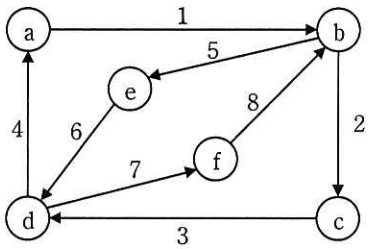
問３　一筆書きに関する次の記述を読んで，設問 1～4 に答えよ。

グラフは，有限個の点の集合と，その中の 2 点を結ぶ辺の集合から成る数理モデルである。グラフの点と点の間をつなぐ辺の列のことを経路という。本問では，任意の 2 点間で，辺をたどることで互いに行き来することができる経路が存在する（以下，強連結という）有向グラフを扱う。強連結な有向グラフの例を図 1 に示す。辺は始点と終点の組で定義する。各辺には 1 から始まる番号が付けられている。



辺1＝(点a，点b)　辺2＝(点b，点c)
辺3＝(点c，点d)　辺4＝(点d，点a)
辺5＝(点b，点e)　辺6＝(点e，点d)
辺7＝(点d，点f)　辺8＝(点f，点b)

凡例　辺＝(辺の始点，辺の終点)

図 1　強連結な有向グラフの例

〔一筆書き〕

本問では，グラフの全ての辺を 1 回だけ通り，出発点から出て出発点に戻る閉じた経路をもつグラフを，一筆書きができるグラフとする。

〔一筆書きの経路の求め方〕

一筆書きの経路を求めるためには，出発点から辺の向きに従って辺を順番にたどり，出発点に戻る経路を見つける探索を行う。たどった経路（以下，探索済の経路という）について，グラフ全体で通過していない辺（以下，未探索の辺という）がない場合は，この経路が一筆書きの経路となる。未探索の辺が残っている場合は，探索済の経路を，未探索の辺が接続する点まで遡り，その点を出発点として，同じ点に戻る経路を見つけて，遡る前までの経路に連結することを繰り返す。

各点を始点とする辺を接続辺という。グラフの各点に対して接続辺の集合が決まり，辺の番号が一番小さい接続辺を最初の接続辺という。同じ始点をもつ接続辺の集合で，辺の番号を小さいものから順番に並べたときに，辺の番号が次に大きい接続辺を次の接続辺ということにする。

図 1 のグラフの各点の接続辺の集合を表 1 に示す。図 1 において，点 b の最初の接

続辺は辺２である。辺２の次の接続辺は辺５となる。辺５の次の接続辺はない。

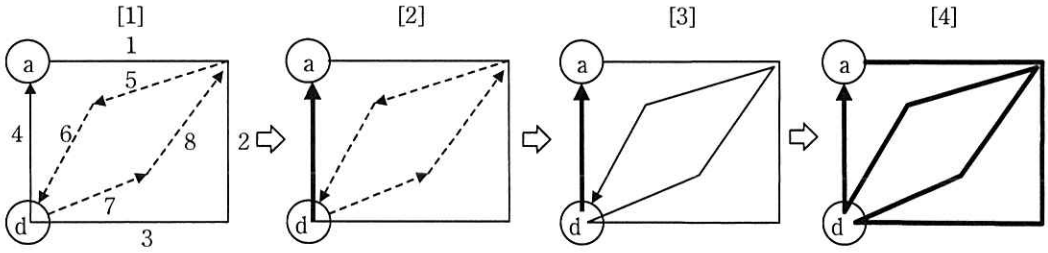
表１　図１のグラフの各点の接続辺の集合

点	接続辺の集合
点ａ	辺１
点ｂ	辺２，辺５
点ｃ	辺３
点ｄ	辺４，辺７
点ｅ	辺６
点ｆ	辺８

一筆書きの経路の探索において，一つの点に複数の接続辺がある場合には，最初の接続辺から順にたどることにする。

図１のグラフで点ａを出発点とした一筆書きの経路の求め方を図２に示す。

経路を構成する辺とその順番が，これ以上変わらない場合，確定済の経路という。



凡例　――：探索済　　----：未探索　　━━：確定済

注記１　点や辺の番号を一部省略している。

注記２　矢印は経路の向きを示す。

図２　図１のグラフで点ａを出発点とした一筆書きの経路の求め方

図２を参考にした一筆書きの経路を求める手順を次に示す。

〔一筆書きの経路を求める手順〕

点ａから探索する場合は，点ａの最初の接続辺である辺１から始め，辺１の終点ｂの最初の接続辺である辺２をたどり，同様に辺３，辺４をたどる。辺４の終点ａからたどれる未探索の辺は存在しないので，これ以上探索が進められない（図２[1]）。

しかし，未探索の辺５，辺６，辺７，辺８が残っているので，未探索の辺が接続する点まで遡る。

終点 a から辺 4 を遡ると，辺 4 の始点 d で未探索の辺 7 が接続している。遡った経路は途中で未探索の辺が存在しないので，これ以上，辺の順番が変わらず，辺 4 は，一筆書きの経路の一部として確定済の経路となる（図 2 [2]）。

点 d から同様に辺 7→辺 8→辺 5→辺 6 と探索できるので，辺 3 までの経路と連結した新しい探索済の経路ができる（図 2 [3]）。

辺 6 の終点 d からは，辺 6→辺 5→辺 8→辺 7→辺 3→辺 2→辺 1 と出発点の点 a まで遡り，これ以上，未探索の辺がないことが分かるので，全ての辺が確定済の経路になる（図 2 [4]）。

