次の問9から問13までの5問については，この中から1問を選択し，選択した問題については，答案用紙の選択欄の(選)をマークして解答してください。

なお，2問以上マークした場合には，はじめの1問について採点します。

問9　次のCプログラムの説明及びプログラムを読んで，設問1～3に答えよ。

〔プログラムの説明〕

文字列の中から，回文（palindrome）を探して表示する関数 find_palindrome である。回文とは，先頭から読んだ文字の並びと末尾から読んだ文字の並びが一致する文字の並びのことである。ただし，ここでは次の条件を満たすものとする。

- 文字列は英字（A～Z，a～z），数字（0～9），記号（!"#%&'()*+,-./:;<=>?[]^_{|}）及び空白文字から成る。
- 文字の並びを読むとき，英字の大文字と小文字は区別しない。
- 文字の並びを読むとき，記号及び空白文字は無視する。
- 回文は英数字で始まり英数字で終わる。ただし，英数字1文字は回文ではない。

本問のプログラムが表示する回文の例を表1に示す。No. 1で示す文字列において，文字の並び bc0cb は回文である。c0c も回文であるが，本問のプログラムでは，文字列の先頭に最も近い文字から始まるものを表示する。また，No. 2で示す文字列において，英字の大文字と小文字は区別しないので，文字の並び Bc0Cb は回文である。さらに，No. 3で示す文字列において，英字の大文字と小文字を区別せず，かつ，記号及び空白文字を無視するので，文字の並び B!c0 Cb は回文である。No. 4で示す文字列において，文字列の先頭に最も近い b を先頭文字とする文字の並び bc0cb と bc0cb1bc0cb は，いずれも回文である。しかし，本問のプログラムでは，先頭文字位置が同じ回文が複数あれば，長さが最も短いものを表示する。

表1　本問のプログラムが表示する回文の例

No.	文字列	表示する回文
1	abc0cbe	bc0cb
2	ABc0CbE	Bc0Cb
3	AB!c0 CbE	B!c0 Cb
4	abc0cb1bc0cbe	bc0cb

(1) 関数 find_palindrome の仕様は，次のとおりである。ここで，引数の値に誤りはないものとする。

機能： 文字列 text の中から，回文を探して表示する。文字列 text の中に複数の回文がある場合，先頭文字位置が文字列の先頭に最も近いものを表示する。さらに，先頭文字位置が同じ回文が複数あれば，長さが最も短いものを表示する。

