SAE - 02.02

Le problème du postier chinois

MOALIGOU ELISE
MALESTROIT LILWEN

iut Lannion

Présentation

Introduction

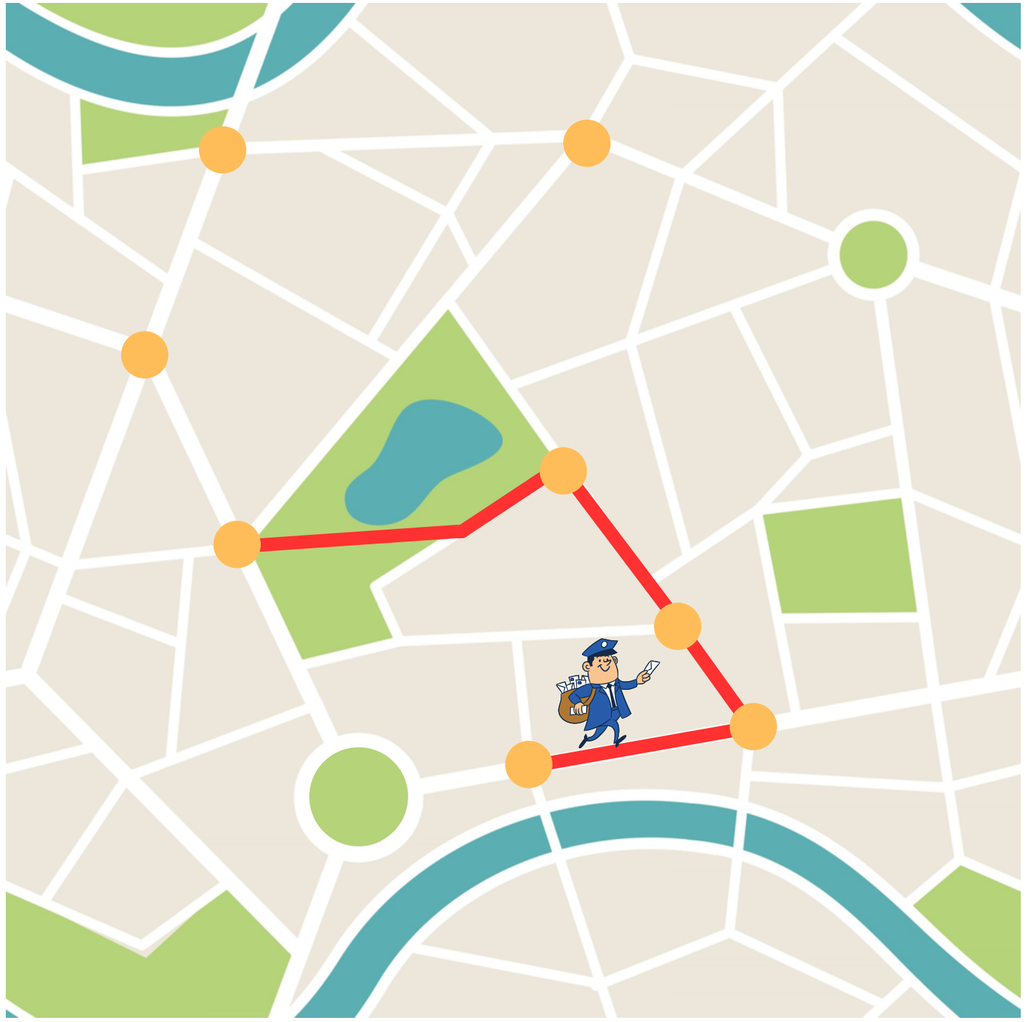
Algorithmes

Comparaison

Conclusion

Introduction

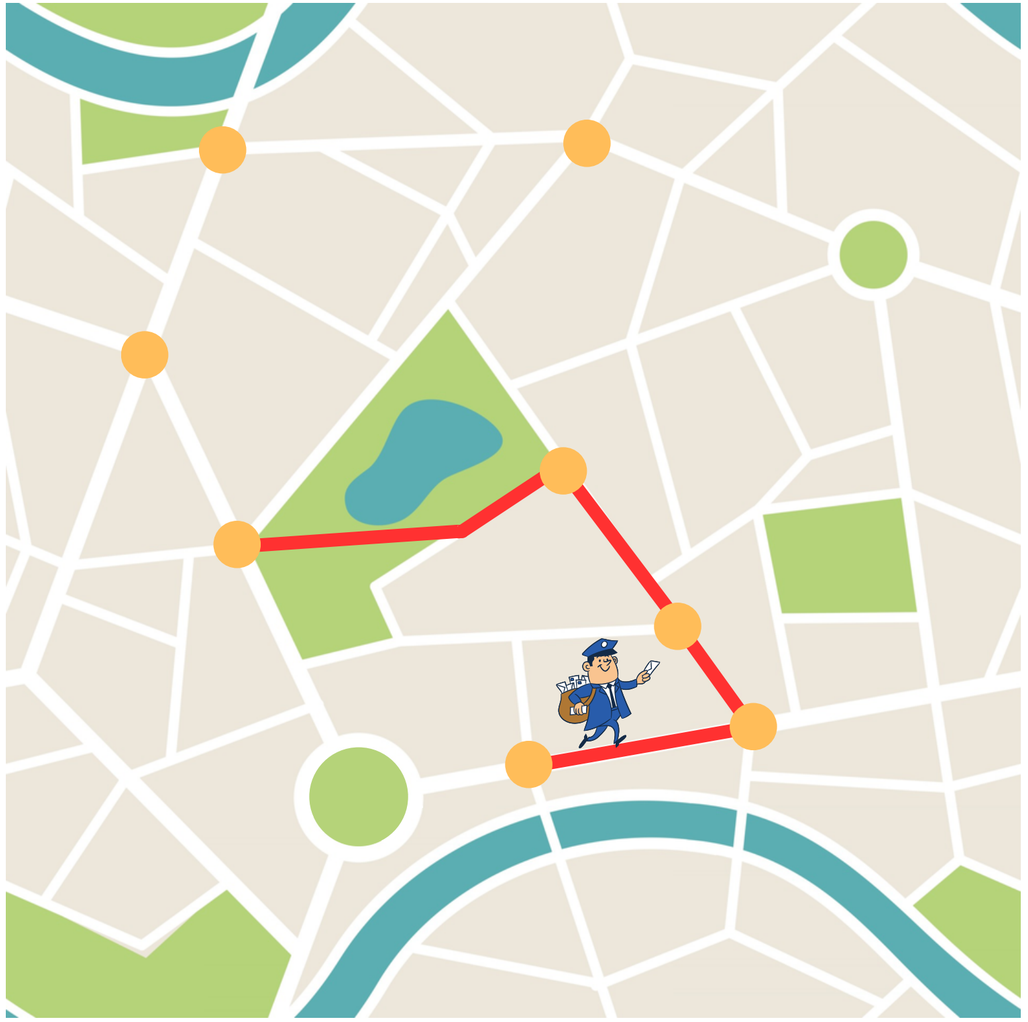
> Modélise le trajet d'un facteur distribuant le courrier



