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1，2に答えよ。

C

〔プログラムの説明〕

関数 check_winning_lot は，複数本のくじのくじ番号を当選番号と比較し，当たりを確認するプログラムである。

(1) くじ番号は，6桁の数字から成る文字列である。数字だけから成る文字列を数字列という。

(2) くじの当たりは1～6等の6等級から成り，図1に示すように各等級には一つ以上の当選番号がある。各等級での当選番号の桁数と個数は異なる。

等級	当選番号
1等	223692
2等	141421，314159
3等	下5桁　27182，00145
4等	下4桁　3301，1028
5等	下3桁　243，101，144
6等	下2桁　03，92

図1　当選番号の例

(3) 当選番号は6桁以下の数字列であり，くじ番号の最下位桁から当選番号の桁数だけ取り出した数字列が当選番号と一致したくじが当たりとなる。

なお，くじ番号によっては，複数の等級の当たりとなる場合もある。図1の例では，くじ番号“223692”のくじは，1等と6等の当たりとなる。

(4) 関数 check_winning_lot の引数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。

lots　　確認するくじのくじ番号表
win_list　当選番号表
winner　　当選本数表

(5) 確認するくじのくじ番号表 lots の要素 lots[i]には，くじ番号の数字列へのポインタが格納されている。最後の要素 lots[Nl]（Nl は，確認するくじの本数）には，空ポインタ定数（NULL）が格納されている。

(6) 当選番号表 win_list の要素 win_list[i]には，i+1 等の番号表へのポインタが格納されている。win_list[i]が指す i+1 等の番号表の要素 win_list[i][j]には，当選番号の数字列へのポインタが格納されている。最後の要素 win_list[i][Ni]（Ni は，i+1 等の当選番号の個数）には，図2に示すように空ポインタ定数が格納されている。

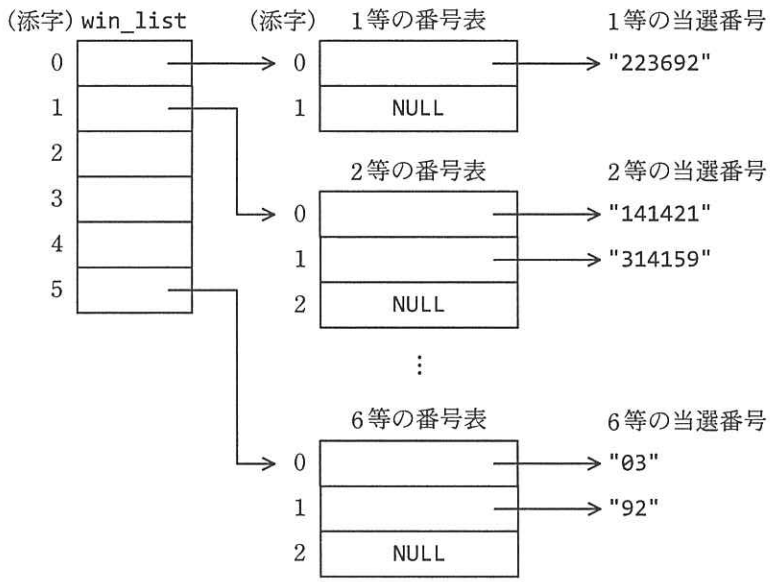

図2　当選番号表の例

(7) 当選本数表 winner の要素 winner[i]には，i+1 等の当たりのくじの本数が格納される。

(8) 関数 check_winning_lot で使用している関数 check_lot と strlen の仕様は次のとおりである。

```
int check_lot(char *lot, char *win);
```

機能：　くじ番号と当選番号を，最下位桁から1桁ずつ順に当選番号の桁数分だけ比較し，一致するかどうかを調べる。

引数：　lot　くじ番号

　　win　当選番号

返却値：確認結果（0：一致しない，1：一致する）

```
unsigned int strlen(const char *s);
```

機能：　文字列の長さを求める。

引数：　s　文字列

返却値：終端を示すナル文字に先行する文字の個数

〔プログラム〕

(行番号)

```
 1 #include <stdlib.h>
 2 #include <string.h>

 3 #define LOT_DNUM 6   /* くじの桁数 */
 4 #define LOT_GNUM 6   /* くじの等級数 */

 5 void check_winning_lot(char *[], char **[LOT_GNUM],
 6                         int [LOT_GNUM]);
 7 int check_lot(char *, char *);

 8 void check_winning_lot(char *lots[],
 9                         char **win_list[LOT_GNUM],
10                         int winner[LOT_GNUM]) {
11     int  i, j, k;

12     for (i = 0; i < LOT_GNUM; i++) {
13         winner[i] = 0;
14         for (j = 0; lots[j] != NULL; j++) {
15             for (k = 0;    a    ; k++) {
16                 winner[i] += check_lot(lots[j], win_list[i][k]);
17             }
18         }
19     }
20 }

21 int check_lot(char *lot, char *win) {
22     int  lenw, result = 1, i;

23     lenw = strlen(win);
24     win += lenw;
25     lot    b    ;
26     for (i = 0;    c    ; i++) {
27         if (*(--lot) != *(--win)) {
28             result = 0;
29         }
30     }
31     return result;
32 }
```

