

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問以上選択した場合には、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。

[プログラムの説明]

関数 `printout_text` は、ファイルから英文テキストを読み込んで、単語が行末の改行で切れないように整形して出力するプログラムである。

(1) 関数 `printout_text` の引数は、次のとおりである。ここで、ファイル名に誤りはないものとする。

`filename`            英文テキストが格納されているファイル名

(2) 英文テキストに含まれる文字は、次のものである。

- ① 英字    `A ~ Z, a ~ z`
- ② 数字    `0 ~ 9`
- ③ 記号    `! " # $ % & ' ( ) * + , - . / : ; < = > ? [ ] ^ _ { | } ~`
- ④ 空白文字
- ⑤ 改行文字

(3) 出力する1行の最大文字数(改行文字を含まない)は80文字である。

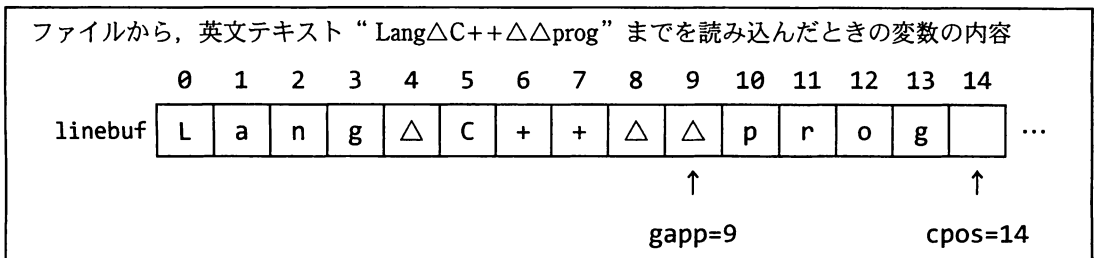
(4) 単語は空白文字及び改行文字を含まない文字列であり、1文字以上の連続する空白文字又は改行文字で区切られている。

(5) 空白文字及び改行文字はそのまま出力する。

(6) 単語の途中で1行の最大文字数を超える場合は、その単語が次の行の先頭になるように出力する。

(7) 単語の長さは、1行の最大文字数を超えない。

(8) プログラム中で使われる変数 `cpos` は出力用行バッファ `linebuf` に次の文字を格納する位置を、変数 `gapp` は `linebuf` に最後に格納した空白文字の位置をもつ。変数 `cpos`, `gapp` の例を、図1に示す。



注 “△” は空白文字を表す。

図 1 変数 cpos, gapp の例

[プログラム]

(行番号)

```

1 #include <stdio.h>

2 #define COLUMNS 80 /* 1行の最大文字数 */

3 void printout_text(char *);

4 void printout_text(char *filename) {
5     FILE *fp;
6     char linebuf[COLUMNS + 1];
7     int ch, ch1, cpos = 0, gapp = -1, i;

8     fp = fopen(filename, "r");
9     while ((ch = fgetc(fp)) != EOF) {
10        switch (ch) {
11            case '\n':
12                linebuf[cpos] = '\0';
13                printf("%s\n", linebuf);
14                cpos = 0;
15                gapp = -1;
16                break;
17            case ' ':
18                if (a) {
19                    linebuf[cpos] = '\0';
20                    printf("%s\n", linebuf);
21                    cpos = 0;
22                }
23                b ;
24                c = ch;
25                break;

```

```

26         default:
27             if (  ) {
28                 i = gapp + 1;
29                 ch1 = linebuf[i];
30                 linebuf[i] = '\0';
31                 printf("%s\n", linebuf);
32                 cpos = 0;
33                 if (i < COLUMNS) {
34                     linebuf[cpos++] = ch1;
35                     ;
36                     while (i < COLUMNS) {
37                         linebuf[cpos++] = linebuf[i++];
38                     }
39                 }
40                 gapp = -1;
41             }
42              = ch;
43         } /* switch 文の終わり */
44     }
45     if (cpos > 0) {
46         linebuf[cpos] = '\0';
47         printf("%s", linebuf);
48     }
49     fclose(fp);
50 }

```

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 cpos < COLUMNS | イ cpos <= COLUMNS |
| ウ cpos > COLUMNS | エ cpos >= COLUMNS |
| オ gapp < COLUMNS | カ gapp <= COLUMNS |
| キ gapp > COLUMNS | ク gapp >= COLUMNS |

bに関する解答群

- |                  |               |
|------------------|---------------|
| ア gapp = -1      | イ gapp = cpos |
| ウ gapp = COLUMNS | エ gapp++      |
| オ gapp--         |               |

cに関する解答群

- |   |                 |   |                 |
|---|-----------------|---|-----------------|
| ア | linebuf[cpos]   | イ | linebuf[cpos++] |
| ウ | linebuf[gapp]   | エ | linebuf[gapp++] |
| オ | linebuf[++cpos] | カ | linebuf[++gapp] |

dに関する解答群

- |   |        |   |          |   |             |
|---|--------|---|----------|---|-------------|
| ア | i = -1 | イ | i = cpos | ウ | i = COLUMNS |
| エ | i++    | オ | i--      |   |             |

