ABC022解説
A:Best Body

A-問題概要

- N日間の高橋君の体重の情報が与えられる
  - 1日目の体重はWキログラム
  - i日目の体重はi-1日目の体重 + A[i] キログラム
- 体重がSキログラム以上Tキログラム以下ならばBestBody
- 高橋君がBestBodyであった日数を求めよ

A-解法

- N日分全ての体重を求めて、いちいちBestBodyか確かめる

- P[i] = i日目の高橋君の体重
  - P[1] = W
  - P[i] = P[i-1] + A[i]

- 入力を受け取りながらPを埋めていけばよい

ABC022解説
B:Bumble Bee

B-問題概要

- 高橋君は異なるN個の花に訪れる
- i番目に訪れた花の種類はA[i]
- i>jかつA[i]=A[j] となるjが存在すればi番目の花は受粉する
- 受粉する花の個数を求めよ

- $1 \leqq N \leqq 100{,}000$

B-考察

- i>jかつA[i]=A[j] となるjが存在すればi番目の花は受粉する
- この条件を考察してみる

B-考察

- i>jかつA[i]=A[j] となるjが存在すればi番目の花は受粉する
- 例：
  - 1, 2, 3, 1, 2, 3, 1, 2, 3, 2, 1

B-考察

- i>jかつA[i]=A[j] となるjが存在すればi番目の花は受粉する
- 例：
  - 1, 2, 3, 1, 2, 3, 1, 2, 3, 2, 1
    ※赤字が受粉する花

B-考察

- i>jかつA[i]=A[j] となるjが存在すればi番目の花は受粉する
- 例：
  - 1, 2, 3, 1, 2, 3, 1, 2, 3, 2, 1
  - 1, , , 1, , , 1, , , , 1
  - , 2, , , 2, , , 2, , 2,
  - , , 3, , , 3, , , 3, ,

  ※種類別に分けてみた

B-考察

- i>jかつA[i]=A[j] となるjが存在すればi番目の花は受粉する
- 例：
  - 1, 2, 3, 1, 2, 3, 1, 2, 3, 2, 1
  - 1, , , 1, , , 1, , , , 1
  - , 2, , , 2, , , 2, , 2,
  - , , 3, , , 3, , , 3, ,
- どうやら同じ種類の中で最初に訪れるもの以外が受粉する

B-解法1

- 同じ種類の中で最初に訪れるもの以外が受粉する
- 毎回、同じ種類の中で何回目に訪れたかの情報を元に受粉するか判断すればよい
- cnt[i] = 種類iの花に今までに何回訪れたか
  - 0で初期化

B-解法1

- cnt[i] = 種類iの花に今までに何回訪れたか
  - 0で初期化

- 擬似コード

for i ← (1, 2, 3, … , N) :

  cnt[A_i] == 0ならば受粉しない

  それ以外ならば受粉する

  cnt[A_i] = cnt[A_i] + 1

B-解法1

- cnt[i] = 種類iの花に今までに何回訪れたか
  - 0で初期化

- 擬似コード

  for i ← (1, 2, 3, ⋯ , N) :

  cnt[A_i] == 0ならば受粉しない

  それ以外ならば受粉する

  cnt[A_i] = cnt[A_i] + 1
- 計算量はO(N)

B-解法2

- 同じ種類の中で最初に訪れるもの以外が受粉する
- 逆に言うと同じ種類の中で最初に訪れるものだけが受粉しない
  - これは花の全種類数と一致
- N – (花の全種類数) が答えになる

B-解法2

- 花の種類数を数えれば良い
  - 各種類について出現回数を求めて、1回以上出現する花が何種類あるか数える
  - STL:setなど集合のデータ構造を使う
- O(N)もしくはO(NlogN)