引数： text 文字列

関数 find_palindrome は，関数 is_palindrome 及び関数 find_char を使用する。

(2) 関数 is_palindrome の仕様は，次のとおりである。ここで，引数の値に誤りはないものとする。

機能： 文字の並び chars が回文かどうかを判定する。

引数： chars 文字の並び

size 文字の並び chars の長さ

返却値：回文の場合は 1

回文でない場合は 0

(3) 関数 find_char の仕様は，次のとおりである。ここで，引数の値に誤りはないものとする。

機能： 文字列 str 中で，文字 ch が最初に現れる位置を求める。ただし，英字の大文字と小文字は区別しない。

引数： str 文字列

ch 文字

返却値：文字 ch が現れる場合は，それが最初に現れる位置へのポインタ

文字 ch が現れない場合は，空ポインタ

(4) プログラム中で使用しているライブラリ関数の概要は，次のとおりである。

isalnum(ch)：文字 ch が英数字のとき，0 以外の値を返し，それ以外のとき，0 を返す。

tolower(ch)：文字 ch が英大文字のとき，その文字に対応する英小文字を返し，それ以外のとき，文字 ch を返す。

putchar(ch)：文字 ch を表示する。

〔プログラム〕

(行番号)

```
 1 #include <stdio.h>
 2 #include <ctype.h>

 3 void find_palindrome(const char*);
 4 int is_palindrome(const char*, int);
 5 const char* find_char(const char*, char);

 6 void find_palindrome(const char* text) {
 7     int i;
 8     int psize;        /* 文字の並びの長さ */
 9     const char* ith;  /* 文字列 text の第 i 文字へのポインタ */
10     const char* hit;  /* 第 i 文字と一致した文字へのポインタ */
11     for (i = 0; text[i] != '\0'; i++) {
12         if (!isalnum(text[i])) {
13             continue;
14         }
15         ith = &text[i];
16         hit = find_char(ith + 1, *ith);
17         while (hit != NULL) {
18             psize = [    a    ];
19             if (is_palindrome(ith, psize)) {
20                 while (ith < hit + 1) {
21                     putchar(*ith);
22                     ith++;
23                 }
24                 putchar('\n');
25                 return;
26             }
27             hit = find_char(hit + 1, *ith);
28         }
29     }
30 }
```

```
31 int is_palindrome(const char* chars, int size) {
32     int l;
33     int r;
34     for (l = 0, r = size - 1; l < r; l++, r--) {
35         while (!isalnum(chars[l])) {
36             l++;
37         }
38         while (!isalnum(chars[r])) {
39             r--;
40         }
41         if ( [ b ] ) {
42             return 0;
43         }
44     }
45     return 1;
46 }

47 const char* find_char(const char* str, char ch) {
48     int i;
49     for (i = 0; str[i] != '\0'; i++) {
50         if ( [ c ] ) {
51             return &str[i];
52         }
53     }
54     return NULL;
55 }
```

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　i　　イ　ith - &text[0]　　ウ　ith - &text[0] + 1

エ　hit - ith　　オ　hit - ith + 1

bに関する解答群

ア　l == r

イ　l != r

ウ　chars[l] == chars[r]

エ　chars[l] != chars[r]

オ　tolower(chars[l]) == tolower(chars[r])

カ　tolower(chars[l]) != tolower(chars[r])

cに関する解答群

ア　ch == str[i]

イ　ch != str[i]

ウ　tolower(ch) == tolower(str[i])

エ　tolower(ch) != tolower(str[i])

オ　(ch == tolower(str[i])) || (tolower(ch) == str[i])

カ　(ch == tolower(str[i])) && (tolower(ch) == str[i])

設問2　関数 find_palindrome が，先頭文字位置が同じ回文が複数ある場合，長さが最も長いものを表示するように，プログラムを変更する。表 2 中の［　　］に入れる正しい答えを，解答群の中から選べ。

(1) 関数 find_char を，関数 find_last_char と置き換える。関数 find_last_char の仕様は，次のとおりである。ここで，引数の値に誤りはないものとする。

機能：　文字列 str の先頭から文字数 count 以内の範囲で文字 ch が現れる最後の位置を求める。ただし，英字の大文字と小文字は区別しない。

引数：　str　　文字列

ch　　文字

count　文字数

返却値：文字 ch が現れる場合は，それが最後に現れる位置へのポインタ

文字 ch が現れない場合は，空ポインタ

(2) 新たに使用するライブラリ関数 strlen(text)は，文字列 text の長さ（ただし，終端ナル文字'\0'は含まない）を返す。

表2 プログラムの変更内容

処置	変更内容
行番号2と3の間に追加	#include <string.h>
行番号5を変更	const char* find_last_char(const char*, char, int);
行番号10と11の間に追加	unsigned int textlen = strlen(text);
行番号16を変更	hit = find_last_char(ith + 1, *ith, textlen - i - 1);
行番号27を変更	hit = find_last_char(ith + 1, *ith, [d]);

行番号47～49を変更	const char* find_last_char(const char* str, char ch, int count) {
	int i;
	for ([e] ; i--) {

dに関する解答群

ア textlen - psize　　イ textlen - i - psize

ウ textlen - i + psize　　エ psize + 2

オ psize - 2

eに関する解答群

ア i = count; i > 0　　イ i = count; i >= 0

ウ i = count - 1; i > 0　　エ i = count - 1; i >= 0

設問3　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。

設問2で変更した関数 find_palindrome を，次の文字列を引数として実行した。ここで“␣”は空白文字を表す。

```
No!␣Madam,␣I'm␣Adam␣Graham.␣This␣is␣my␣gym.
```

関数 find_palindrome が回文の表示を終了するまでに，関数 find_last_char は［ f ］回呼び出され，その最後の呼出しにおける引数 count の値は［ g ］である。

fに関する解答群

ア　2　　イ　3　　ウ　4

エ　5　　オ　6　　カ　7

gに関する解答群

ア　13　　イ　20　　ウ　31

エ　36　　オ　38