Introduction

Modélise le trajet d'un facteur distribuant le courrier

BUT Trouver le chemin le plus court

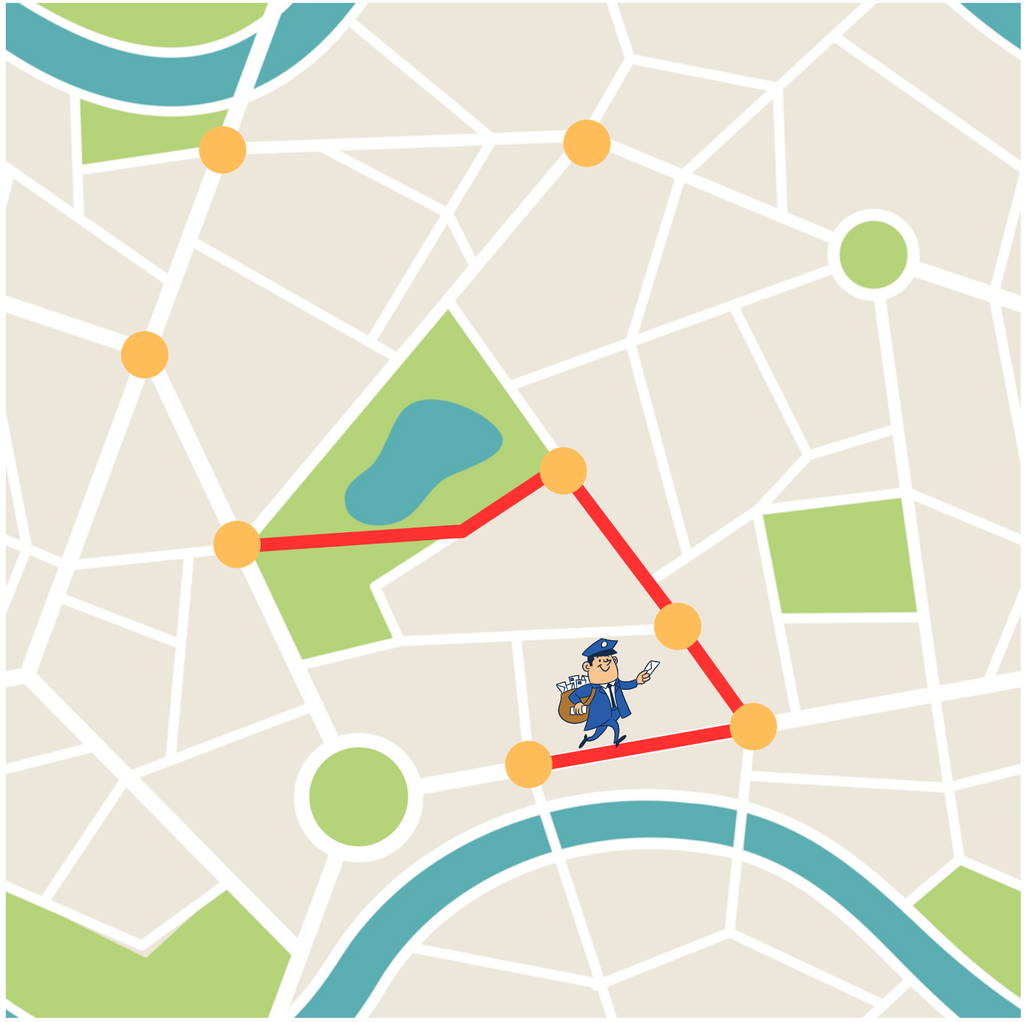


Introduction

Modélise le trajet d'un facteur distribuant le courrier

BUT Trouver le chemin le plus court

Ville == Graphe
Sommets == Intersections
Arêtes == Rues



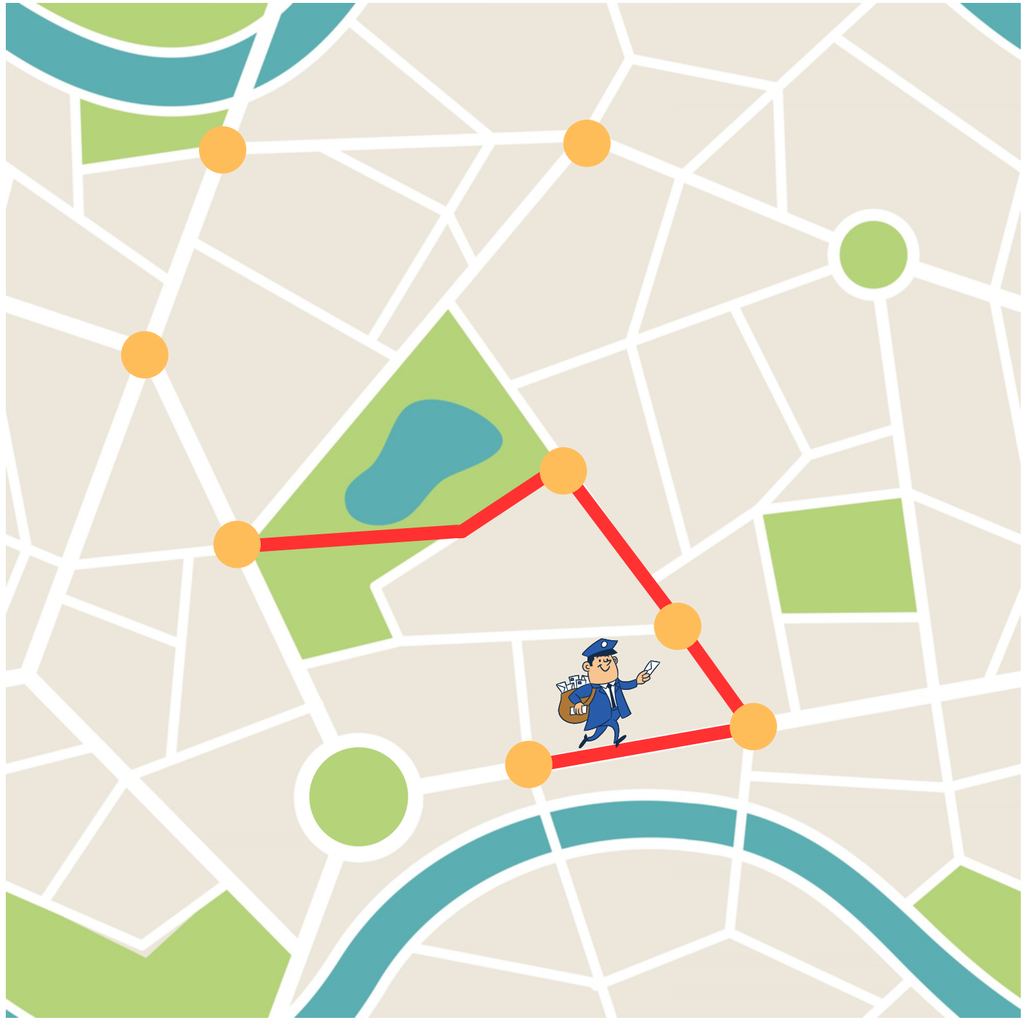
Introduction

Modélise le trajet d'un facteur distribuant le courrier

BUT Trouver le chemin le plus court

Ville == Graphe
Sommets == Intersections
Arêtes == Rues

Graphe eulérien :