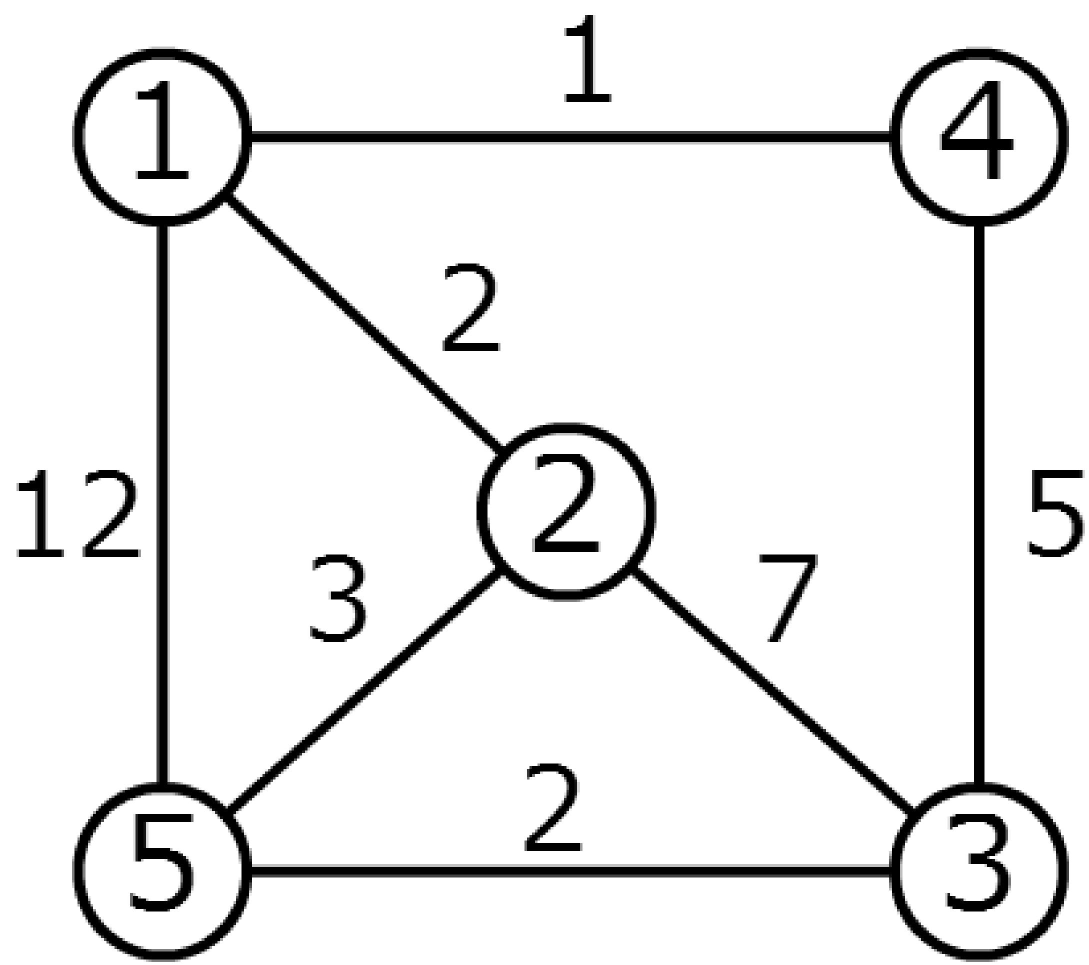
ABC022解説
C:Blue Bird

C-問題概要

- 辺に重み（長さ）のある無向グラフが与えられる
  - 頂点数N ≦ 300, 辺数 M ≦ N(N-1)/2
- 頂点1から開始して、頂点1に戻ってくる、同じ辺を二度通らない経路があるか求めよ。あるならばその中で最短なものの長さを求めよ。