一筆書きの経路は，次の(1)～(4)の手順で求められる。

(1) 一筆書きの経路の出発点を決める。

(2) 出発点から，未探索の辺が存在する限り，その辺をたどり，たどった経路を探索済の経路に追加する。

(3) 探索済の経路を未探索の辺が接続する点又は一筆書きの経路の出発点まで遡る。遡った経路は，探索済の経路から確定済の経路にする。未探索の辺が接続する点がある場合は，それを新たな出発点として，(2)に戻って新たな経路を見つける。

(4) 全ての辺が確定済の経路になった時点で探索が完了して，その確定済の経路が一筆書きの経路になる。

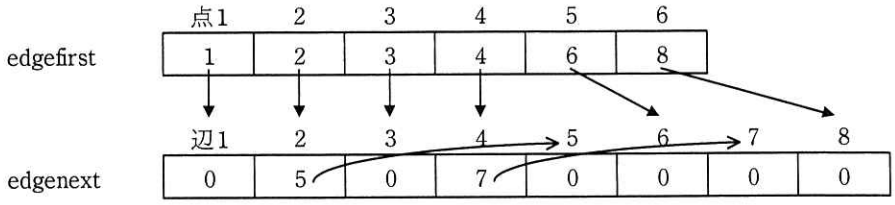
〔一筆書きの経路を求めるプログラム〕

一筆書きの経路を求める関数 directedE のプログラムを作成した。

実装に当たって，各点を点 n（n は 1～N）と記す。例えば，図 1 のグラフでは，点 a は点 1，点 b は点 2 と記す。

グラフの探索のために，あらかじめ，グラフの点に対する最初の接続辺の配列 edgefirst 及び接続辺に対する次の接続辺の配列 edgenext を用意しておく。edgenext において，次の接続辺がない場合は，要素に 0 を格納する。

図 1 のグラフの場合の配列 edgefirst，edgenext を図 3 に示す。



注記　edgefirst にはグラフの点に対する最初の接続辺の番号を格納している。
edgenext には接続辺の次の接続辺の番号を格納している。

図３　図１のグラフの場合の配列 edgefirst，edgenext

edgefirst によって点２の最初の接続辺が辺２であることが分かり，点２から最初にたどる接続辺は辺２となる。edgenext によって，辺２の次の接続辺が辺５であることが分かるので，点２から次にたどる接続辺は辺５となる。辺５の次の接続辺はないので，点２からたどる接続辺はこれ以上ないことが分かる。

プログラム中で使用する定数と配列を表２に，作成した関数 directedE のプログラムを図４に示す。

全ての配列の添字は１から始まる。

表２　使用する定数と配列

名称	種類	内容
N	定数	グラフの点の個数
M	定数	グラフの辺の個数
start[m]	配列	start[m] には，辺 m の始点の番号が格納されている。
end[m]	配列	end[m] には，辺 m の終点の番号が格納されている。
edgefirst[n]	配列	edgefirst[n] には，点 n の最初の接続辺の番号が格納されている。
edgenext[m]	配列	edgenext[m] には，辺 m の次の接続辺の番号が格納されている。次の接続辺がない場合は 0 が格納されている。
current[n]	配列	current[n] には，点 n を始点とする未探索の辺の中で最小の番号を格納する。点 n を始点とする未探索の辺がない場合は 0 を格納する。
searched[m]	配列	一筆書きの経路を構成する探索済の辺の番号を順番に格納する。（探索済の経路）
path[m]	配列	一筆書きの経路を構成する確定済の辺の番号を順番に格納する。（確定済の経路）

```
function directedE()
    for ( i を 1 から N まで 1 ずつ増やす ) // 各点での未探索の辺の番号を初期化
        current[i] ← edgefirst[i]
    endfor
    top ← 1                              // 探索済の経路の辺の格納位置を初期化
    last ← M                             // 確定済の経路の辺の格納位置を初期化
    x ← 1                                // 出発点は点 1
    while ( ①last が 1 以上 )
        if ( current[x] が [ ア ] でない )
            temp ← current[x]            // 点 x からたどる接続辺は temp
            searched[top] ← temp         // 接続辺 temp を探索済の経路に登録
            current[x] ← [ イ ]          // 点 x から次にたどる未探索の辺を格納
            x ← end[temp]                // 接続辺 temp の終点を点 x にする
            top ← top + 1
        else
            top ← [ ウ ]                 // 探索済の辺を遡る
            temp ← searched[top]         // 遡った辺は temp
            path[last] ← temp            // 辺 temp を確定済にする
            x ← [ エ ]
            last ← last − 1
        endif
    endwhile
endfunction
```

図 4　関数 directedE のプログラム

設問 1　図 4 中の [ア] ～ [エ] に入れる適切な字句を答えよ。

設問 2　図 1 のグラフで関数 directedE を動作させたとき，while 文中の if 文は，何回実行されるか，数値で答えよ。

設問 3　一筆書きができない強連結な有向グラフで関数 directedE を動作させたとき，探索はどのようになるかを，解答群の中から選び，記号で答えよ。

解答群

ア　探索が完了するが，配列 path に格納された経路は一筆書きの経路にならない。

イ　探索が完了せずに終了して，配列 path に格納された経路は一筆書きの経路にならない。

ウ　探索が無限ループに陥り，探索が終了しない。

設問 4　図 4 のプログラムは，配列 searched を配列 path に置き換えることで，使用する領域を減らすことができる。このとき，無駄な繰返しが発生しないように，下線①の繰返し条件を，変数 top と last を用いて変更せよ。