## 第 3 問　（選択問題）　次の文章を読み，後の問い（問 1 ～ 3 ）に答えよ。（配点　35）

A さんはゲーム会社に勤務しており，複数のプレイヤーが参加するゲームでのプレイヤーの順番（以下，**手番**と呼ぶ。）を管理するプログラムを開発している。ここでは図 1 に示すように，「高橋」を先頭として「高橋」→「石村」→「天野」→「小池」→「渡辺」の順で時計回りに手番が進み，最後の「渡辺」の次は「高橋」に戻り，再び同じ順で手番が繰り返される例について考える。

**問 1**　次の文章を読み，空欄 **ア** ・ **イ** に当てはまる数字をマークせよ。また，空欄 **ウ** ・ **エ** に入れるのに最も適当なものを，後の解答群のうちから一つずつ選べ。

A さんは「プレイヤーの情報は配列で管理するが，ゲームでの手番は配列の並びにかかわらず，自由に順序を設定できるようにしてほしい」と指示を受けた。そこで，表 1 のように列 1 にはプレイヤーの名前，列 2 には「次のプレイヤー」の行番号を格納するデータの構造を考えた。列 2 は図 1 の矢印に相当し，「高橋」の次のプレイヤーを指し示す矢印は「石村」を指しているので，表 1 における「高橋」の行の列 2 には「石村」の行番号である 3 が入る。「石村」の次のプレイヤーは「天野」なので，「石村」の行の列 2 には **ア** が入る。最後の「渡辺」の次のプレイヤーは先頭の「高橋」なので，「渡辺」の行の列 2 には **イ** が入る。なお，設問の都合により，表 1 の一部は値を“？”で隠している。

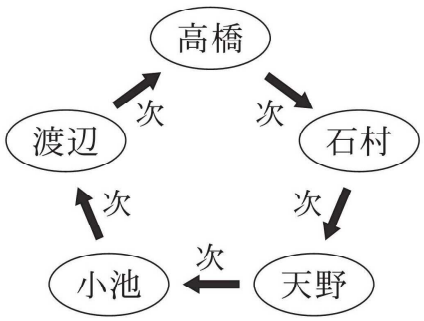


図 1　プレイヤーの手番の並び

表 1　手番を管理するデータ

| 行＼列 | 1（名前） | 2（次） |
|---|---|---|
| 1 | 小池 | ？ |
| 2 | 天野 | ？ |
| 3 | 石村 | ア |
| 4 | 渡辺 | イ |
| 5 | 高橋 | 3 |

Aさんは，表1を**Player**という名前の2次元配列で扱うことにした。表1の行番号と列番号を添字として用い，**Player[**＜行番号＞**,**＜列番号＞**]**と表す。ただし，行番号と列番号に0は用いないこととする。5行目の「高橋」を例にすると，名前は5行1列にあるため，**Player[5,1]**に「高橋」を格納し，次のプレイヤーの行番号は5行2列にあるため，**Player[5,2]**に3を格納する。

Aさんはこの配列を用いて，ゲームに参加しているすべてのプレイヤーの名前を，先頭から手番の順に表示する手続きを図2のように書いた。変数**sento**は先頭のプレイヤーの行番号を，変数**n**はゲームの参加者数を，それぞれ格納している。変数**p**を用いて手番の並びを**n**人分たどりながら，**(03)**行目でプレイヤーの名前を表示する手続きとなっている。

```
(01)  p ← sento
(02)  i を 1 から n まで 1 ずつ増やしながら，
(03)  |   [ ウ ] を表示する
(04)  |   p ← [ エ ]
(05)  を繰り返す
```

図2　プレイヤーの名前を手番の順に表示する手続き

[ ウ ] · [ エ ] の解答群

| | | |
|---|---|---|
| ⓪ sento | ① p ＋ 1 | ② Player[p,p] |
| ③ Player[p,1] | ④ Player[p,2] | ⑤ Player[sento,p] |
| ⑥ Player[i,1] | ⑦ Player[i,2] | ⑧ Player[i,p] |