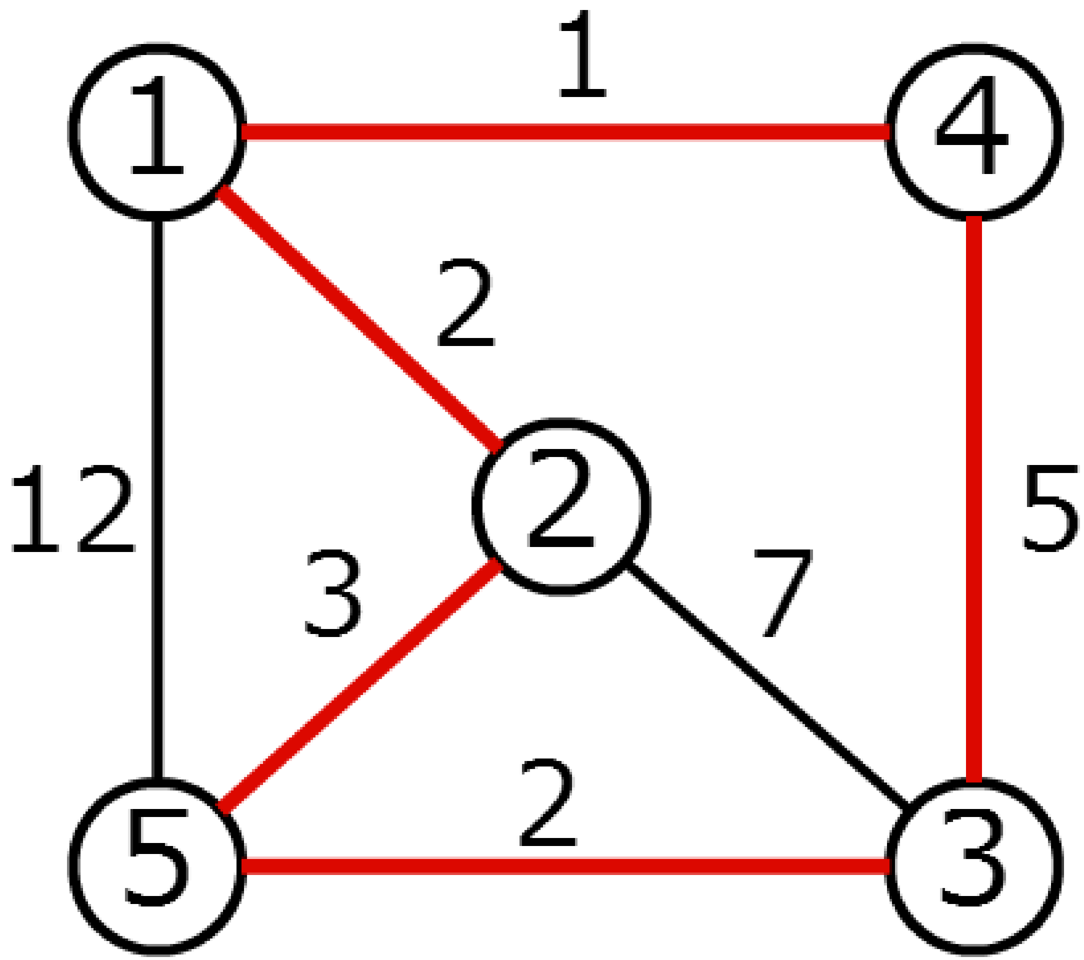
C-例

- 入出力1



C-例

- 入出力1：正解の経路

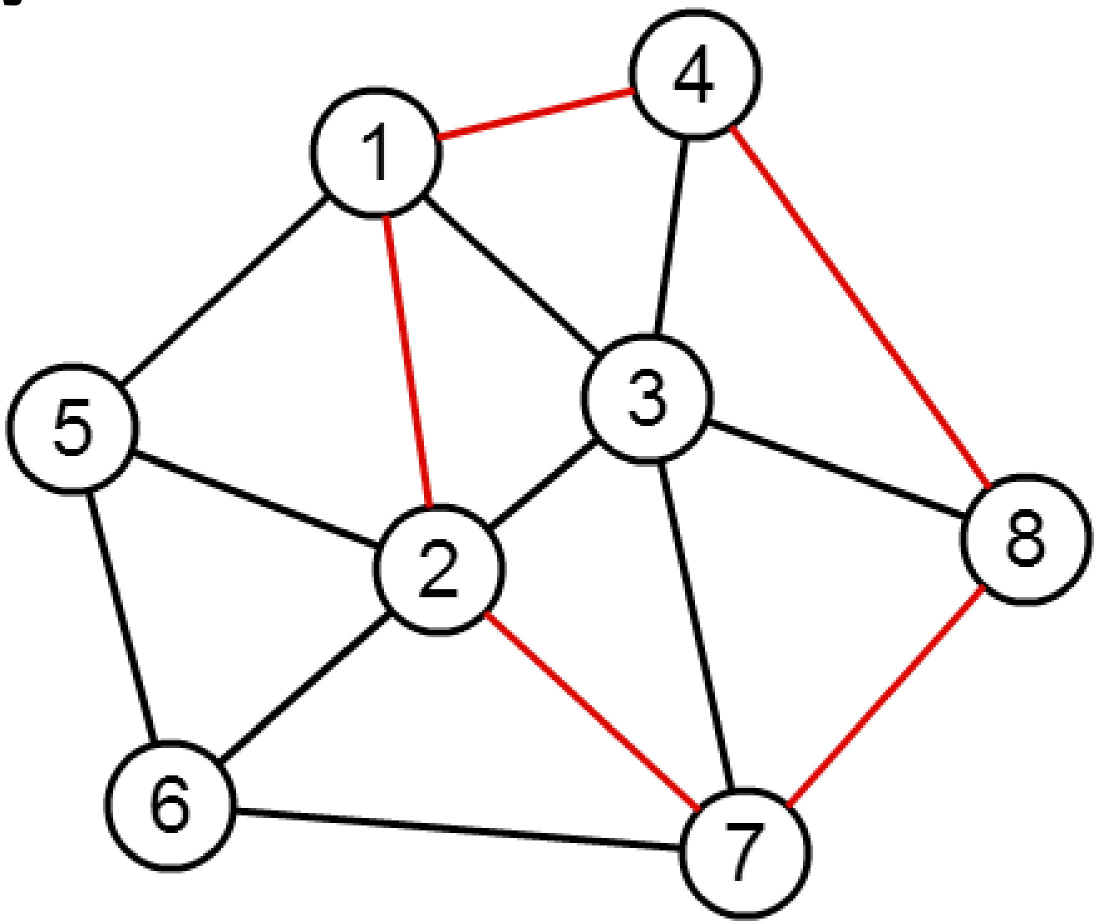


C-考察

- 見た目は最短経路問題に似ている
  - 今求めたいのは最短「閉路」なのでそのまま Dijkstraなどを使うことはできない
- 最短閉路も部分的には最短経路に似た性質を持っている