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。

入力した英文テキスト（以下、入力英文テキストという）中に複数回出現する単語をキーワードという。最初の出現場所でキーワードとする単語と参照番号を定義し、それ以降の出現場所では参照番号で参照するように入力英文テキストの様式を変更した。入力英文テキストを整形して出力するときには、参照番号で示されているキーワードは、本来の文字列に戻して出力するようにプログラムを変更する。ここで、プログラム中の  a  ～  d  には、正しい答えが入っているものとする。

(1) 入力英文テキスト中でのキーワードの表現様式は図2のとおりである。

なお、キーワードの長さは20文字以下であり、キーワードの個数は50以下である。また、入力英文テキストに含まれる文字として文字“\”を追加する。追加された文字“\”は、キーワードの定義及び参照以外に使われることはない。

最初の出現場所でのキーワードと参照番号の定義様式：

\D 参照番号\単語\

参照番号 キーワードの参照番号（0～49）

単語 キーワードとする単語

キーワードの参照様式：

\R 参照番号\

参照番号 キーワードの参照番号（0～49）

図2 キーワードの定義様式と参照様式

(2) 入力英文テキストを、次の規則で、出力する英文テキスト（以下、出力英文テキストという）に変換しながら、キーワードを含め単語が行末の改行で切れないように整形して出力する。

- ① キーワードと参照番号を定義している部分については、定義しているキーワードだけを出力英文テキストに残す。変換例を図3に示す。

入力英文テキスト	Lang△\D2\C++\△△prog
出力英文テキスト	Lang△C++△△prog

注 “△” は空白文字を表す。

図3 キーワードと参照番号を定義している部分の変換例

- ② キーワードを参照している部分については、参照番号に対応するキーワードで置き換える。変換例を図4に示す。

入力英文テキスト	Develop△△\R2\△△compiler
出力英文テキスト	Develop△△C++△△compiler

注 “△” は空白文字を表す。

図4 キーワードを参照している部分の変換例

- ③ その他の部分は、〔プログラムの説明〕の(1)～(8)に従う。

(3) キーワード処理に対応するために、プログラムを表のとおりに変更する。

表 プログラムの変更内容

処置	プログラムの変更内容
行番号1と2の間に追加	#include <string.h>
行番号2と3の間に追加	#define WORDL 20 /* キーワードの最大文字数 */ #define REFMX 50 /* キーワードの最大個数 */
行番号7と8の間に追加	char reftbl[REFMX][WORDL + 1]; int ch2, j;
行番号25と26の間に追加	〔プログラムの追加部分〕

〔プログラムの追加部分〕で使用している関数 `strlen` の仕様は次のとおりである。

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

機能：文字列 `s` の長さを求める。

返却値：終端を示すナル文字に先行する文字の個数。

〔プログラムの追加部分〕

```
case '\\':
    ch1 = fgetc(fp);
    i = 0;
    while ((ch2 = fgetc(fp)) != '\\') {
        i = ;
    }
    if () {
        j = 0;
        while ((reftbl[i][j++] = fgetc(fp)) != '\\');
        reftbl[i][j - 1] = '\\0';
    }
    j = strlen(reftbl[i]);
    if () {
        linebuf[cpos] = '\\0';
        printf("%s\n", linebuf);
        cpos = 0;
        gapp = -1;
    }
    strcpy(&linebuf[cpos], reftbl[i]);
    cpos += j;
    break;
```

eに関する解答群

- |                |                      |
|----------------|----------------------|
| ア ch2          | イ ch2 - '0'          |
| ウ i + ch2      | エ i + ch2 - '0'      |
| オ i * 10 + ch2 | カ i * 10 + ch2 - '0' |

fに関する解答群

- |              |              |               |
|--------------|--------------|---------------|
| ア ch1 == 'D' | イ ch1 == 'R' | ウ ch1 == '\\' |
| エ i == 0     | オ i != 0     |               |

gに関する解答群

ア  $cpos \geq COLUMNS$

ウ  $cpos + j - 1 \geq COLUMNS$

オ  $gapp + j - 1 \geq COLUMNS$

イ  $cpos + j \geq COLUMNS$

エ  $gapp \geq COLUMNS$