**問 2**　次の文章を読み，空欄 **オ** ～ **ス** に入れるのに最も適当なものを，後の解答群のうちから一つずつ選べ。

このゲームでは，プレイヤーがゲームに新たに参加したり抜けたりすることがある。図 3 ・図 4 は，図 1 で示した手番の並びでの例である。A さんは，プレイヤーを手番の並びに追加する手続きを考えた。図 3 では，「高橋」と「石村」の間に「三田」が新たに追加されている。この場合，「三田」の次のプレイヤーは「高橋」の次のプレイヤーである **オ** になり，「高橋」の次のプレイヤーは **カ** になる。つまり，あるプレイヤー(行番号 **x**)の次に新たなプレイヤー(行番号 **tuika**)を追加する手順は，図 5 のように，配列 **Player** においてそれぞれの列 2 の値を変更する手続きになる。**(03)** 行目ではゲームの参加者数 **n** の値を変更している。

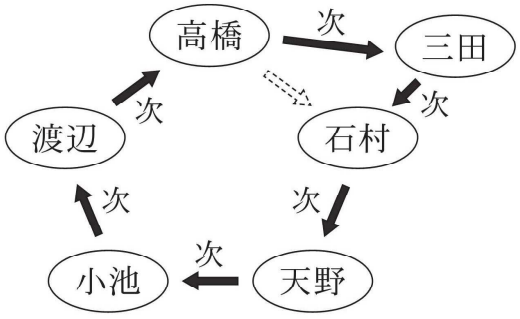


図 3　プレイヤーを追加する例

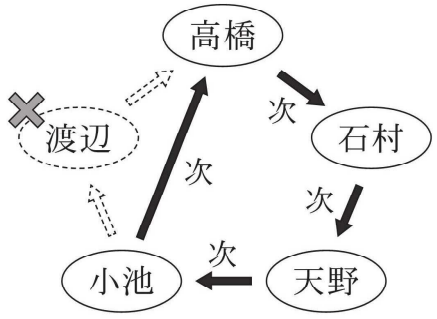


図 4　プレイヤーが抜ける例

```
(01)  Player[tuika,2]← キ
(02)  Player[x,2]← ク
(03)  n← ケ
```

図 5　プレイヤーを手番の並びに追加する手続き

次に，A さんは任意のプレイヤーが手番の並びから抜ける手続きの作成にとりかかった。抜けるプレイヤーを配列から削除するのではなく，手番の並びを変更し，そのプレイヤーにたどり着かないようにする手順を考えた。図 4 は，図 1 で示した手番の並びから「渡辺」が抜ける例である。この場合，「渡辺」の前にいる「小池」の次のプレイヤーは **コ** になる。そこで，手番の並びから抜けるプレイヤー(行番号 **nuke**)の前にいるプレイヤーを特定し，そのプレイ

ヤーの列 2 の値を変更する手続きを図 6 のように書いた。**問 1** と同じく，変数 **sento** は先頭のプレイヤーの行番号を格納している。変数 **p** を用いて先頭から手番の並びをたどりながら，抜けるプレイヤーの前にいるプレイヤーの行番号を求め，**(05)**行目でそのプレイヤーの列 2 の値を変更している。**(06)**行目では，ゲームの参加者数 **n** の値を変更している。この手続きでは，先頭のプレイヤーが抜けた場合の変数 **sento** の値の変更処理は省略している。

```
(01)  p ← sento
(02)  [ サ ] の間,
(03)  |   p ← [ エ ]
(04)  を繰り返す
(05)  Player[p,2] ← [ シ ]
(06)  n ← [ ス ]
```

図 6　プレイヤーが手番の並びから抜ける手続き

[ オ ] · [ カ ], [ コ ] の解答群

| | | |
|---|---|---|
| ⓪ 「高橋」 | ① 「石村」 | ② 「天野」 |
| ③ 「小池」 | ④ 「渡辺」 | ⑤ 「三田」 |

[ キ ] ~ [ ケ ], [ シ ] · [ ス ] の解答群

| | | |
|---|---|---|
| ⓪ Player[x,2] | ① Player[nuke,2] | ② Player[tuika,2] |
| ③ n | ④ n + 1 | ⑤ n − 1 |
| ⑥ tuika | ⑦ tuika + 1 | ⑧ tuika − 1 |
| ⑨ nuke | ⓐ nuke + 1 | ⓑ nuke − 1 |

[ サ ] の解答群

| | |
|---|---|
| ⓪ Player[p,2] = sento | ① Player[p,2] ≠ sento |
| ② Player[p,2] = nuke | ③ Player[p,2] ≠ nuke |