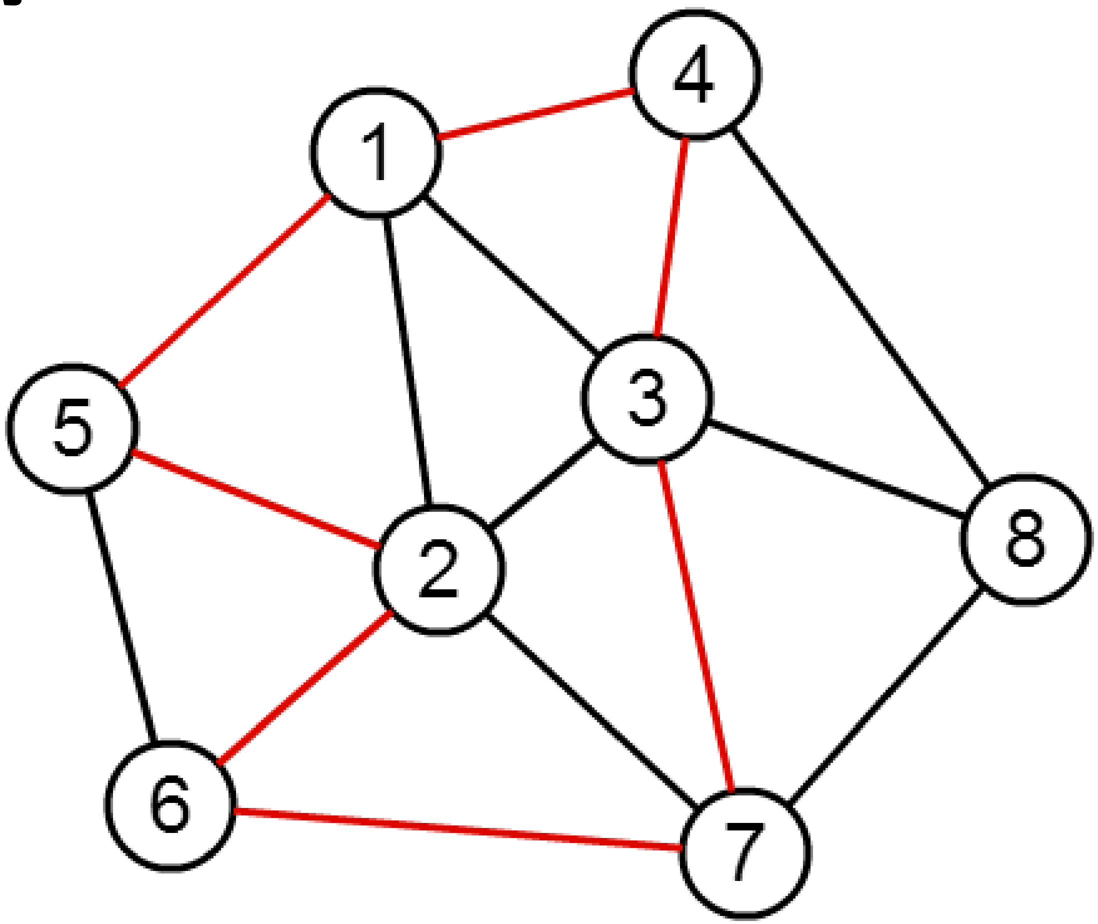
C-考察

- 閉路の例



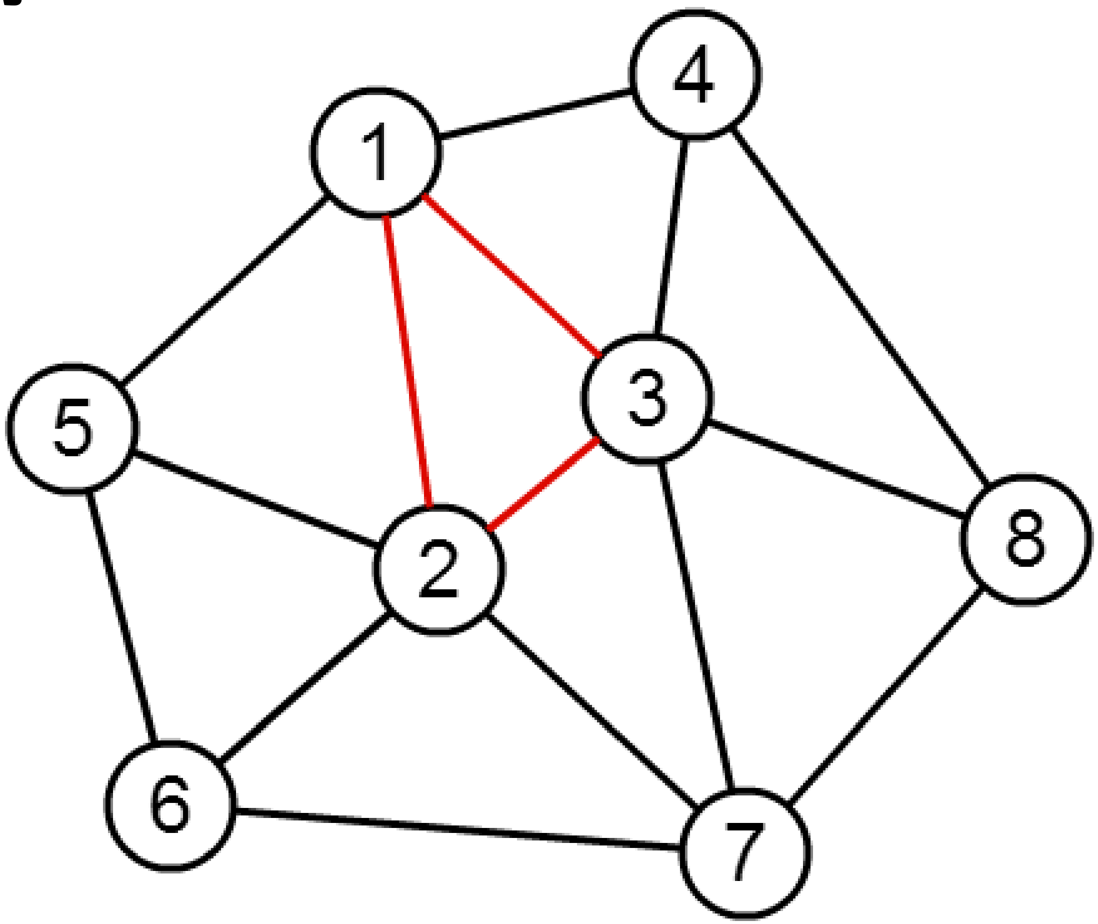
C-考察

- 閉路の例



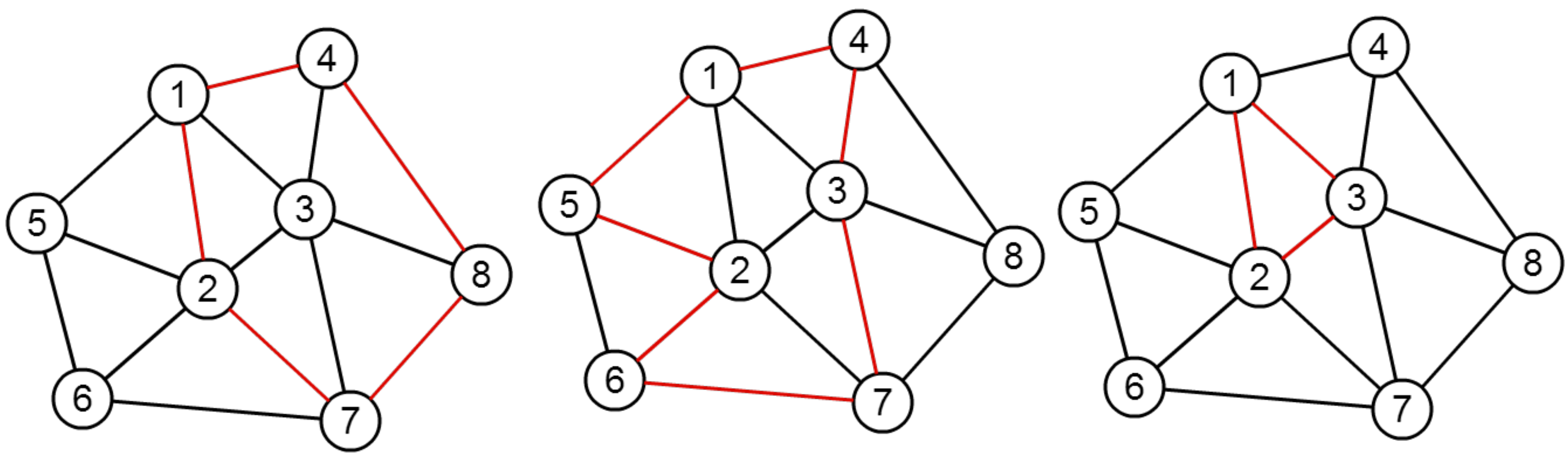
C-考察

- 閉路の例