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　k < LOT_DNUM　　イ　k <= LOT_DNUM

ウ　win_list[i][k] == NULL　　エ　win_list[i][k] != NULL

オ　*win_list[i][k] == NULL　　カ　*win_list[i][k] != NULL

b に関する解答群

ア　= lenw　　イ　= LOT_DNUM

ウ　+= lenw　　エ　+= LOT_DNUM

c に関する解答群

ア　(i < lenw) && (result == 0)

イ　(i < lenw) && (result == 1)

ウ　(i < LOT_DNUM) && (result == 0)

エ　(i < LOT_DNUM) && (result == 1)

オ　result == 0

カ　result == 1

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。ただし，f1 と f2 に入れる答えは，f に関する解答群の中から組合せとして正しいものを選ぶものとする。

くじの当たりを増やすために，6 桁の当選番号に対して前後賞が用意されることになった。これに合わせて関数 check_winning_lot に，前後賞の当選結果を自動判定する処理を追加する。ここで，プログラム中の［ a ］～［ c ］には正しい答えが入っているものとする。

(1)　関数 check_winning_lot の引数を次のとおりに変更する。ここで，引数の値に誤りはないものとする。

lots　　確認するくじのくじ番号表

win_list　当選番号表

winner　　当選本数表

special　　前後賞の当選本数表

(2) 前後賞は，6桁の当選番号に対して用意される。例えば，当選番号“123456”に対する前後賞の番号は，“123455”と“123457”となる。当選番号“000000”に対する前後賞の番号は“999999”と“000001”，当選番号“999999”に対する前後賞の番号は“999998”と“000000”とする。

(3) 前後賞の当選結果の確認は，確認するくじのくじ番号を数値とみなし，それよりも1だけ少ない数値と多い数値を表す数字列を作成し，その数字列を当選番号と比較することによって行う。このとき，くじ番号“000000”よりも1だけ少ない数値を表す数字列は“999999”に，くじ番号“999999”よりも1だけ多い数値を表す数字列は“000000”にする。

(4) 前後賞の当選本数表の要素special[i]にはi+1等の前後賞の当たりのくじの本数が格納される。

(5) 処理の追加に対応するために，プログラムを表1のとおりに変更する。

表1　プログラムの変更内容

処置	変更内容
行番号5，6を変更	`void check_winning_lot(char *[], char **[LOT_GNUM],` `int [LOT_GNUM], int [LOT_GNUM]);`

行番号8～10を変更	`void check_winning_lot(char *lots[],`
	`char **win_list[LOT_GNUM],`
	`int winner[LOT_GNUM], int special[LOT_GNUM]) {`

行番号11と12の間に追加	`int pflg, nflg, m;` `char pnum[LOT_DNUM + 1], nnum[LOT_DNUM + 1];`
行番号13と14の間に追加	`special[i] = 0;`

行番号14と15の間に追加	`/* 前後賞の当選確認用数字列の作成 */`
	`pflg = nflg = 0;`
	`for (m = [ d ]; m >= 0; m--) {`
	`pnum[m] = nnum[m] = [ e ];`
	`if (pflg == 0) {`
	`if (pnum[m] == [ f1 ]) {`
	`pnum[m] = [ f2 ];`
	`} else {`
	`pnum[m]--;`
	`pflg = 1;`
	`}`

	}
	if (nflg == 0) {
	if (nnum[m] == [f2]) {
	nnum[m] = [f1];
	} else {
	nnum[m]++;
	nflg = 1;
	}
	}
	}
	pnum[LOT_DNUM] = nnum[LOT_DNUM] = '\0';

行番号 16 と 17 の間に追加	if (strlen(win_list[i][k]) == LOT_DNUM) {
	special[i] += check_lot(pnum, win_list[i][k]);
	special[i] += check_lot(nnum, win_list[i][k]);
	}

C

dに関する解答群

ア i	イ j	ウ LOT_DNUM
エ LOT_DNUM + 1	オ LOT_DNUM + i	カ LOT_DNUM + j
キ LOT_DNUM - 1	ク LOT_DNUM - i	ケ LOT_DNUM - j

eに関する解答群

ア *lots[j]	イ *lots[j] + m
ウ *lots[j] + m + 1	エ *(lots[j] + m)
オ *(lots[j] + m + 1)	

fに関する解答群

	f1	f2
ア	'0'	'1'
イ	'0'	'9'
ウ	'1'	'0'
エ	'1'	'9'
オ	'9'	'0'
カ	'9'	'1'