**問 3** 次の文章を読み，空欄 セ ～ チ に当てはまる数字をマークせよ。また，空欄 ツ ～ ナ に入れるのに最も適当なものを，後の解答群のうちから一つずつ選べ。

A さんは，各プレイヤーの前の手番にいるプレイヤーを事前にすべて求めておけば，プレイヤーが手番の並びから抜ける手続きが簡素になると考えた。図 7 は，図 1 に「前のプレイヤー」を指し示す白矢印を追記した図である。表 2 は，図 7 の白矢印の情報を管理できるように，表 1 に「前のプレイヤー」の行番号を管理する列 3 を追加したものである。「高橋」を例にすると，「高橋」の前のプレイヤーは「渡辺」なので，表 2 において「高橋」の行の列 3 は 4 になる。同様に，「石村」と「渡辺」の行の列 3 は，それぞれ セ ，ソ となる。なお，設問の都合により，表 2 の一部は値を“？”で隠している。

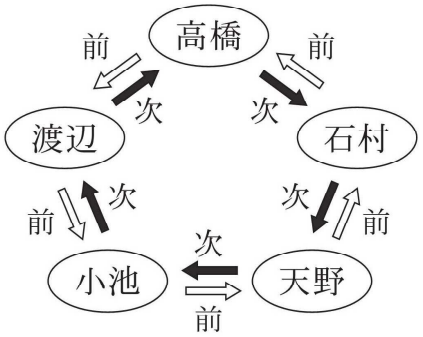


図 7　プレイヤーの手番の並び
（図 1 に白矢印を追記）

表 2　表 1 に前のプレイヤーの情報を加えたデータ

| 行＼列 | 1（名前） | 2（次） | 3（前） |
|---|---|---|---|
| 1 | 小池 | ? | ? |
| 2 | 天野 | ? | ? |
| 3 | 石村 | ア | セ |
| 4 | 渡辺 | イ | ソ |
| 5 | 高橋 | 3 | 4 |

表 2 も 2 次元配列 **Player** で扱うことができる。「高橋」を例にすると，「高橋」の前のプレイヤーの行番号は 4 であることから，**Player[** タ **,** チ **]** に 4 を格納する。

A さんは，列 2 の値をもとに列 3 の値を自動的に求める手続きを作成した。図 8 では，図 2 と同様に変数 **p** を用いて先頭の行番号（変数 **sento**）から **n** 人分まで手番の並びをたどりながら，**(03)** 行目で変数 **q** に「次のプレイヤー」の行番号を格納し，**(04)** 行目で「次のプレイヤー」の列 3 に現在のプレイヤーの行番号を格納している。なお，現在のプレイヤーとは，変数 **p** を用いてたどっている手番のプレイヤーである。

最後に A さんは，表 2 の情報を用いて任意のプレイヤー（行番号 **nuke**）が手番の並びから抜ける手続きを，図 9 のように考えた。**(01)** 行目では変数 **mae** に抜けるプレイヤーの前にいるプレイヤーの行番号を，**(02)** 行目では変数

tugi に抜けるプレイヤーの次のプレイヤーの行番号を，それぞれ格納している。(03)～(04)行目では，プレイヤーが抜けた後の手番の並びになるように，これらの変数の値を配列 Player に格納している。(05)行目ではゲームの参加者数 n の値を変更している。なお，問 2 と同様，先頭のプレイヤーが抜けた場合の処理は省略している。

```
(01)  p ← sento
(02)  i を 1 から n まで 1 ずつ増やしながら，
(03)  |   q ← [ツ]
(04)  |   Player[q,3] ← [テ]
(05)  |   p ← [エ]
(06)  を繰り返す
```

図 8　列 3 の値を求める手続き

```
(01)  mae ← Player[nuke,3]
(02)  tugi ← Player[nuke,2]
(03)  [ト] ← tugi
(04)  [ナ] ← mae
(05)  n ← [ス]
```

図 9　表 2 の情報を用いて手番の並びからプレイヤーが抜ける手続き

[ツ] · [テ] の解答群

⓪ Player[p,2]　① Player[q,2]　② Player[i,2]
③ p　④ q　⑤ i

[ト] · [ナ] の解答群

⓪ Player[mae,2]　① Player[mae,3]
② Player[tugi,2]　③ Player[tugi,3]
④ Player[nuke,2]　⑤ Player[nuke,3]