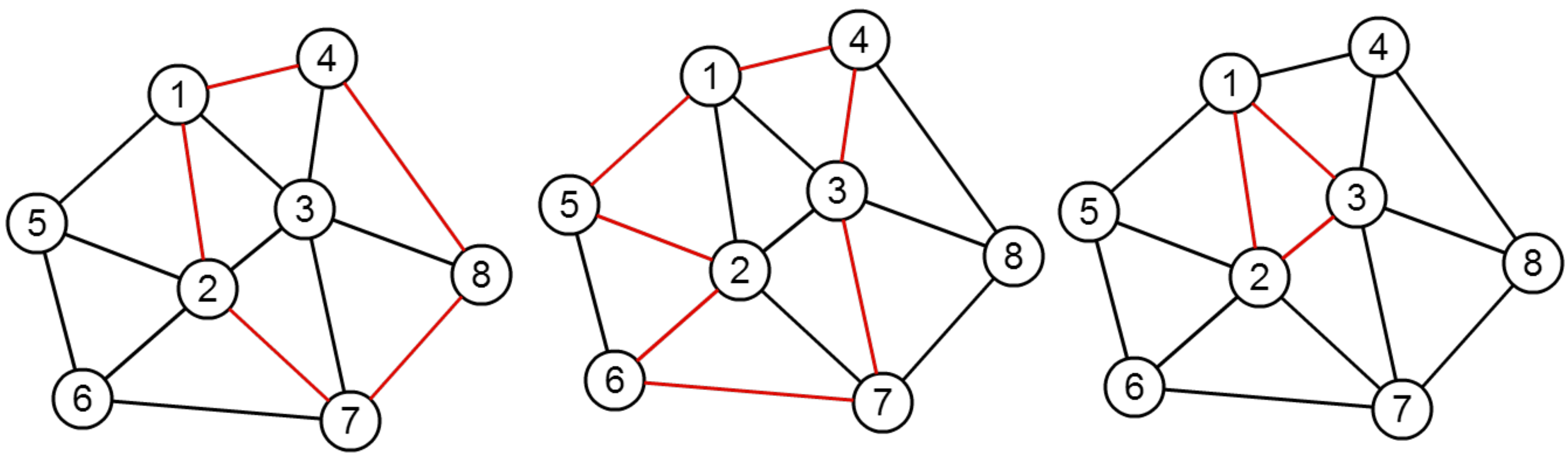
C-考察

- 頂点1に隣接する頂点はちょうど2つ含まれる



C-考察

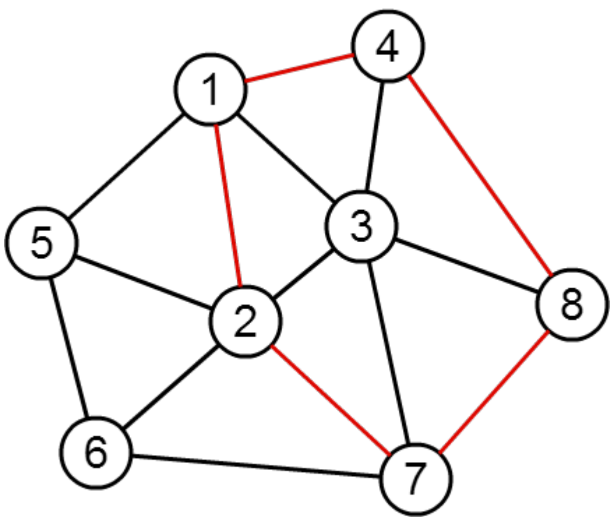
- 頂点1に隣接する頂点はちょうど2つ含まれる
  - 頂点1に隣接する頂点はO(N)個
  - どの2つを選ぶかはO(N^2)通り

