次の問８は必須問題です。必ず解答してください。

問８　次のプログラムの説明及びプログラムを読んで，設問１，２に答えよ。

副プログラム ShortestPath は，N 個（N＞1）の地点と，地点間を直接結ぶ経路及び距離が与えられたとき，出発地から目的地に至る最短経路とその距離を求めるプログラムである。最短経路とは，ある地点から別の地点へ最短距離で移動する際の経由地を結んだ経路である。副プログラム ShortestPath では，出発地の隣接地点から開始して，目的地に向かって最短距離を順次確定する。ある地点の隣接地点とは，その地点から他の地点を経由せずに直接移動できる地点のことである。

図１は，地点数 N が７の経路の例で，経路をグラフで表現したものである。図１において，丸は地点を示し，各地点には０から始まる番号（以下，地点番号という）が順番に割り当てられている。線分は地点間を直接結ぶ経路を示し，線分の上に示す数字はその距離を表す。また，経路上は，双方向に移動できる。

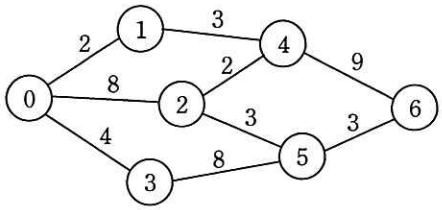

図１　地点数 N が７の経路の例

〔プログラムの説明〕

(1) 副プログラム ShortestPath の引数の仕様を，表１に示す。ここで，出発地から目的地までを結ぶ経路は，少なくとも一つ存在するものとする。また，配列の要素番号は，０から始まる。

表1　副プログラム ShortestPath の引数の仕様

引数	データ型	入出力	説明
Distance[][]	整数型	入力	地点間の距離が格納されている2次元配列
nPoint	整数型	入力	地点数
sp	整数型	入力	出発地の地点番号
dp	整数型	入力	目的地の地点番号
sRoute[]	整数型	出力	出発地から目的地までの最短経路上の地点の地点番号を目的地から出発地までの順に設定する1次元配列
sDist	整数型	出力	出発地から目的地までの最短距離

(2)　Distance[i][j]（i＝0,…,nPoint－1; j＝0,…,nPoint－1）には，地点 i から地点 j までの距離が格納されている。ただし，地点 i と地点 j が同一の地点の場合は 0，地点 i が地点 j の隣接地点ではない場合は －1 が格納されている。図1の例における配列 Distance の内容は，表2のとおりである。

表2　図1の例における配列 Distance の内容

i＼j	0	1	2	3	4	5	6
0	0	2	8	4	−1	−1	−1
1	2	0	−1	−1	3	−1	−1
2	8	−1	0	−1	2	3	−1
3	4	−1	−1	0	−1	8	−1
4	−1	3	2	−1	0	−1	9
5	−1	−1	3	8	−1	0	3
6	−1	−1	−1	−1	9	3	0

(3)　行番号5～10では，変数，配列に初期値を格納する。

①　最短距離を返却する変数 sDist に初期値として ∞（最大値を表す定数）を格納し，出発地から目的地までの最短距離が設定されていないことを示す。

②　最短経路を返却する要素数が nPoint である配列 sRoute の全ての要素に初期値として －1 を格納し，最短経路上の地点の地点番号が設定されていないことを示す。

③　出発地から各地点までの最短距離を設定する配列を pDist とする。pDist は1

次元配列であり，要素数は nPoint である。配列 pDist の全ての要素に初期値として ∞ を格納する。

④ 出発地から各地点までの最短距離が確定しているかどうかを識別するための配列を pFixed とする。pFixed は 1 次元配列であり，要素数は nPoint である。配列 pFixed の全ての要素に初期値として false を格納し，最短距離が未確定であることを示す。最短距離が確定したときに true を設定する。例えば，出発地から地点 i までの最短距離が確定したとき，pFixed[i] は true となり，その最短距離は pDist[i] に設定されている。pFixed[i] が false の場合は，地点 i までの最短距離は未確定であり，pDist[i] の値は最短距離として確定されていない。

(4) 行番号 11 では，出発地から出発地自体への最短距離 pDist[sp]に 0 を設定する。

(5) 行番号 12 ～ 39 の最短経路探索処理では，出発地から各地点までの最短距離を算出しながら，最短経路を求める。

① 行番号 13 ～ 22

配列 pFixed を調べ，出発地から全ての地点までの最短距離が確定していれば，最短経路探索処理を抜けて (6) に進む。

② 行番号 23 ～ 29

出発地からの最短距離が未確定の地点の中で，出発地からの距離が最も短い地点を探し，その地点を sPoint とし，その地点の最短距離を確定する。

③ 行番号 30 ～ 38

各地点に対して (ア), (イ) を実行し，①に戻る。

(ア) 地点 sPoint の隣接地点であり，かつ，出発地からの最短距離が未確定であるかどうかを調べる。

(イ) (ア) の条件を満たす地点 j に関して，出発地から地点 sPoint を経由して地点 j に到達する経路の距離を求め，その距離が既に算出してある pDist[j]よりも短ければ，pDist[j]及び pRoute[j]を更新する。ここで，pDist[j]は，出発地から地点 j までの仮の最短距離となる。pRoute[j]には，そのときの，地点 j の直前の経由地の地点番号を設定する。

