(u',...,v')***, retourne le chemin ***(u,...,v,u',...,v')***.

`ALGORITHME` Euler
`ENTREES` ***G=(V,E)*** un graphe dont tous les sommets sont de degré pair,***x*** un sommet de ***V***
`SORTIE` ϕ un cycle eulérien sur la composante connexe de ***x***

ϕ : LISTE des sommets du cycle dans l'ordre de parcours
Initialiser ϕ := (***x***)
*// Base de la récursivité : **x** est isolé*
`Si` ***x*** est un sommet isolé
`Alors`
    `Retourner` ϕ
`Sinon` *// On construit un cycle contenant **x***
    Initialiser ***y*** := ***x***
    `Tant Que` ***y*** n'est pas un sommet isolé
        Choisir ***z*** l'un de ses voisins
        Supprimer l'arête ***(y,z)*** ; ***y*** := ***z***
        $\phi \leftarrow y$ *// on ajoute le sommet au cycle*
    `Fin TantQue`
    *// Appel récursif sur chacun des **k** sommets du cycle ϕ en concaténant les réponses*
    `Retourner` ***Euler***(***G***, ϕ (1)) $\circ$ ... $\circ$ ***Euler***(***G***, ϕ (k))
`Fin Si`

Théoreme

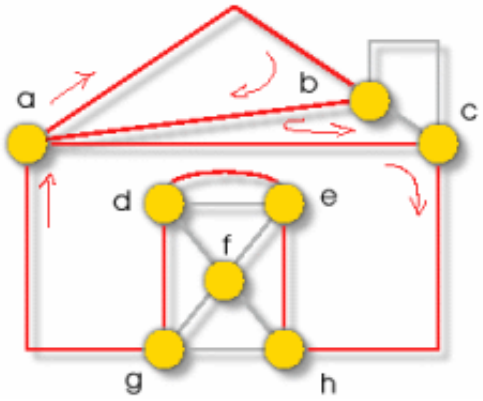
L'algorithme d'Euler construit un cycle eulérien en temps ***O(|E|)*** sur tout graphe eulérien ***G=(V,E)***

Preuve. Tout d'abord remarquons que la première phase de l'algorithme construit bien un cycle

contenant ***x***. En effet chaque fois que nous arrivons et repartons d'un sommet dans notre marche, nous supprimons 2 de ses arêtes incidentes. Tous les sommets étant de degré pair, seul le sommet de départ, ***x***, peut être déconnecté lors de l'arrivée à ce sommet.
Le fait que l'algorithme construit un cycle eulérien peut alors se montrer par induction sur le nombre d'arêtes du graphe, de manière similaire à la preuve du théorème d'Euler. Les arêtes du graphe étant supprimées au fur et à mesure de la construction, elles apparaissent bien exactement une fois dans le cycle final.

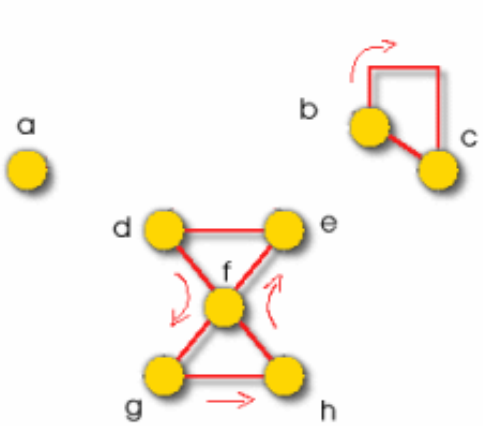
Complexité. A chaque fois qu'une arête du graphe est parcourue dans la première phase de l'algorithme pour construire le cycle ϕ, elle est supprimée. Le parcours du cycle pour effectuer les appels récursifs se fait en temps proportionnel au nombre de ses arêtes. La complexité totale de l'algorithme est donc bien en ***O(|E|)***. Il est optimal en temps, puisqu'un cycle eulérien comprend |***E***| arêtes.

Si nous appliquons l'algorithme d'Euler à notre maison, après ajout de l'arête ***(a,b)*** pour le rendre eulérien.



La première phase construit par exemple le cycle ***a,b,a,c,h,e,d,g,a*** en partant du sommet ***a***.

Evidemment d'autres choix dans le parcours du graphe peuvent conduire à des cycles différents, voire directement à un cycle eulérien.



Récursivement l'algorithme est appelé sur chacun des somemts du cycle.

- Le sommet ***a*** étant isolé, l'algorithme retourne immédiatement ***(a)***.
- Pour le sommet ***b***, l'algorithme construit récursivement le cycle ***(b,c,b)***.
- Le sommet ***c*** étant maintenant isolé, l'algorithme retourne ***(c)***.
- Pour le sommet ***h***, l'algorithme construit le cycle ***(h,f,e,d,f,g,h)***.
- Les sommets restant à visiter sur le cycle, ***(e,d,g,a)***, sont désormais tous isolés.

L'algorithme retourne le cycle ***(a)*** ○ ***(b,c,b)*** ○ ***(a)*** ○ ***(c)*** ○ ***(h,f,e,d,f,g,h)*** ○ ***(e)*** ○ ... ○ ***(a)***
c'est à dire ***(a,b,c,b,a,c,h,f,e,d,f,g,h,e,d,g,a)***