```c
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>

#define WORD_LEN 20

struct term {
   char word[WORD_LEN];
   int freq[10];
   int df;
   struct term* next;
};

struct term* head = NULL;
int term_cnt = 0;

struct tfidf {
   char *word;
   double tfidf;
};

void add_term(const char* s, int fno) {
   struct term* node;
   int i;

   node = malloc(sizeof(struct term));
   strcpy(node->word, s);
   for (i = 0; i <= 9; i++)
      node->freq[i] = 0;
   node->freq[fno] = i;

   node->df = 0;
   node->next = head;
   head = node;
   term_cnt++;
}

struct term* find_term(const char* s) {
   struct term* p;

   p = head;
   while(p != NULL) {
      if (strcmp(p->word, s) == 0) {
```

```
            return p;
        } else {
            p = p->next;
        }
    }
    return NULL;
};

void print_term_list() {
    struct term* p;

    p = head;
    while (p != NULL) {
        int i;

        printf("%s ", p->word);
        for (i = 0; i <= 9; i++)
            printf("%d ", p->freq[i]);
        printf("(%d)", p->df);
        p = p->next;
        printf("\n");
    }
}

void read_word(FILE* fp, char s[WORD_LEN]) {
    int c;
    int i = 0;
    while ((c = fgetc(fp)) != EOF) {
        if (isalpha(c)) {
            if (i < WORD_LEN - 1) {
                s[i] = tolower(c);
                i++;
            } else {
                s[i] = '\0';
                break;
            }
        } else {
            s[i] = '\0';
            return;
        }
    }
}

void compute_df() {
    struct term* ptr;
    int i;
```

```
    for (ptr = head; ptr != NULL; ptr = ptr->next) {
        for (i = 0; i < 10; i++) {
            if (ptr->freq[i] > 0)
                ptr->df++;
        }
    }
}

void compute_tfidf(struct tfidf arr[], int fno) {
    // fno번 문서에 대해 arr[]의 내용을 채움.
}

void print_tfidf(struct tfidf arr[]) {
    //arr[]의 내용 출력 (arr[0]~arr[4])
}

int main() {
    FILE* fp;
    char word[WORD_LEN];
    char fname[20];

    for (int i = 0; i <= 9; i++) {
        sprintf(fname, "doc\\%d.txt", i);
        fp = fopen(fname, "r");

        while (!feof(fp) {
            struct term* p;
            read_word(fp, word);
            if (strlen(word) == 0)
                continue;

            p = find_term(word);
            if (p == NULL)
                add_term(word, i);
            else
                p->freq[i]++;
        }
        fclose(fp);
    }
    compute_df();
    print_term_list();

    struct tfidf arr[term_cnt];
    for (int i=0; i<=9; i++) {
        printf("%d.txt: ", i);
```

```
        print_tfidf(arr);
    }

    return 0;
}
```