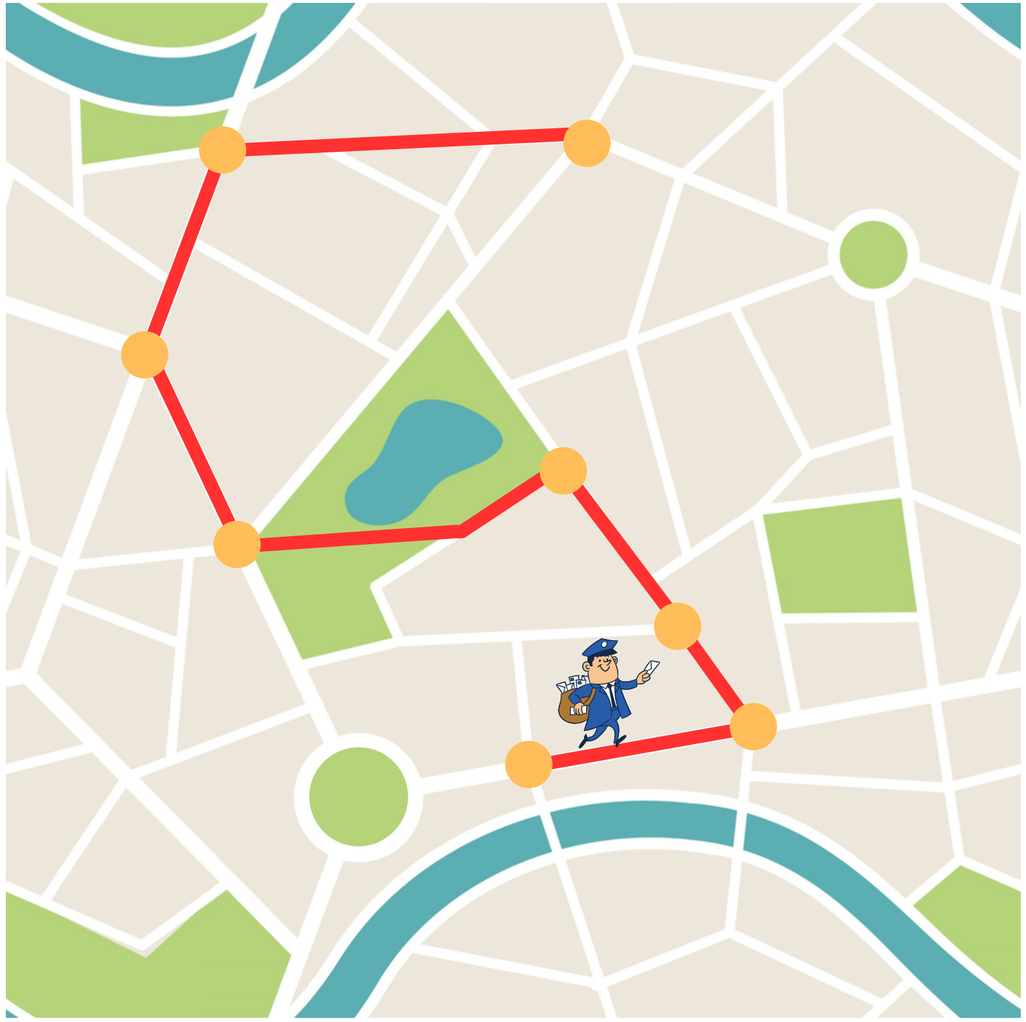
Introduction

Modélise le trajet d'un facteur distribuant le courrier

BUT Trouver le chemin le plus court

Ville == Graphe
Sommets == Intersections
Arêtes == Rues

Graphe eulérien :
    Connexe



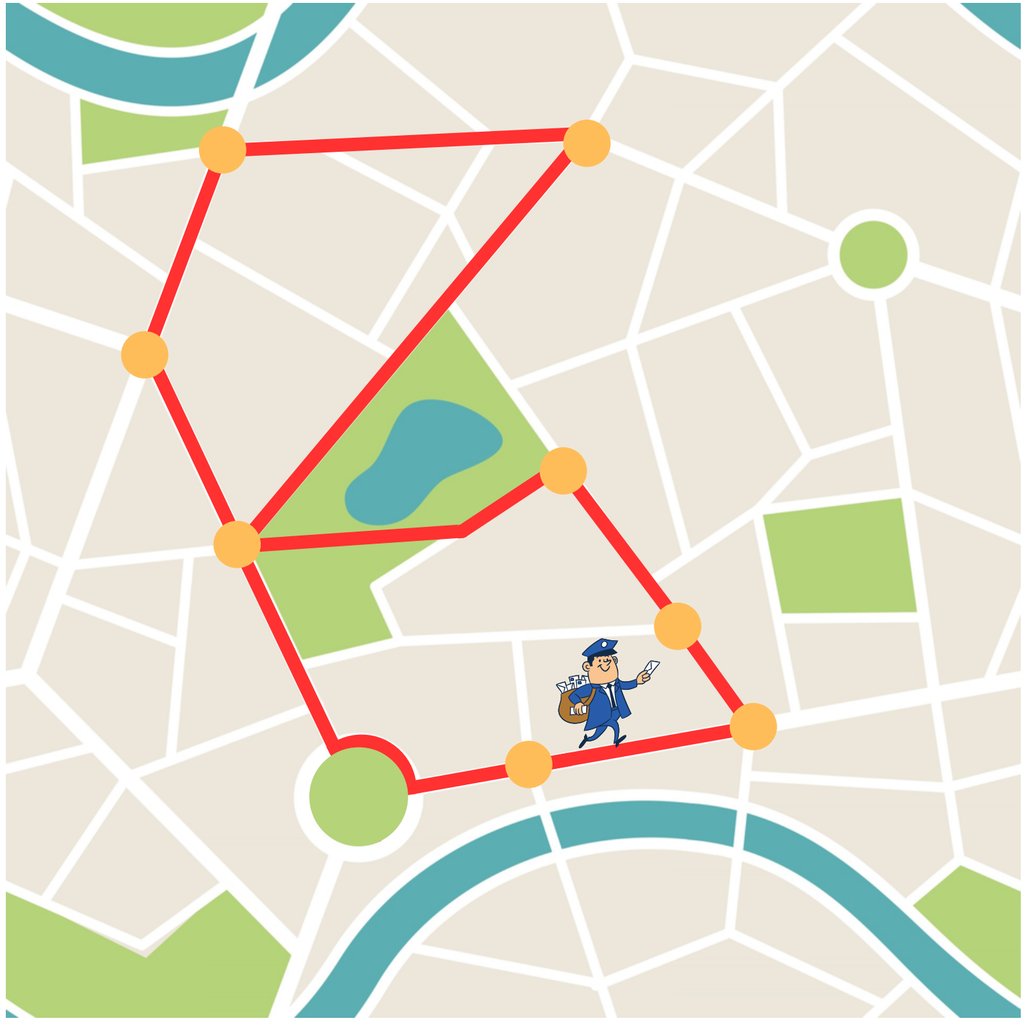
Introduction

Modélise le trajet d'un facteur distribuant le courrier

BUT Trouver le chemin le plus court

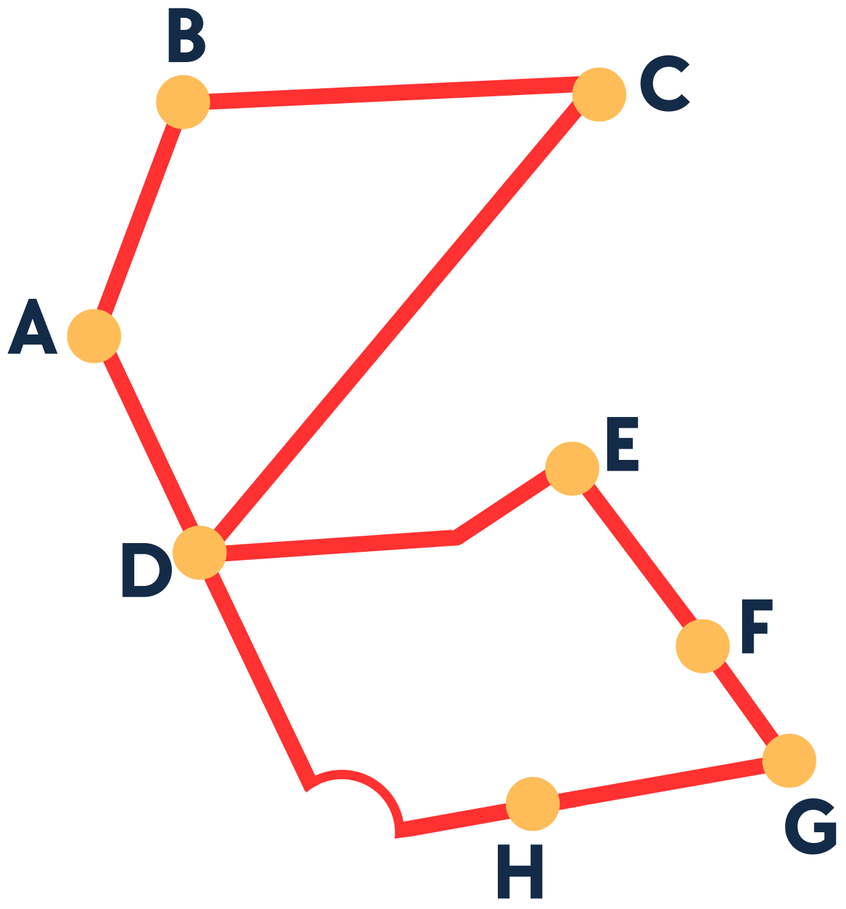
Ville == Graphe
Sommets == Intersections
Arêtes == Rues

