問8　文字列の誤りの検出（データ構造及びアルゴリズム）（H29秋-FE 午後問8）

【解答】

［設問1］　a－エ，b－ウ，c－エ
［設問2］　d－オ
［設問3］　e－ウ
［設問4］　f－カ

【解説】

与えられた手順に従って文字列の誤りを検出するための検査文字の生成と，検査文字付文字列の検証を行うアルゴリズムの問題である。

問題文に示された手順を理解して，〔プログラムの説明〕や〔検査文字付文字列の生成例〕から実際の計算をイメージしながら把握することが重要である。

手順を確認していく。

まず，問題文冒頭には「文字列の誤りを検出するために，N種類の文字に0，1，…，N－1の整数値を重複なく割り当て，検査文字を生成するプログラムと，元となる文字列の末尾に検査文字を追加した検査文字付文字列を検証するプログラムである。ここで扱う30種類の文字，及び文字に割り当てた数値を，表1に示す」と記述されている。

次に，〔プログラムの説明〕の(1)検査文字の生成を確認していく。

① 文字列の末尾の文字を1番目の文字とし，文字列の先頭に向かって奇数番目の文字に割り当てた数値を2倍してNで割り，商と余りの和を求め，全て足し合わせる。
② 偶数番目の文字に割り当てた数値は，そのまま全て足し合わせる。
③ ①と②の結果を足し合わせる。
④ Nから，③で求めた総和をNで割った余りを引く。さらにその結果を，Nで割り，余りを求める。求めた数値に対応する文字を検査文字とする。

N＝30，文字列ipa␣␣を使って具体的に計算していくと，表Aのようになる。

表A

		5番目	4番目	3番目	2番目	1番目
文字列		i	p	a	␣	␣
文字に割り当てた数値		12	19	4	0	0
①奇数番目	2倍する。	24	－	8	－	0
	30で割り，商と余りの和を求める。	24÷30＝0…24 0＋24＝24	－	8÷30＝0…8 0＋8＝8	－	0÷30＝0…0 0＋0＝0
	全て足し合わせる。	24＋8＋0＝32				
②偶数番目	そのまま全て足し合わせる。	19＋0＝19				
③	①と②を足し合わせる。	32＋19＝51				
④	30から，③の総和を30で割った余りを引く。	51÷30＝1…21　　30－21＝9				
	さらに30で割り，余りを求める。	9÷30＝0…9				
	対応する文字を検査文字とする。	表1を確認すると9に対応する文字はfになる。				

次に，(2)検査文字付文字列の検証も確認していく。

① 検査文字付文字列の末尾の文字を1番目の文字とし，文字列の先頭に向かって偶数番目の文字に割り当てた数値を2倍してNで割り，商と余りの和を求め，全て足し合わせる。
② 奇数番目の文字に割り当てた数値は，そのまま全て足し合わせる。
③ ①と②の結果を足し合わせる。
④ ③で求めた総和がNで割り切れる場合は，検査文字付文字列に誤りがないと判定する。Nで割り切れない場合は，検査文字付文字列に誤りがあると判定する。

N＝30，文字列ipa␣␣使って具体的に計算していくと，表Bのようになる。

表B

		6番目	5番目	4番目	3番目	2番目	1番目
文字列		i	p	a	␣	␣	f
文字に割り当てた数値		12	19	4	0	0	9
①偶数番目	2倍する。	24	－	8	－	0	－
	30で割り，商と余りの和を求める。	24÷30＝0…24 0＋24＝24	－	8÷30＝0…8 0＋8＝8	－	0÷30＝0…0 0＋0＝0	－
	全て足し合わせる。	24＋8＋0＝32					
②奇数番目	そのまま全て足し合わせる。	19＋0＋9＝28					
③	①と②を足し合わせる。	32＋28＝60					
④	③の総和を30で割る。	60÷30＝2…0　　割り切れる場合は，誤りがない。					

さらに〔プログラムの仕様〕と関数calcCheckCharacterのプログラムを確認する。検査文字を生成する文字列とその文字列の長さを渡され，検査文字を返す関数である。表2から引数は次の二つである。

・input[]：検査文字を生成する文字列を関数に渡す1次元配列である。
・len：文字列の文字列長（1以上）を関数に渡す。

表1に従って文字に割り当てられた数値を求めるには，文字を引数として渡し，割り当てられた数値を返す関数getValueを使用する。また，求めた検査文字に対応する数値から表1に従って文字に変換するには，数値を引数として渡し，対応する文字を返す関数getCharを使用する。

プログラムの中で使用される変数は，次のとおりである。

・N：文字の種類の数を表す。
・sum：奇数番目の計算と偶数番目の計算の総和を求める。
・is_even：偶数番目かどうかを判定する。
・i：配列input[]が処理する文字の添字を示す。
・value：文字に割り当てられた数値を格納する。
・check_value：検査文字に対応する数値を格納する。

