

問題 4 次のプログラムの説明および疑似言語の記述形式の説明を読み、設問に答えよ。

[プログラムの説明]

1 次元配列 word の中に格納された文字データを配列内で中央揃えする関数 cent である。

1 次元配列 word の要素数は N 個とし、添字は 0 から始まる。よって、1 次元配列 word の 0～N-1 番目に文字データが格納されている。

0	1	2	3	4	5	6	...	N-2	N-1
△	m	o	j	i	△	△	...	△	△

△は空白を表す

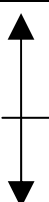
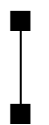
図 1 配列 word

配列wordに与えられる文字は、1つの単語だけであり、単語の前後には空白が格納されている。

プログラムでは、配列wordを作業用配列wkに転記し、単語の前後の空白を数えて、一旦すべて空白で埋められた配列wordの適切な位置に単語が戻される。作業用配列wkの要素数は処理に十分な大きさであり、添字は0から始まる。

空白の個数が奇数の場合は、単語の後ろの空白が前より1個多くなる。

[疑似言語の記述形式の説明]

記述形式	説明
○	手続き、変数などの名前、型などを宣言する
・変数 ← 式	変数に式の値を代入する
{文}	注釈を記述する
 条件式 ・処理 1 ・処理 2	選択処理を示す。 条件式が真の時は処理 1 を実行し、 偽の時は処理 2 を実行する。
 条件式 ・処理	前判定繰り返し処理を示す。 条件式が真の間、処理を実行する。

<設問> プログラム中の□に入るべき適切な字句を解答群から選べ。

[プログラム]

○cent (文字型: word[], 整数型: N)

○文字型: wk[]

○整数型: i, st, en

・ i ← 0

```
■ □ (1)
  ・ wk[i] ← word[i]
  ・ word[i] ← " "
  ・ i ← i + 1
```

・ st ← 0

```
■ wk[st] = " "
  ・ □ (2)
```

・ en ← N - 1

```
■ wk[en] = " "
  ・ □ (3)
```

・ □ (4)

```
■ st <= en
  ・ □ (5)
  ・ i ← i + 1
  ・ st ← st + 1
```

・ return

(1) の解答群

ア. i < N - 1

イ. i < N

ウ. i < N + 1

(2), (3) の解答群

ア. i ← i + 1

イ. i ← i - 1

ウ. st ← st + 1

エ. st ← st - 1

オ. en ← en + 1

カ. en ← en - 1

(4) の解答群

ア. $i \leftarrow en - st + 1$
イ. $i \leftarrow en + st - 1$
ウ. $i \leftarrow N - en - st + 1$
エ. $i \leftarrow N - en + st - 1$
オ. $i \leftarrow (N - en - st + 1) / 2$
カ. $i \leftarrow (N - en + st - 1) / 2$

(5) の解答群

ア. $wk[i] \leftarrow word[st]$	イ. $wk[st] \leftarrow word[i]$
ウ. $word[i] \leftarrow wk[st]$	エ. $word[st] \leftarrow wk[i]$

< 選 択 問 題 >

選択問題は問題から1つ選択し解答せよ。

選択した問題は必ず、解答用紙「選択欄」にマークすること。

※選択欄にマークがなく、解答のみの場合は採点を行いません。

各構成は以下のとおり。

選択問題

- | | |
|------------|---------------|
| ・ C 言語の問題 | 15 ページ～18 ページ |
| ・ 表計算の問題 | 19 ページ～23 ページ |
| ・ アセンブラの問題 | 24 ページ～28 ページ |

次のC言語プログラムの説明を読み、各設問に答えよ。

[プログラムの説明]

0から9の数値の中から、4つの数値を当てるゲームがある。4つの数値はコンピュータが重複しないようにランダムに設定する。

この数値を当てるために、キーボードから4つの数値を入力する。コンピュータの設定した値と比べ、入力した値に対して次のような判断を行う。

判断 1: キーボードから入力した値の1つが、コンピュータが設定した値の1つと一致し、その位置も一致している場合、「1ヒット」とする。

判断 2: キーボードから入力した値の1つが、コンピュータが設定した値の1つと一致するが、位置は異なっている場合、「1ブロー」とする。

キーボードから4つの値を入力するごとに、「ヒット」の数と「ブロー」の数を表示してヒントとする。このゲームは、4つの値とその並びが完全に一致すれば終了となる。

(例) コンピュータの設定した値: 0468

キーボードから入力した値: 0346

この例では、キーボードから入力した「0」は値と位置が合っており、「4」と「6」は一致する値はあるが位置は異なる。この場合は、1ヒット2ブローとなる。

このゲームの流れは、次のようになる。

1. コンピュータで4つの数値を設定し、配列へ格納する。
2. 以下の処理を4つの数値の値と位置が一致するまで繰り返す。
 - 2.1 キーボードから値を入力する。
 - 2.2 入力した数値が4つでなければエラーメッセージを表示し、2.1へ戻る。
 - 2.3 入力した数値を1つずつ配列に格納する。
 - 2.4 入力した値に重複があればエラーメッセージを表示し、2.1へ戻る。
 - 2.5 ヒットの数を求める。
 - 2.6 ブローの数を求める。
 - 2.7 ヒットとブローの数を表示する。