Graphe eulérien :
- Connexe
- Sommet de degré pair



Description des Algorithmes

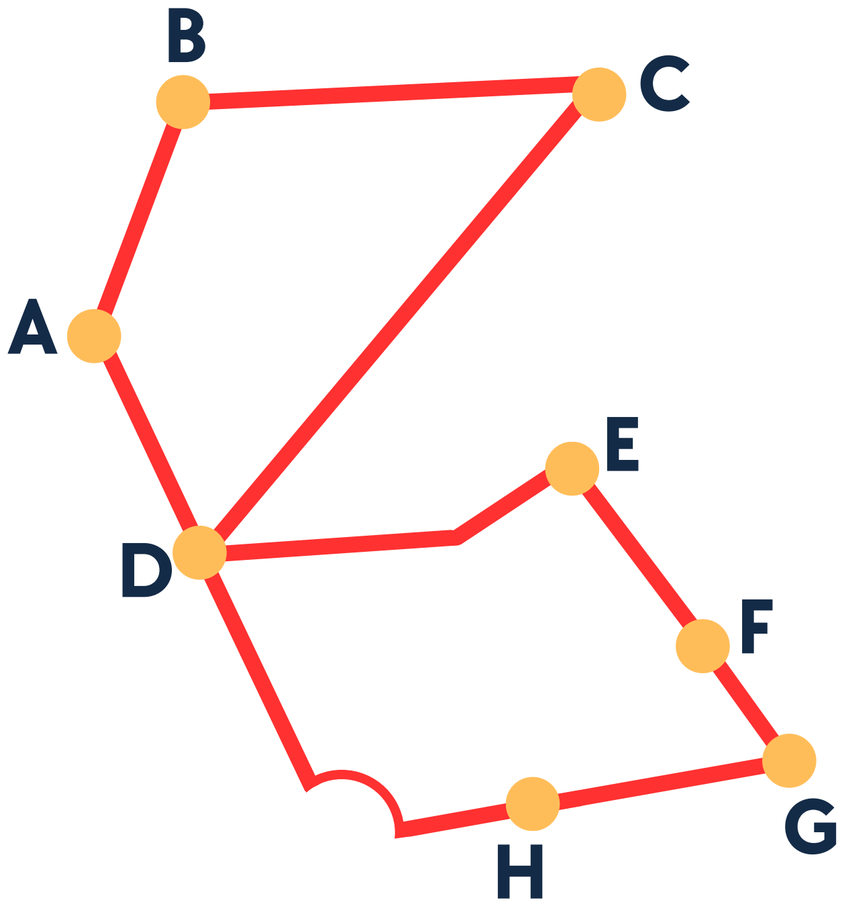
Approche générale



→ Travail sur un graphe eulérien

Description des Algorithmes

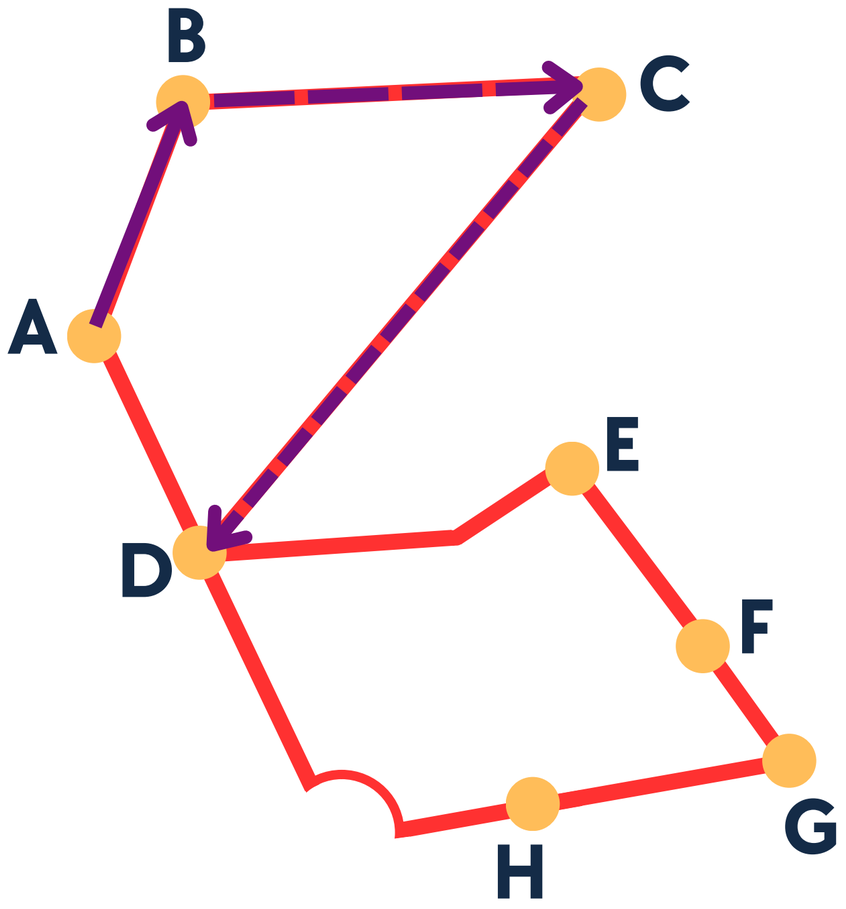
Approche générale



- Travail sur un graphe eulérien
- Si non(le graphe est eulérien) :
  - Rendre le graphe eulérien

Description des Algorithmes

Approche générale



- Travail sur un graphe eulérien
  - Si non(le graphe est eulérien) :
    - Rendre le graphe eulérien
- Effectuer un cycle eulérien