続いて，プログラムを確認する。行番号とともに次に示す。

〔プログラム〕
（行番号）

```
 1  ○文字型関数：calcCheckCharacter（文字型：input[]，整数型：len）
 2  ○整数型：N，sum，i，value，check_value
 3  ○論理型：is_even
 4  ・N ← 30
 5  ・sum ← 0
 6  ・is_even ← false
 7  ■i：len，i ＞ 0，－1
 8  |・value ← getValue(input[i])
 9  |▲is_even ＝ [ a1 ]
10  |├・sum ← sum ＋ value
11  |▼・sum ← sum ＋ (value × 2) ÷ N ＋ (value × 2) % N
12  |
13  |・is_even ← not is_even
14  ■
15  ・check_value ← [ b ]
16  ・return getChar(check_value)
```

・行番号4～6：変数の初期設定を行う。
　・行番号4：文字の種類の数Nに30を設定する。
　・行番号5：偶数番目，奇数番目の文字に対応する数値から求めた値の総和を求める変数sumに初期値0を設定する。
　・行番号6：偶数番目かどうかを判定するフラグis_evenにfalseを設定し“奇数番目”とする。
・行番号7～14：末尾の文字である1番目の文字から，文字列の長さ分だけ前に順に1文字ずつ，割り当てられた数値に変換する。文字の位置が偶数番目なら偶数番目の計算，奇数番目なら奇数番目の計算をそれぞれ行い，総和をsumとして求める。〔プログラムの説明〕(1)検査文字の生成の①，②，③に対応する処理である。
　・行番号7：1番目の文字は末尾の文字であり，引数で与えられるinput[]の配列の添字をlenから1まで，0より大きい間，順に－1ずつ変化させる。文字列の末尾である1番目input[len]からinput[1]まで1文字ずつ処理対象にする。
　・行番号8：getValue関数で処理対象の文字input[i]を表1に対応する数値に変換しvalueに設定する。
　・行番号9～12：偶数番目か奇数番目かを条件分けし，偶数番目なら総和sumに値を加算し（行番号10），奇数番目なら数値に変換した値を2倍してNで割った商と余りを合計に加算する（行番号11）。午後問題冒頭の「共通に使用される擬似言語の記述形式」の〔演算子と優先順位〕の注記に「整数同士の除算では，整数の商を結果として返す。%演算子は，剰余算を表す」とある。したがって，(value × 2)もNも整数なので，valueを2倍してNで割った商は(value × 2) ÷ Nであり，余りは(value × 2) % Nで求められる。
　・行番号13：論理型の変数is_evenの否定をis_evenに設定することで，falseならtrueに，trueならfalseに値が変わる。これによって，偶数番目，奇数番目が交互に設定される。
・行番号15：Nから，行番号7～14で求めた総和sumをNで割った余りを引いて，その結果を，Nで割った余りをcheck_valueとして求める処理であると考えられる。
・行番号16：check_valueで求めた数値を，関数getCharを使用して表1の対応する文字に変換し，検査文字として返す。行番号15，16は(1)検査文字の生成の④に対応する処理である。

続いて，関数validateCheckCharacterのプログラムを確認する。検査文字付文字列を検証し，文字列に誤りがなければtrueを，誤りがあればfalseを返すプログラムである。表3から引数は次の二つである。

・input[]：検査文字付文字列を関数に渡す1次元配列である。
・len：文字列の文字列長（2以上）を関数に渡す。

表1に従って文字に割り当てられた数値を求めるには，関数getValueを使用する。

プログラムの中で使用される変数N，sum，i，valueは関数calcCheckCharacterと同じである。

そのほかに，プログラムの中で使用される変数は，次のとおりである。

・is_odd：奇数番目かどうかを判定する
・ret_value：論理型　戻り値（true，又はfalse）

続いて，プログラムを確認する。行番号とともに次に示す。

〔プログラム〕
（行番号）

```
21  ○論理型関数：validateCheckCharacter（文字型：input[]，整数型：len）
22  ○整数型：N，sum，i，value
23  ○論理型：is_odd，ret_value
24  ・N ← 30
25  ・sum ← 0
26  ・is_odd ← true
27  ・ret_value ← true
28  ■i：len，i ＞ 0，－1
29  |・value ← getValue(input[i])
30  |▲is_odd ＝ [ a2 ]
31  |├・sum ← sum ＋ value
32  |▼・sum ← sum ＋ (value × 2) ÷ N ＋ (value × 2) % N
33  |
34  |・is_odd ← not is_odd
35  ■
```