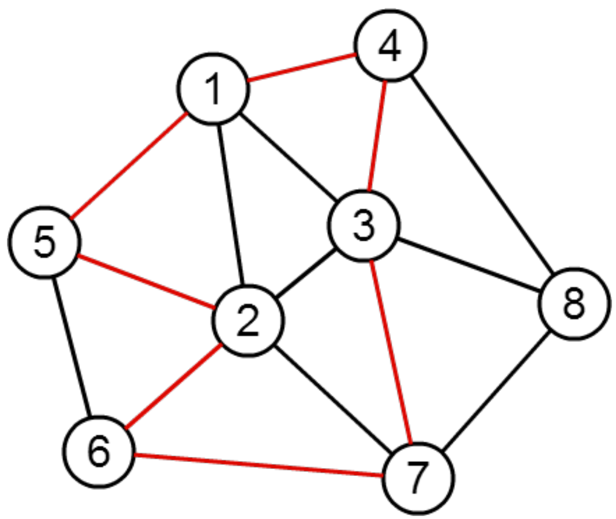


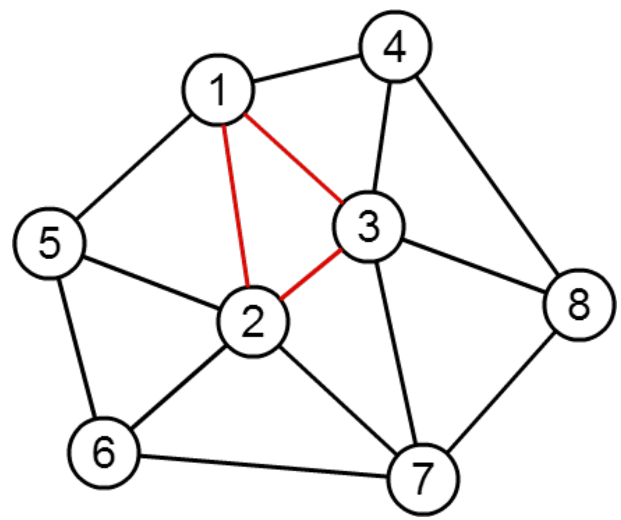
C-考察

- 頂点1に隣接する辺を除いて考えてみる

Before



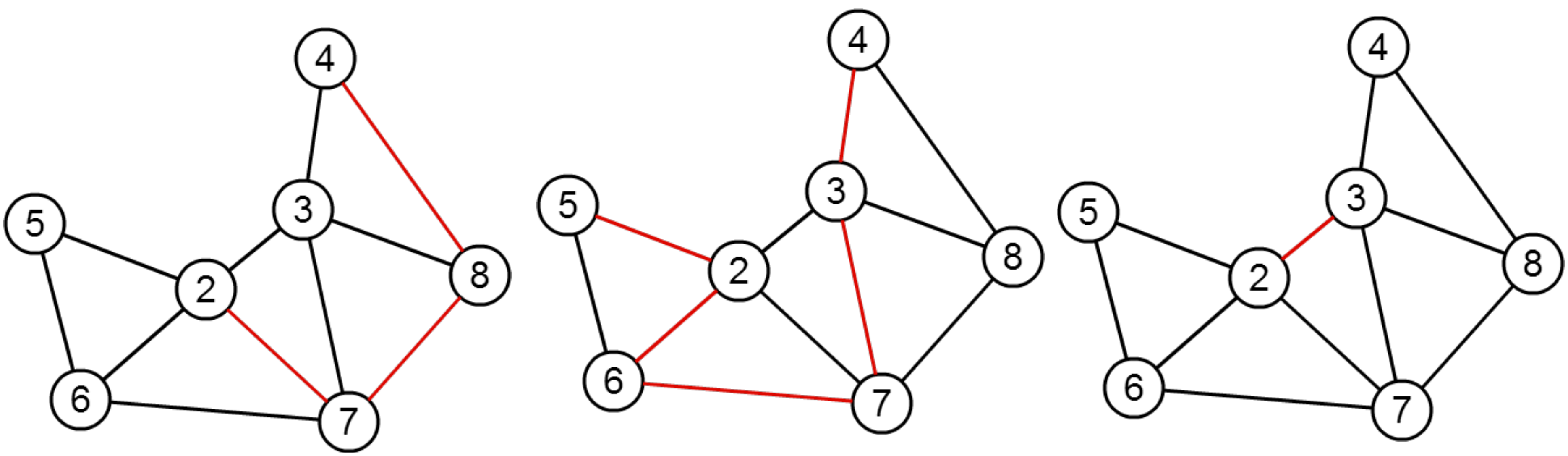




C-考察

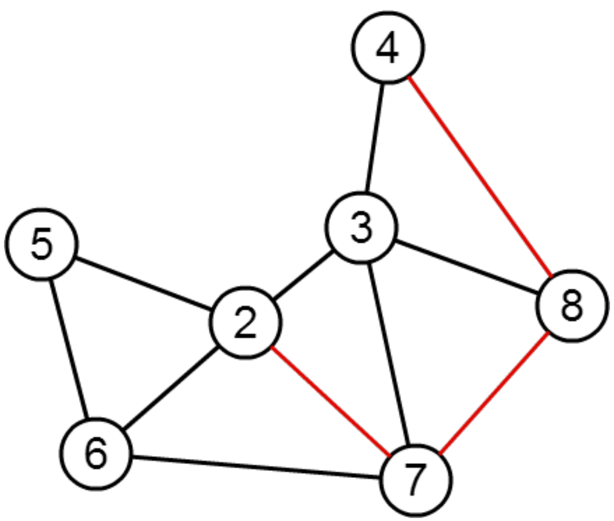
- 頂点1に隣接する辺を除いて考えてみる

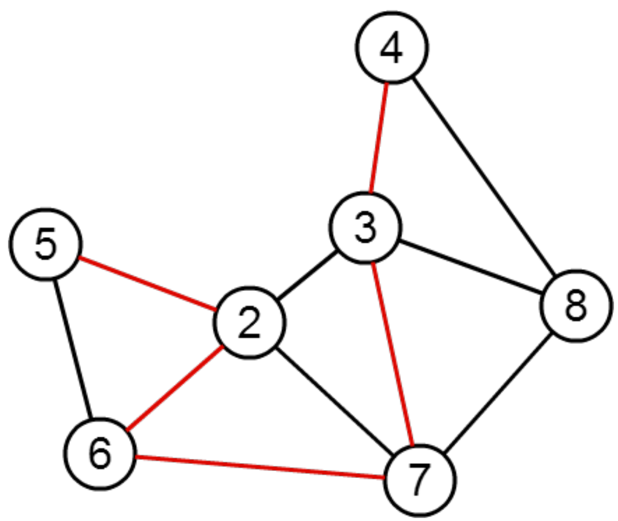
After

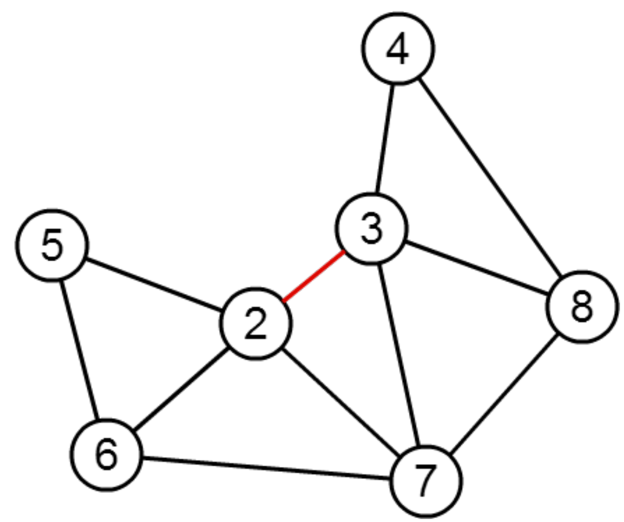


C-考察

- 頂点1に隣接する辺を除いて考えてみる
- 閉路が経路になった
  - 扱いやすい！







C-考察

- 頂点1に隣接する頂点のうち、どの2つを経路に含めるか決めれば、あとは最短経路問題になる
  - 頂点1に隣接する辺を除いたグラフでの最短経路問題

C-アルゴリズム

- 頂点1に隣接する頂点のうちどの2つを閉路に含めるか全通り試す
  - 頂点X,Yを選んだとする
- 頂点1に隣接する辺を除いたグラフでXとYの最短経路の長さを求める
  - Sとする
- 1とX,1とYを結ぶ辺の長さの和とSを足しあわせたものがX,Yを含む閉路の中で最短の閉路