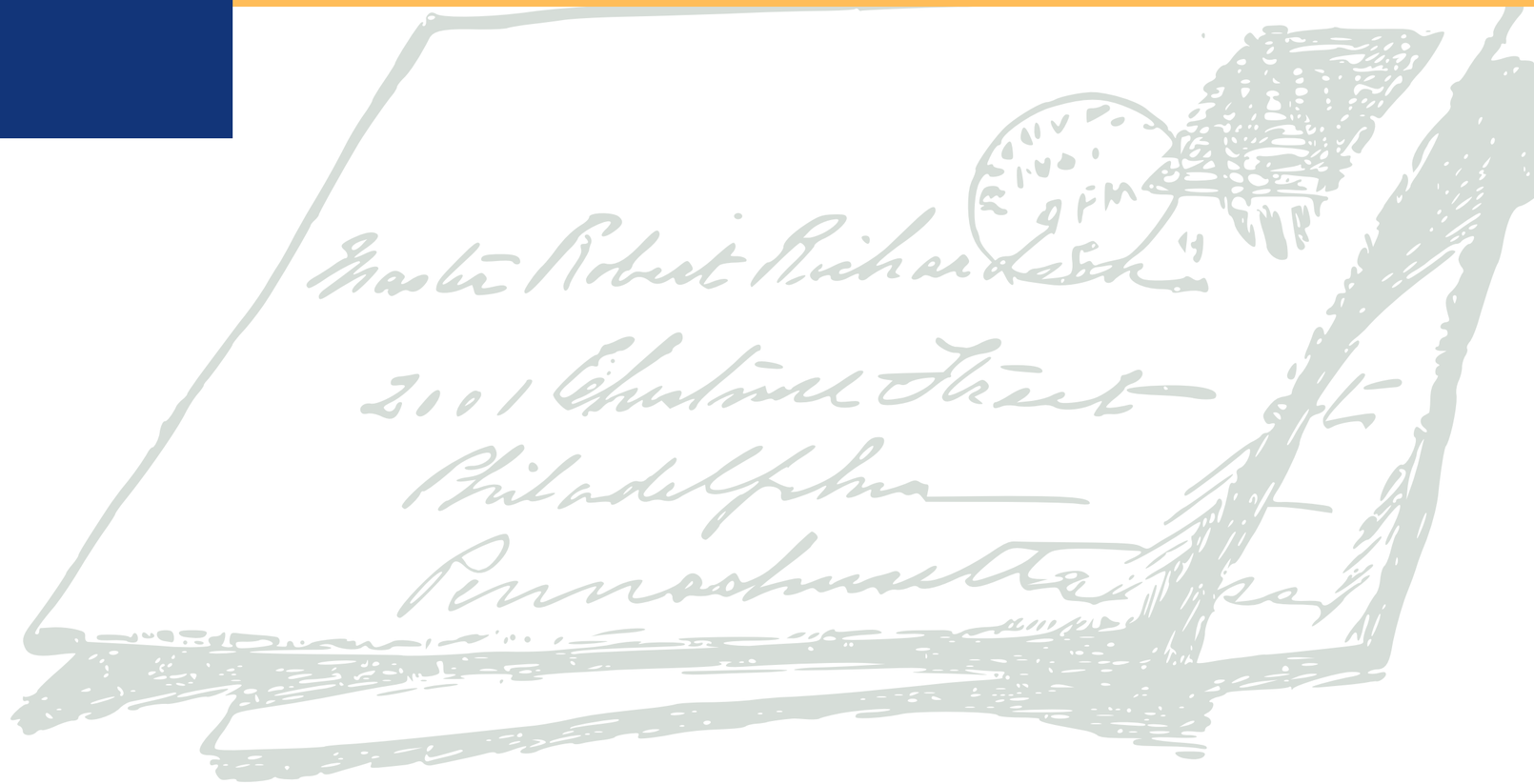
["A", "B", "C", "D", "E", "F", "G", "H", "A"]

Description des Algorithmes

Base commune

`algorithme_pc1/_pc2()`

Retourne le chemin que le postier devra emprunter



Description des Algorithmes

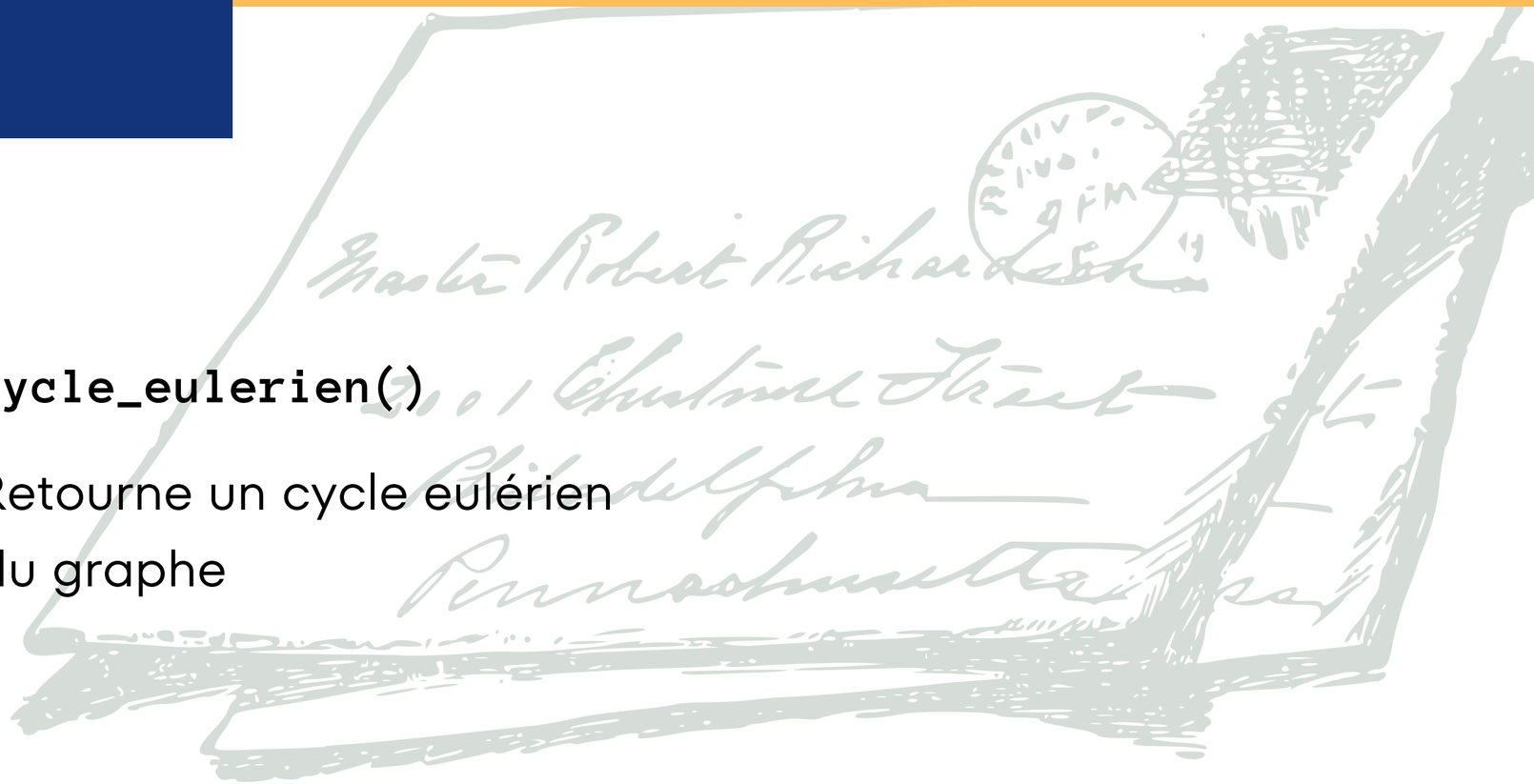
Base commune

`algorithme_pc1/_pc2()`

Retourne le chemin que le postier devra emprunter

`cycle_eulerien()`

Retourne un cycle eulérien du graphe



Description des Algorithmes

Base commune

`algorithme_pc1/_pc2()`

Retourne le chemin que le postier devra emprunter

`cycle_eulerien()`

Retourne un cycle eulérien du graphe

`sommet_suivant_cycle_eulerien()`

Retourne le sommet devant être le prochain dans la création du cycle

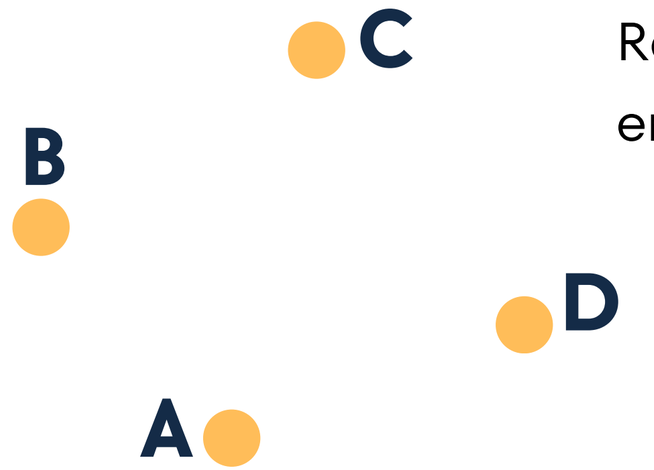
Description des Algorithmes

Algorithme 1

`transforme_toEulerien()`

Transforme un graphe en un graphe eulérien

Cas des sommets isolés :