問8　文字列の誤りの検出（データ構造及びアルゴリズム）（H29秋-FE 午後問8）

【解答】

［設問1］　a－エ，b－ウ，c－エ
［設問2］　d－オ
［設問3］　e－ウ
［設問4］　f－カ

【解説】

与えられた手順に従って文字列の誤りを検出するための検査文字の生成と，検査文字付文字列の検証を行うアルゴリズムの問題である。

問題文に示された手順を理解して，〔プログラムの説明〕や〔検査文字付文字列の生成例〕から実際の計算をイメージしながら把握することが重要である。

手順を確認していく。

まず，問題文冒頭には「文字列の誤りを検出するために，N種類の文字に0，1，…，N－1の整数値を重複なく割り当て，検査文字を生成するプログラムと，元となる文字列の末尾に検査文字を追加した検査文字付文字列を検証するプログラムである。ここで扱う30種類の文字，及び文字に割り当てた数値を，表1に示す」と記述されている。

次に，〔プログラムの説明〕の(1)検査文字の生成を確認していく。

① 文字列の末尾の文字を1番目の文字とし，文字列の先頭に向かって奇数番目の文字に割り当てた数値を2倍してNで割り，商と余りの和を求め，全て足し合わせる。
② 偶数番目の文字に割り当てた数値は，そのまま全て足し合わせる。
③ ①と②の結果を足し合わせる。
④ Nから，③で求めた総和をNで割った余りを引く。さらにその結果を，Nで割り，余りを求める。求めた数値に対応する文字を検査文字とする。

N＝30，文字列ipa␣␣を使って具体的に計算していくと，表Aのようになる。

表A

		5番目	4番目	3番目	2番目	1番目
文字列		i	p	a	␣	␣
文字に割り当てた数値		12	19	4	0	0
①奇数番目	2倍する。	24	－	8	－	0
	30で割り，商と余りの和を求める。	24÷30＝0…24 0＋24＝24	－	8÷30＝0…8 0＋8＝8	－	0÷30＝0…0 0＋0＝0
	全て足し合わせる。	24＋8＋0＝32				
②偶数番目	そのまま全て足し合わせる。	19＋0＝19				
③	①と②を足し合わせる。	32＋19＝51				
④	30から，③の総和を30で割った余りを引く。	51÷30＝1…21　　30－21＝9				
	さらに30で割り，余りを求める。	9÷30＝0…9				
	対応する文字を検査文字とする。	表1を確認すると9に対応する文字はfになる。				

次に，(2)検査文字付文字列の検証も確認していく。

① 検査文字付文字列の末尾の文字を1番目の文字とし，文字列の先頭に向かって偶数番目の文字に割り当てた数値を2倍してNで割り，商と余りの和を求め，全て足し合わせる。
② 奇数番目の文字に割り当てた数値は，そのまま全て足し合わせる。
③ ①と②の結果を足し合わせる。
④ ③で求めた総和がNで割り切れる場合は，検査文字付文字列に誤りがないと判定する。Nで割り切れない場合は，検査文字付文字列に誤りがあると判定する。

N＝30，文字列ipa␣␣使って具体的に計算していくと，表Bのようになる。

表B

		6番目	5番目	4番目	3番目	2番目	1番目
文字列		i	p	a	␣	␣	f
文字に割り当てた数値		12	19	4	0	0	9
①偶数番目	2倍する。	24	－	8	－	0	－
	30で割り，商と余りの和を求める。	24÷30＝0…24 0＋24＝24	－	8÷30＝0…8 0＋8＝8	－	0÷30＝0…0 0＋0＝0	－
	全て足し合わせる。	24＋8＋0＝32					
②奇数番目	そのまま全て足し合わせる。	19＋0＋9＝28					
③	①と②を足し合わせる。	32＋28＝60					
④	③の総和を30で割る。	60÷30＝2…0　　割り切れる場合は，誤りがない。					

さらに〔プログラムの仕様〕と関数calcCheckCharacterのプログラムを確認する。検査文字を生成する文字列とその文字列の長さを渡され，検査文字を返す関数である。表2から引数は次の二つである。

・input[]：検査文字を生成する文字列を関数に渡す1次元配列である。
・len：文字列の文字列長（1以上）を関数に渡す。

表1に従って文字に割り当てられた数値を求めるには，文字を引数として渡し，割り当てられた数値を返す関数getValueを使用する。また，求めた検査文字に対応する数値から表1に従って文字に変換するには，数値を引数として渡し，対応する文字を返す関数getCharを使用する。

プログラムの中で使用される変数は，次のとおりである。

・N：文字の種類の数を表す。
・sum：奇数番目の計算と偶数番目の計算の総和を求める。
・is_even：偶数番目かどうかを判定する。
・i：配列input[]が処理する文字の添字を示す。
・value：文字に割り当てられた数値を格納する。
・check_value：検査文字に対応する数値を格納する。

