과제 1번

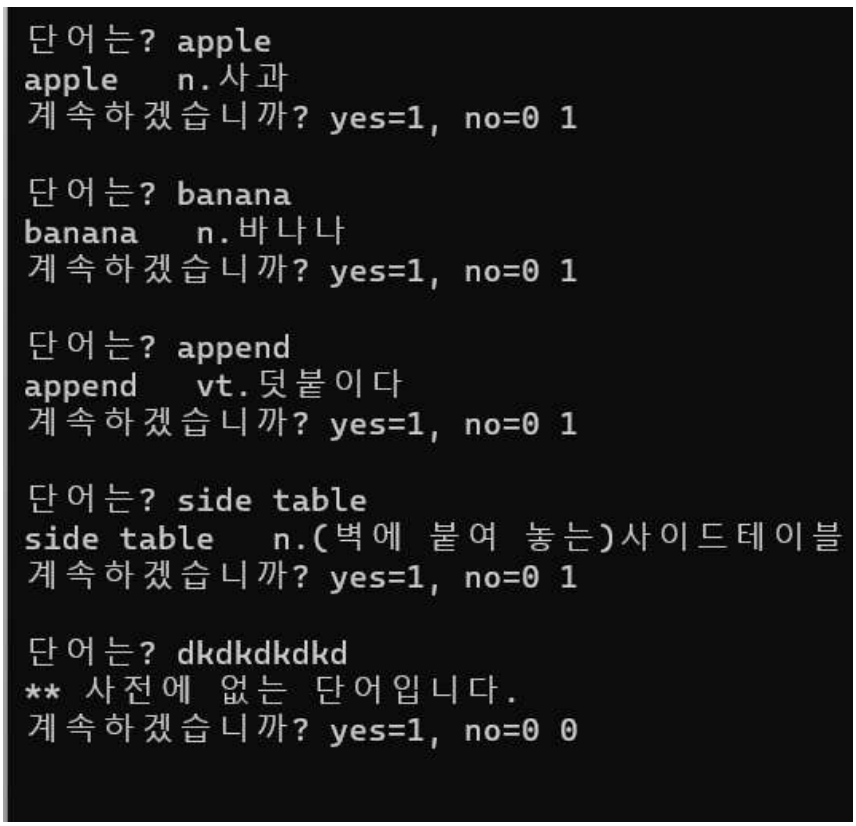
< 사용 변수 >

- 구조체 DICTIONARY에는 영단어에 해당하는 char *형 변수 word와 의미에 해당하는 char* 형 변수 meaning이 있다.
- main 함수에서는 먼저 struct DICTIONARY 자료형의 wordlist 변수를 [52000]개 배열로 선언했다. File의 포인터인 pFile 변수와, char * 자료형의 token과 search, 사용자가 찾고자 하는 단어인 char[30] 자료형의 str1 변수가 있다. char[120] 자료형의 str3 변수는 fgets를 사용해 한 줄을 얻어올 때 잠시 저장해두는 변수이다. check은 char 형으로, 단어가 있는지 확인하는 변수이다. 마지막으로 int 형 i는 저장 개수를 나타내는 변수이다.

< 풀이 방법 >

[main 함수]

- 먼저 구조체에 단어와 해석을 저장한다. 이 때, strtok와 strcpy를 사용한다. 매번 malloc을 사용하는 것은 heap 오류를 일으키기에 calloc으로 미리 할당받은 후 사용한다. 그러기 위해 strlen을 사용해 포인터의 위치를 옮기는 과정을 거쳤다.
- 사용자가 찾고자 하는 단어를 fgets를 통해 입력받은 후, for문을 사용해 해당 단어가 있는지를 확인한다. 있다면 check는 1이 되고, 그 뜻을 출력한다. 만약 단어가 없는 경우, check는 여전히 0일 것이기에 없는 단어임을 출력한다.
- 마지막으로 계속 검색할 것인지 check를 통해 확인한다.



과제 2번

< 사용 변수 >

- File의 포인터인 pFile 변수와, 사용자 입력 단어인 str3가 있다. str3는 char [50] 자료형이다. str2는 str3를 잠시 저장해 두는 배열로, char[50] 자료형이며, line의 경우, char[80] 자료형 변수로, 사전 데이터의 한 줄 한 줄을 확인할 때 사용하는 변수이다. char *[100] 자료형의 past는 이전에 적은 단어를 기억할 때 사용되고, char* ptr 변수는 컴퓨터의 입력 내용에 대한 포인터 변수이다. char 자료형의 check는 조건 만족 여부를, miss는 시도 실패 횟수, score은 점수, round는 라운드를, wrong은 실제 있는 단어인지 확인할 때 사용된다.

< 풀이 방법 >

[main 함수]

- miss가 3이기 전까지만 반복한다. 파일을 불러온 후, 첫 번째 round에서만 컴퓨터의 입력을 확인한다. 이후, 사용자의 입력을 받은 후, 1) 끝말잇기를 제대로 수행했는지, 2) 단어 길이가 적당한지 3) 이미 입력한 단어인지, 4) 사전에 있는 단어인지를 확인한다. check를 사용해 각 조건을 모두 충족하는지 확인한다.
- 사용자 단어 입력과 동시에 past 배열에 입력 값을 저장해, 이후 라운드에서 이미 입력한 내용을 다시 입력했는지 확인하는 데 사용한다. 이때 malloc을 사용해 문자열을 저장한다.
- 만약 check 값이 기준 (round가 1일 때 제외하고는 3)에 미치지 못한다면 miss로 간주한다.

```
1 라운드
컴퓨터: abstinent
단어는? tall
현재점수 1 점

2 라운드
단어는? ld
단어의 길이가 잘못되었습니다.
없는 단어입니다.
현재점수 1 점

3 라운드
단어는? dance
현재점수 2 점

4 라운드
단어는? top
잘못된 끝말잇기입니다.
현재점수 2 점

5 라운드
단어는? point
현재점수 3 점

6 라운드
단어는? tall
이미 나온 단어입니다.
현재점수 3 점
```

과제 3번

< 사용 변수 >

- 구조체 INFO 안에는 길이에 해당하는 length, 모음 개수에 해당하는 moeum, 그리고 char * 자료형의 word가 존재한다. word의 경우, 동적할당을 사용할 것이기에 포인터 자료형으로 만들었다.
- main 함수에서 struct INFO의 info 구조체 변수를 [52000]개 정도의 배열로 선언했고, str3은 fgets를 위한 char[50] 자료형이고, char *의 token은 strtok를 위한 변수이다. int i는 자료의 개수를, pFile은 파일 자료형의 변수이다.

< 풀이 방법 >

[main 함수]

- 먼저 파일을 읽기 전용으로 열고, while문과 fgets, strtok를 사용해 영단어만을 추출해 낸다. 이 때, malloc을 사용해 단어의 길이만큼 info[i].word에 동적할당한다. strlen을 사용해 단어의 길이를 info[i].length에 저장하고, if문을 사용해 모음의 개수도 확인한다.
- qsort 함수를 사용해 단어 길이가 긴 순서대로 먼저 정렬한다. 이 때, compare 함수는 info[n]->length를 비교한다. 10등까지를 출력한 후, 다시 qsort를 사용해 모음이 많은 순서대로 10개를 출력한다. 이 때 compare2 함수는 info[n]->moeum을 비교한다. 10등까지 출력한다.

- compare/compare2 함수에서 단순히 return에 수식을 두었더니 제대로 출력되지 않았다. 아마 동일 순위가 많아서 인듯 하다. return 0/-1/1; 을 명시적으로 제시하면 제대로 작동한다.