C-アルゴリズム

- 頂点1に隣接する頂点のうちどの2つを閉路に含めるか全通り試す
  - 頂点X,Yを選んだとする
- 頂点1に隣接する辺を除いたグラフでXとYの最短経路の長さを求める
  - Sとする
- 1とX,1とYを結ぶ辺の長さの和とSを足しあわせたものがX,Yを含む閉路の中で最短の閉路

O(N^2)

O(？)

C-アルゴリズム

- 「頂点1に隣接した辺を除いたグラフ」という共通のグラフの様々な点対に対して最短経路を求めなければならない
  - 毎回計算するとDijkstraでもO(N^2)かかる
  - それをO(N^2)回繰り返すので合計でO(N^4)

C-アルゴリズム

- 「頂点1に隣接した辺を除いたグラフ」という共通のグラフの様々な点対に対して最短経路を求めなければならない
  - 毎回計算するとDijkstraでもO(N^2)かかる
  - それをO(N^2)回繰り返すので合計でO(N^4)
- あらかじめ全点対の最短経路を求めておけば良い
  - Warshall-Floydを使えばO(N^3)を1回ですむ

C-まとめ

- 「頂点1に隣接する辺を除いたグラフ」の全点対の最短経路を前計算しておく
- 頂点1に隣接する頂点のうちどの2つを閉路に含めるか全通り試す
- 前計算した最短経路の結果と合わせて最短閉路の長さを求める

C-まとめ

- 「頂点1に隣接する辺を除いたグラフ」の全点対の最短経路を前計算しておく
  O(N^3)
- 頂点1に隣接する頂点のうちどの2つを閉路に含めるか全通り試す
  O(N^2)
- 前計算した最短経路の結果と合わせて最短閉路の長さを求める

合計O(N^3)

C-おまけ(別解)

- 最短経路木を使った解法があります
- 「最短経路木に含まれる辺いくつか」と「最短経路木に含まれない辺をちょうど1つ」使うことで最短閉路が存在するならば必ず作ることができます
  - 証明などが難しいので詳しい内容は割愛
- ボトルネックは最短経路木の構築でO(N^2)もしくはO(ElogN) ※Eは辺の数
- 今回は制約がN≦300なのでこれを使わなくても良い

ABC022解説
D:Big Bang

D-問題概要

- 座標平面上の要素数Nの点集合が2つ与えられる（AとBとする）
- BはAに対して
  - 平行移動
  - 回転
  - P倍に相似拡大

  したものだということがわかっている
- Pを求めよ

D-考察

- 相似な2つの図形の相似比を求める問題
- どの点がどの点に対応するのかがわかれば簡単な除算の問題になる
  - 残念ながら対応がわからない
  - 平行移動や回転のせいで求めるのも難しそう
- どの点がどの点に対応するのか具体的に求めなくてもいい方法はないだろうか？

D-考察

- Aにおける「あらゆる長さ」がBではP倍になっている
  - 「あらゆる長さ」の中で簡単に求められるものは無いだろうか
  - どんな長さを求めるかによって様々な解法が考えられる

D-解法1

- 全点対距離の総和
  - 長さのみで決まるユニークな値
    - 平行移動や回転の影響を受けない
  - 愚直に全部計算しましょう
    - O(N^2)：部分点獲得

D-解法2

- 最近点対の長さ
  - 最も距離が近い2点の長さ
    - 平行移動や回転によって変わることはない

  - 自明な解法は全点対の長さを調べる方法
    - O(N^2)：部分点獲得
  - 分割統治や特殊なデータ構造によって速く解けることが知られている
    - O(NlogN)：満点獲得

D-解法3

- 最遠点対の長さ
  - 最も距離が遠い2点の長さ
    - 平行移動や回転によって変わることはない
  - 自明な解法は全点対の長さを調べる方法
    - O(N^2)：部分点獲得
  - 凸包を利用した速い解法がある
    - O(NlogN)：満点獲得

D-解法4

- 凸包の外周の長さ
  - 全ての頂点を内側に含む最小の凸多角形の外周
    - 平行移動や回転の影響を受けない
  - 凸包は様々な高速なアルゴリズムがある
    - O(NlogN)等：満点獲得

D-解法5

- 重心からの最遠点までの距離
  - N頂点の重心から最も遠い所にある頂点までの距離
    - 平行移動や回転の影響を受けない
  - 全点対調べなくて良いので楽
    - O(N)等：満点獲得

D-解法6

- 重心からの最近点までの距離
  - 値が0になることがあり、うまくPを求めることができないので、場合分けが必要

    最遠点よりは複雑だが、なお楽である
    - O(N)等：満点獲得

D-まとめ

- 全点対の長さの総和
- 最近点対の長さ
- 最遠点対の長さ
- 凸包の外周の長さ
- 重心からの最遠点までの長さ
- 重心からの最近点までの長さ

D-まとめ

- 全点対の長さの総和　O(N^2)
- 最近点対の長さ　O(NlogN)
- 最遠点対の長さ　O(NlogN)
- 凸包の外周の長さ　O(NlogN)
- 重心からの最遠点までの長さ　O(N)
- 重心からの最近点までの長さ　O(N)

他にもいろいろ有ります。実装しやすいものを選びましょう。