Relit ces derniers entre eux

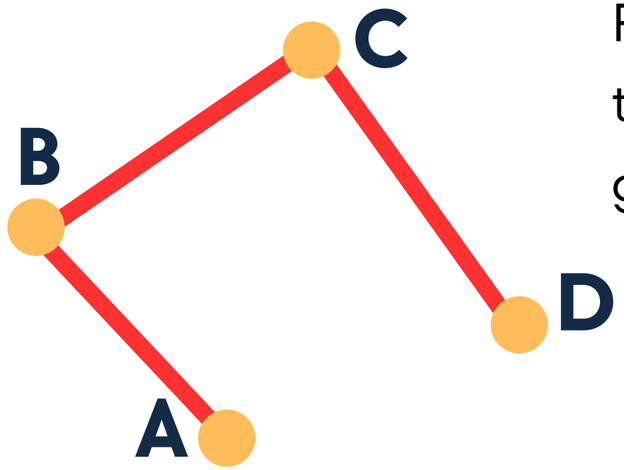
Description des Algorithmes

Algorithme 1

`transforme_toEulerien()`

Transforme un graphe en un graphe eulérien

Cas des sommets isolés :



Relit le dernier trouvé au reste du graphe

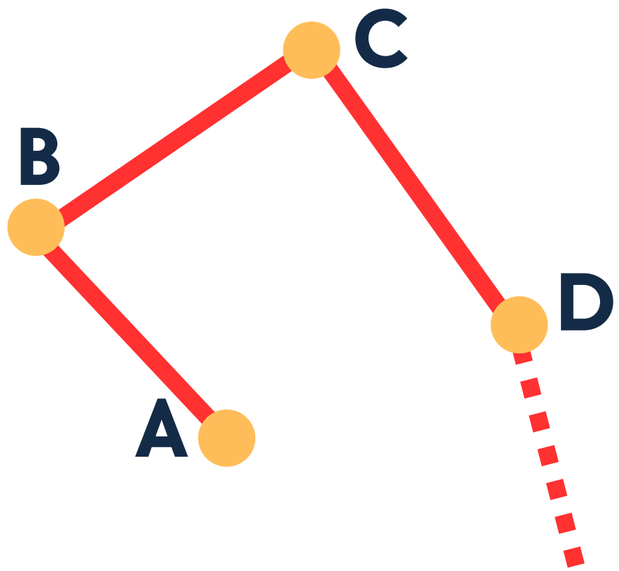
Description des Algorithmes

Algorithme 1

`transforme_toEulerien()`

Transforme un graphe en un graphe eulérien

Cas des sommets isolés :



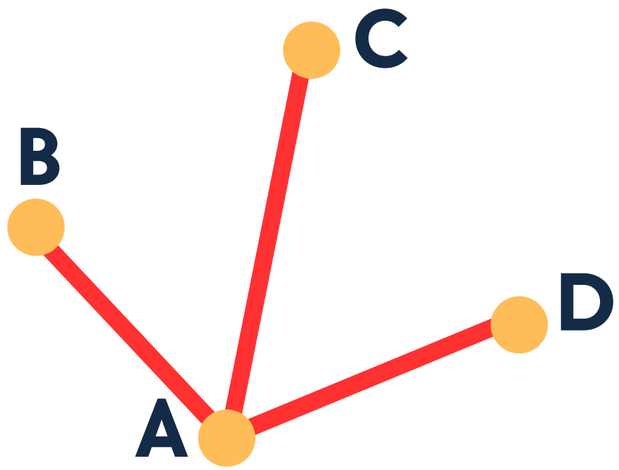
Description des Algorithmes

Algorithme 1

`transforme_toEulerien()`

Transforme un graphe en un graphe eulérien

Cas des sommets de degré impair :



Relit les sommets impairs entre eux
Sans créer créer de double

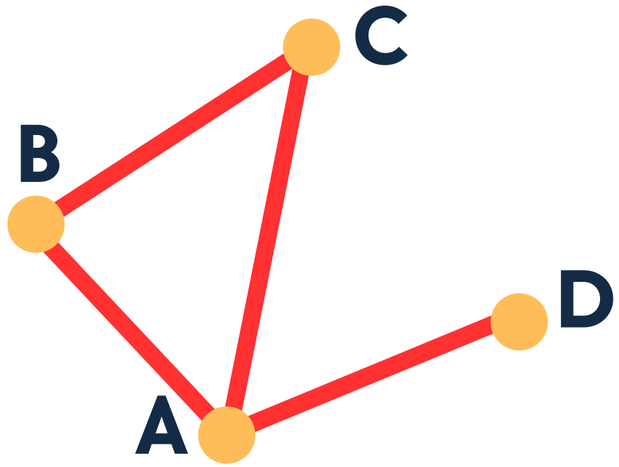
Description des Algorithmes

Algorithme 1

`transforme_toEulerien()`

Transforme un graphe en un graphe eulérien

Cas des sommets de degré impair :



Impossible de relier les sommets.
Suppression d'une arêtes

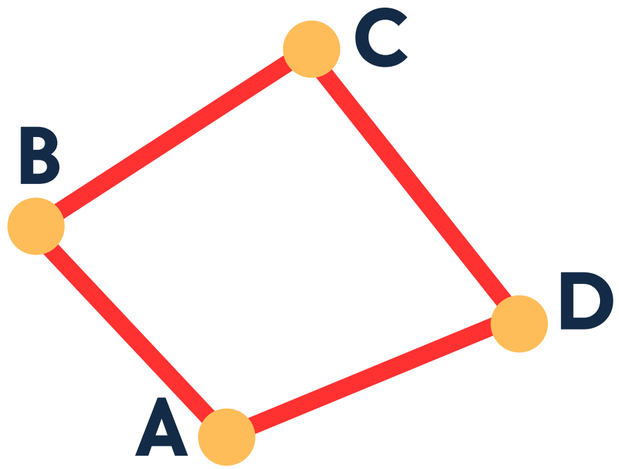
Description des Algorithmes

Algorithme 1

`transforme_toEulerien()`

Transforme un graphe en un graphe eulérien

Cas des sommets de degré impair :



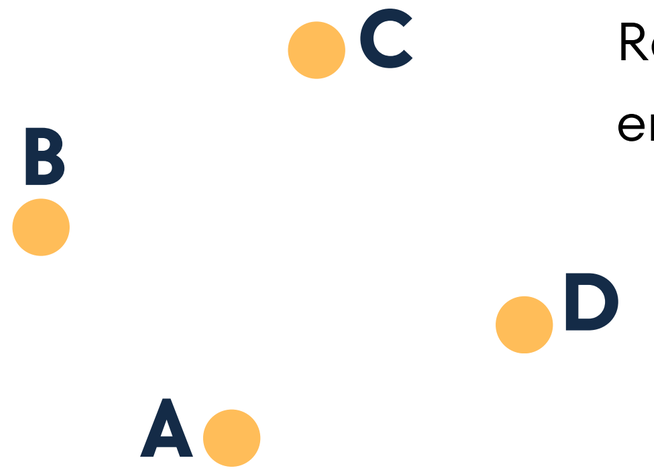
Description des Algorithmes

Algorithme 2

`transforme_toEulerien()`

Transforme un graphe en un graphe eulérien

Cas des sommets isolés :