続いて，プログラムを確認する。行番号とともに次に示す。

〔プログラム〕
（行番号）

```
 1  ○文字型関数：calcCheckCharacter（文字型：input[]，整数型：len）
 2  ○整数型：N，sum，i，value，check_value
 3  ○論理型：is_even
 4  ・N ← 30
 5  ・sum ← 0
 6  ・is_even ← false
 7  ■i：len，i ＞ 0，－1
 8  |・value ← getValue(input[i])
 9  |▲is_even ＝ [ a1 ]
10  |├・sum ← sum ＋ value
11  |▼・sum ← sum ＋ (value × 2) ÷ N ＋ (value × 2) % N
12  |
13  |・is_even ← not is_even
14  ■
15  ・check_value ← [ b ]
16  ・return getChar(check_value)
```

・行番号4～6：変数の初期設定を行う。
　・行番号4：文字の種類の数Nに30を設定する。
　・行番号5：偶数番目，奇数番目の文字に対応する数値から求めた値の総和を求める変数sumに初期値0を設定する。
　・行番号6：偶数番目かどうかを判定するフラグis_evenにfalseを設定し“奇数番目”とする。
・行番号7～14：末尾の文字である1番目の文字から，文字列の長さ分だけ前に順に1文字ずつ，割り当てられた数値に変換する。文字の位置が偶数番目なら偶数番目の計算，奇数番目なら奇数番目の計算をそれぞれ行い，総和をsumとして求める。〔プログラムの説明〕(1)検査文字の生成の①，②，③に対応する処理である。
　・行番号7：1番目の文字は末尾の文字であり，引数で与えられるinput[]の配列の添字をlenから1まで，0より大きい間，順に－1ずつ変化させる。文字列の末尾である1番目input[len]からinput[1]まで1文字ずつ処理対象にする。
　・行番号8：getValue関数で処理対象の文字input[i]を表1に対応する数値に変換しvalueに設定する。
　・行番号9～12：偶数番目か奇数番目かを条件分けし，偶数番目なら総和sumに値を加算し（行番号10），奇数番目なら数値に変換した値を2倍してNで割った商と余りを合計に加算する（行番号11）。午後問題冒頭の「共通に使用される擬似言語の記述形式」の〔演算子と優先順位〕の注記に「整数同士の除算では，整数の商を結果として返す。%演算子は，剰余算を表す」とある。したがって，(value × 2)もNも整数なので，valueを2倍してNで割った商は(value × 2) ÷ Nであり，余りは(value × 2) % Nで求められる。
　・行番号13：論理型の変数is_evenの否定をis_evenに設定することで，falseならtrueに，trueならfalseに値が変わる。これによって，偶数番目，奇数番目が交互に設定される。
・行番号15：Nから，行番号7～14で求めた総和sumをNで割った余りを引いて，その結果を，Nで割った余りをcheck_valueとして求める処理であると考えられる。
・行番号16：check_valueで求めた数値を，関数getCharを使用して表1の対応する文字に変換し，検査文字として返す。行番号15，16は(1)検査文字の生成の④に対応する処理である。

続いて，関数validateCheckCharacterのプログラムを確認する。検査文字付文字列を検証し，文字列に誤りがなければtrueを，誤りがあればfalseを返すプログラムである。表3から引数は次の二つである。

・input[]：検査文字付文字列を関数に渡す1次元配列である。
・len：文字列の文字列長（2以上）を関数に渡す。

表1に従って文字に割り当てられた数値を求めるには，関数getValueを使用する。

プログラムの中で使用される変数N，sum，i，valueは関数calcCheckCharacterと同じである。

そのほかに，プログラムの中で使用される変数は，次のとおりである。

・is_odd：奇数番目かどうかを判定する
・ret_value：論理型　戻り値（true，又はfalse）

続いて，プログラムを確認する。行番号とともに次に示す。

〔プログラム〕
（行番号）

```
21  ○論理型関数：validateCheckCharacter（文字型：input[]，整数型：len）
22  ○整数型：N，sum，i，value
23  ○論理型：is_odd，ret_value
24  ・N ← 30
25  ・sum ← 0
26  ・is_odd ← true
27  ・ret_value ← true
28  ■i：len，i ＞ 0，－1
29  |・value ← getValue(input[i])
30  |▲is_odd ＝ [ a2 ]
31  |├・sum ← sum ＋ value
32  |▼・sum ← sum ＋ (value × 2) ÷ N ＋ (value × 2) % N
33  |
34  |・is_odd ← not is_odd
35  ■
```