この中の、2.4の処理を関数 check、2.5の処理を関数 hit、2.6の処理を関数 blowで行う。

[関数の説明]

check 関数

引 数：整数型配列

機 能：配列中の値に重複があるか調べる

戻り値：重複がなければ 0, 重複があれば 1

hit 関数

引 数：整数型配列 1, 整数型配列 2

機 能：コンピュータの設定した 4 つの数値（整数型配列 1）とキーボードから入力した 4 つの数値（整数型配列 2）の値と位置が一致する要素位置の数を数える

戻り値：一致した要素位置の数

blow 関数

引 数：整数型配列 1, 整数配列 2

機 能：コンピュータの設定した 4 つの数値（整数型配列 1）とキーボードから入力した 4 つの数値（整数型配列 2）の値は一致するが位置は異なる要素位置の数を数える

戻り値：一致した要素位置の数

[プログラム]

```
#define NUMS 4          /* 要素の数 */
```

```
int check(int *man) {  
    int i, j, ret;
```

```
    ret = 0;
```

```
    for(i = 0; i < NUMS; i++) {  
        for(j = 0; j < NUMS; j++) {  
            if ( (1) ) ret = 1;  
        }  
    }
```

```
    return ret;
```

```
}
```

```

int hit(int comp[], int man[]) {
    int i, ret;

    ret = 0;
    for(i = 0; i < NUMS; i++) {
        if ( (2) ) ret++;
    }
    return ret;
}

```

```

int blow(int comp[], int man[]) {
    int i, j, ret;

    ret = 0;
    for(i = 0; i < NUMS; i++) {
        for (j = 0; j < NUMS; j++) {
            if ( (3) ) ret++;
        }
    }
    return ret;
}

```

<設問 1> プログラム中の に入れるべき適切な式を解答群から選べ。

(1) の解答群

- | | |
|--|--|
| ア. <code>man[i] > man[j]</code> | イ. <code>man[i] < man[j]</code> |
| ウ. <code>i == j && man[i] == man[j]</code> | エ. <code>i != j && man[i] == man[j]</code> |

(2) の解答群

- | | |
|-------------------------------------|-------------------------------------|
| ア. <code>comp[i] == man[i]</code> | イ. <code>comp[i] != man[i]</code> |
| ウ. <code>comp[i] < man[i]</code> | エ. <code>comp[i] > man[i]</code> |

(3) の解答群

- | | |
|---|---|
| ア. <code>comp[i] > man[j]</code> | イ. <code>comp[i] < man[j]</code> |
| ウ. <code>i == j && comp[i] == man[j]</code> | エ. <code>i != j && comp[i] == man[j]</code> |

<設問 2> 関数 check は、点線で囲まれた繰り返し処理に無駄な部分がある。そこで、繰り返し処理を以下のように書き換える。書き換えるプログラム中の [] に入れるべき適切な式を解答群から選べ。

```
for(i = 0; i < [ (4) ]; i++) {  
    for (j = i + 1; j < [ (5) ]; j++) {  
        if ( [ (1) ] ) ret = 1;  
    }  
}
```

(4) , (5) の解答群

- | | |
|--------------------|------------------------|
| ア. NUMS - i | イ. NUMS - 1 |
| ウ. NUMS | エ. NUMS + 1 |
| オ. NUMS + i | カ. NUMS + i - 1 |

<設問 3> 次のキーボードから入力した値をチェックするための手順に関する記述中の [] に入れるべき適切な字句を解答群から選べ。

プログラムの説明 2.2 で「入力した数値が 4 つでなければエラーメッセージを表示し、2.1 へ戻る」の判断は、入力した 4 つの値を 4 桁の整数値と考え、ある範囲内に収まるかを調べることで対処できる。

この場合、重複した値があればエラーになることも考慮し、[(6)] より小さい、または [(7)] より大きい値であれば、エラーメッセージを表示する。

(6) , (7) の解答群

- | | |
|---------|---------|
| ア. 123 | イ. 1000 |
| ウ. 1111 | エ. 1234 |
| オ. 9876 | カ. 9999 |

<設問 4> このゲームの繰り返しを終了するための条件「4 つの数値の値と位置が一致する」となる場合、ヒットの数とブローの数の関係を解答群から選べ。

(8) の解答群

- ア. ヒットの数 が 0 でブローの数 が 0
- イ. ヒットの数 が 0 でブローの数 が 4
- ウ. ヒットの数 が 4 でブローの数 が 0
- エ. ヒットの数 が 4 でブローの数 が 4