Relit ces derniers entre eux

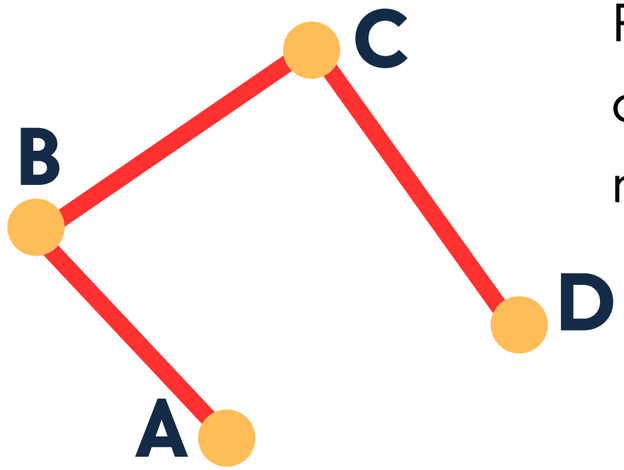
Description des Algorithmes

Algorithme 2

`transforme_toEulerien()`

Transforme un graphe en un graphe eulérien

Cas des sommets isolés :



Relit le premier et dernier trouvé au reste du graphe

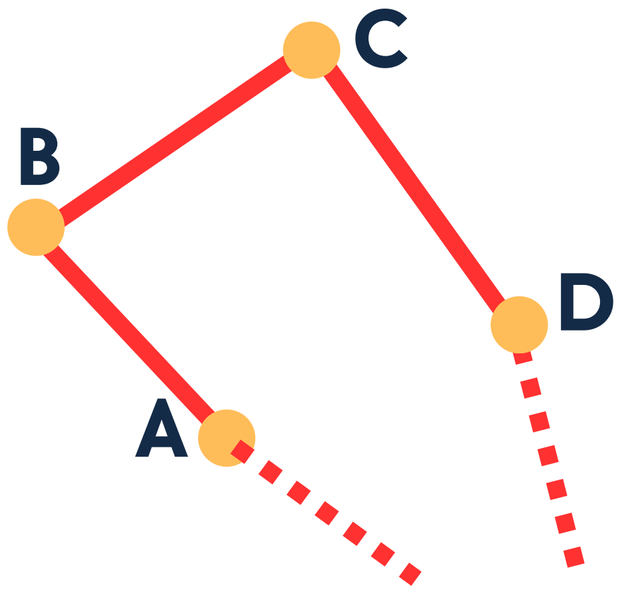
Description des Algorithmes

Algorithme 2

`transforme_toEulerien()`

Transforme un graphe en un graphe eulérien

Cas des sommets isolés :



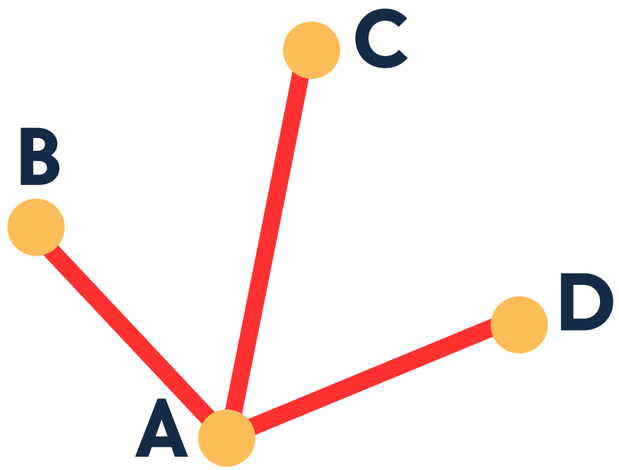
Description des Algorithmes

Algorithme 2

`transforme_toEulerien()`

Transforme un graphe en un graphe eulérien

Cas des sommets de degré impair :



Relit les sommets impairs entre eux

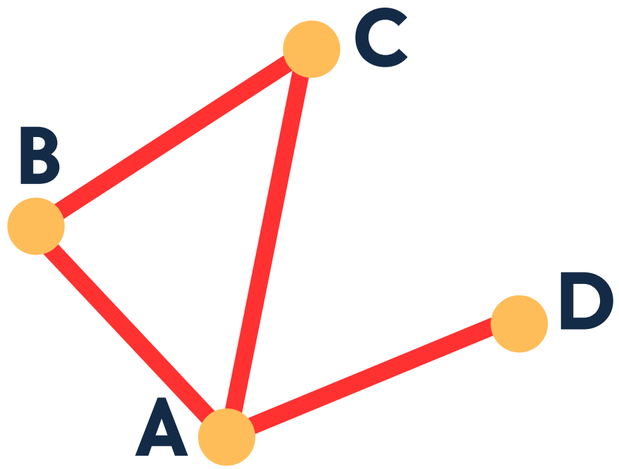
Description des Algorithmes

Algorithme 2

`transforme_toEulerien()`

Transforme un graphe en un graphe eulérien

Cas des sommets de degré impair :



Relit les sommets restant entre eux, même si l'arête existe déjà

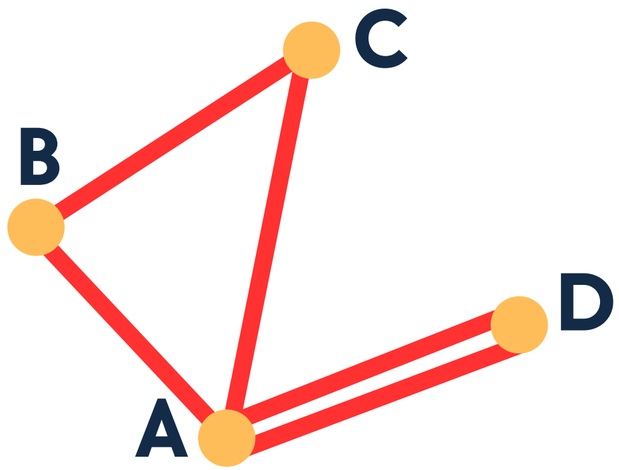
Description des Algorithmes

Algorithme 2

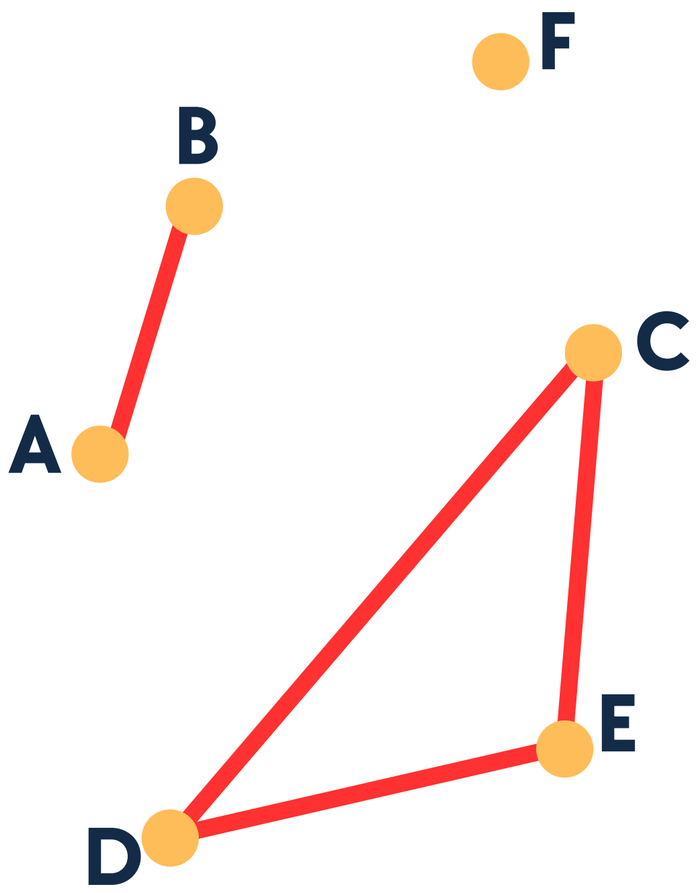
`transforme_toEulerien()`

Transforme un graphe en un graphe eulérien

Cas des sommets de degré impair :