(6) 行番号 40 ～ 48 では，出発地から目的地までの最短距離を sDist に，最短経路上の地点の地点番号を目的地から出発地までの順に配列 sRoute に設定する。

〔プログラム〕

（行番号）

```
1  ○副プログラム: ShortestPath( 整数型 : Distance[][], 整数型 : nPoint,
                    整数型 : sp, 整数型 : dp, 整数型 : sRoute[], 整数型 : sDist )
2  ○整数型 : pDist[nPoint], pRoute[nPoint]
3  ○論理型 : pFixed[nPoint]
4  ○整数型 : sPoint, i, j, newDist
5  ・sDist ← ∞     /* 出発地から目的地までの最短距離に初期値を格納する */
6  ■ i: 0, i < nPoint, 1
7  |  ・sRoute[i] ← −1  /* 最短経路上の地点の地点番号に初期値を格納する */
8  |  ・pDist[i] ← ∞  /* 出発地から各地点までの最短距離に初期値を格納する */
9  |  ・pFixed[i] ← false  /* 各地点の最短距離の確定状態に初期値を格納する */
10 ■
11 ・pDist[sp] ← 0     /* 出発地から出発地自体への最短距離に 0 を設定する */
12 ■ true  /* 最短経路探索処理 */
13 |  ・i ← 0
14 |  ■ i < nPoint            /* 未確定の地点を一つ探す */
15 |  |  ▲ not(pFixed[i])
16 |  |  |  ・break            /* 最内側の繰返しから抜ける */
17 |  |  ▼
18 |  |  ・i ← i + 1
19 |  ■
20 |  ▲ i = nPoint        /* 出発地から全ての地点までの最短距離が確定 */
21 |  |  ・break           /* していれば，最短経路探索処理を抜ける */
22 |  ▼
23 |  ■ j: i + 1, j < nPoint, 1  /* 最短距離がより短い地点を探す */
24 |  |  ▲ [ a ] and pDist[j] < pDist[i]
25 |  |  |  ・i ← j
26 |  |  ▼
27 |  ■
28 |  ・sPoint ← i  <------------------------------ α
29 |  ・pFixed[ [ b ] ] ← true  /* 出発地からの最短距離を確定する */
30 |  ■ j: 0, j < nPoint, 1
31 |  |  ▲ Distance[sPoint][j] > 0 and not(pFixed[j])
32 |  |  |  ・newDist ← pDist[sPoint] + Distance[sPoint][j]
33 |  |  |  ▲ newDist < pDist[j]
34 |  |  |  |  ・pDist[j] ← newDist
35 |  |  |  |  ・pRoute[j] ← sPoint
36 |  |  |  ▼
37 |  |  ▼
38 |  ■
39 ■  <------------------------------------------- β
```

```
40  ・sDist ← pDist[dp]
41  ・j ← 0
42  ・i ← dp
43  ■ i ≠ sp
44  |  ・[  c  ] ← i
45  |  ・i ← [  d  ]
46  |  ・j ← j + 1
47  ■
48  ・[  c  ] ← sp
```

設問1　プログラム中の［　　］に入れる正しい答えを，解答群の中から選べ。

a に関する解答群

ア　not(pFixed[i])　　イ　not(pFixed[j])

ウ　pFixed[i]　　エ　pFixed[j]

b に関する解答群

ア　dp　　イ　nPoint　　ウ　nPoint − 1

エ　sp　　オ　sPoint

c，d に関する解答群

ア　pRoute[dp]　　イ　pRoute[i]　　ウ　pRoute[j]　　エ　pRoute[sp]

オ　sRoute[dp]　　カ　sRoute[i]　　キ　sRoute[j]　　ク　sRoute[sp]

設問2　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

図1において，出発地の地点番号 sp の値が 0，目的地の地点番号 dp の値が 6 の場合について，プログラムの動きを追跡する。行番号 12 ～ 39 の最短経路探索処理の繰返しで，行番号 28 の α において sPoint に代入された値は，繰返しの 1 回目は 0，2 回目は 1，3 回目は［　e　］となる。また，行番号 30 ～ 38 の処理が終了した直後の β における配列 pDist と配列 pRoute の値を，表3に示す。最短経路探索処理の繰返しが 3 回目のときの β における配列 pDist の値は［　f　］となり，配列 pRoute の値は［　g　］となる。ここで，配列

pRoute の全ての要素には初期値として 0 が格納されているものとする。

表3　βにおける配列 pDist と配列 pRoute の値

最短経路探索処理の繰返し	配列 pDist	配列 pRoute
1回目	0,2,8,4, ∞ , ∞ , ∞	0,0,0,0,0,0,0
2回目	0,2,8,4,5, ∞ , ∞	0,0,0,0,1,0,0
3回目	f	g
⋮	⋮	⋮

e に関する解答群

ア　2　　イ　3　　ウ　4　　エ　5　　オ　6

f に関する解答群

ア　0,2,8,4,5,10,14

イ　0,2,8,4,5,10, ∞

ウ　0,2,8,4,5,11,14

エ　0,2,8,4,5,11, ∞

オ　0,2,8,4,5,12,14

カ　0,2,8,4,5,12, ∞

g に関する解答群

ア　0,0,0,0,1,3,0

イ　0,0,0,0,1,3,5

ウ　0,0,0,0,2,2,0

エ　0,0,0,0,2,2,5

オ　0,0,4,0,1,2,0

カ　0,0,4,0,2,2,5