Comparaison des Algorithmes

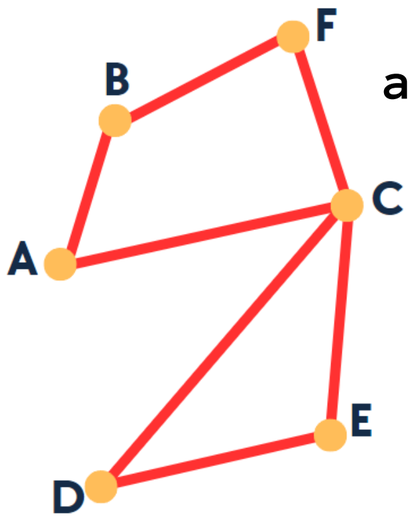


graphe initial

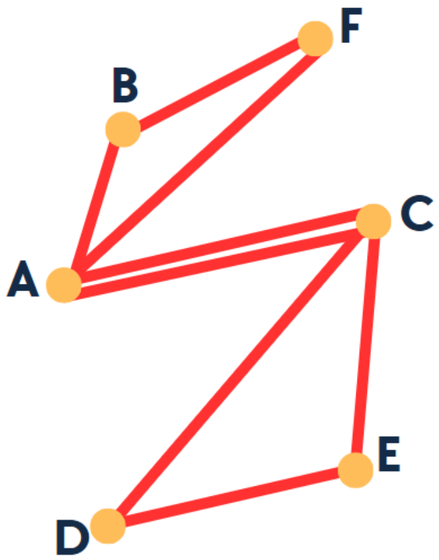
LePostierChinois1.py

LePostierChinois2.py

Comparaison des Algorithmes



algorithme_pc1

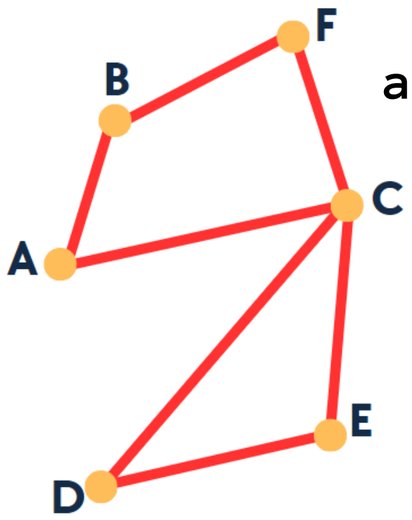


algorithme_pc2

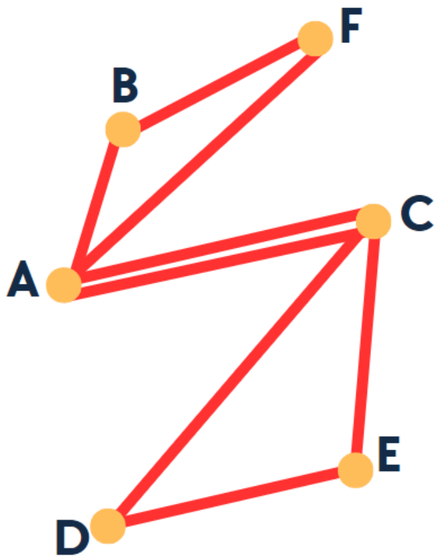
LePostierChinois1.py

LePostierChinois2.py

Comparaison des Algorithmes



`algorithme_pc1`



`algorithme_pc2`

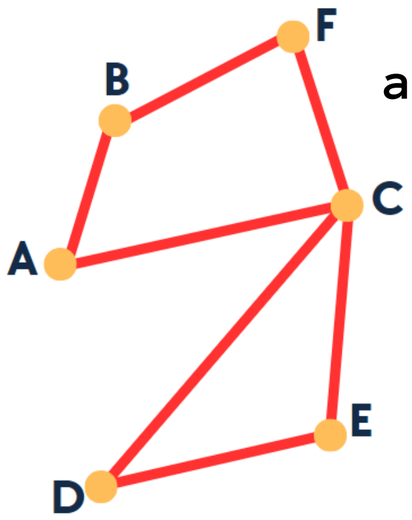
`LePostierChinois1.py`

Création de **3 arêtes** ; 0 suppression

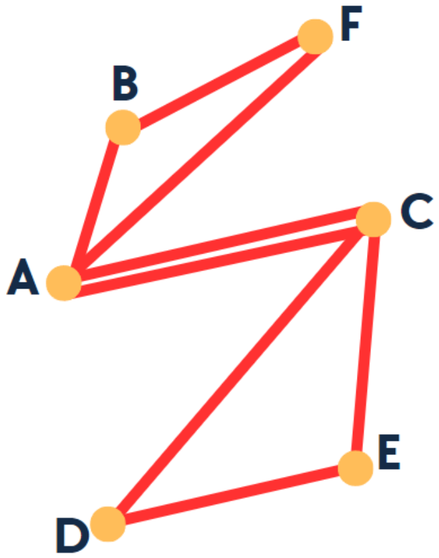
`LePostierChinois2.py`

Création de **4 arêtes** ; 1 arêtes doubles

Comparaison des Algorithmes



`algorithme_pc1`



`algorithme_pc2`

`LePostierChinois1.py`

Création de **3 arêtes** ; 0 suppression

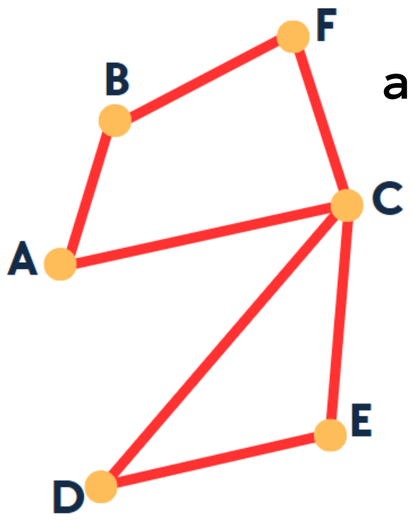
Temps d'exécution : **0.456** secondes

`LePostierChinois2.py`

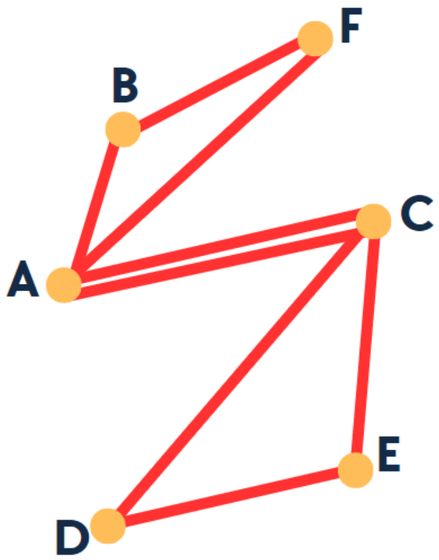
Création de **4 arêtes** ; 1 arêtes doubles

Temps d'exécution : **0.503** secondes

Comparaison des Algorithmes



algorithme_pc1



algorithme_pc2

LePostierChinois1.py

Création de **3 arêtes** ; 0 suppression

Temps d'exécution : **0.456** secondes

["A", "B", "F", "C", "E", "D", "C", "A"]

→ **8** étapes

LePostierChinois2.py

Création de **4 arêtes** ; 1 arêtes doubles

Temps d'exécution : **0.503** secondes

["A", "B", "F", "A", "C", "D", "E", "C", "A"]

→ **9** étapes

Conclusion

Solution Alternative

Construire un chemin sans modifier le graphe

→ Plus de réalisme

→ Repasse donc plus d'une fois dans la même arête (rue)

Ne modifie pas la ville

["C", "F", "C", "E", "D", "C", "B", "A", "B", "C"]

→ Chemin beaucoup plus long