숙제: TRIE 및 이를 이용한 spelling correction 시스템 개발

• 개발 세부 사항: 프로그램은 다음 단계 순서로 수행하도록 작성한다.

(1) 모든 레코드 저장 단계:
-화일 "Corpus_dictionary_AP_Penn_RARE.txt"는 각 줄마다 word, windex, wfrequency 정보가 있다 (빈 칸으로 구분됨).
-각 줄의 정보를 읽어 한 레코드로 만든다. 여기에서 레코드의 키(key)는 word 로 한다.
-이 레코드를 Trie에 저장한다. key는 trie에 저장하고 해당 레코드는 master 화일(binary file)에 저장한다.
-Trie에서 key의 마지막 노드 ('₩0'를 가진 노드)는 데이타 화일의 해당 레코드 위치 (byte address)를 가지고 있도록 한다.
-이 과정에서는 수업 중에 설명한 다음 함수들을 이용하도록 한다:
hang_down, find_trie, insert_trie

(2) 탐색 실험 단계:
-탐색할 키(여기서는 한 영어 단어)를 입력 받아 탐색을 수행하는 작업을 반복한다.
-탐색 과정에서 spell correction 작업을 수행하면서 탐색한다. 그 결과 매우 많은 교정 후보를 생성한다.
발견된 후보는 교정결과 테이블 cwords에 저장한다.
-만약 TRIE에 저장된 한 키가 입력 단어와 완전매칭 되는 것이 있다면 이것 하나만 답으로 출력한다.
주: 출력은 단어만 출력하도로 한다 (아래 실행 예 참고.)
-그렇지 않다면 벌점(penalty)이 작은 순서로 상위 10 개의 교정 결과를 출력한다.
주: 출력은 각 교정 결과와 해당 벌점만 출력하도로 한다 (아래 실행 예 참고.)
-벌점 주기: 각 종류마다 오류 1 개 당 다음의 벌점을 주도록 한다.
substitution: 1.1; deletion: 1.3; insertion: 1.6; transposition: 1.9.
-탐색 과정에서 2 까지의 교정만 시도한다. 더 이상의 교정은 수행하지 않도록 한다.
(3개까지 시도하는 경우 교정결과 테이블의 크기를 더 많이 키워야 함.)
- $ 가 탐색할 단어이면 프로그램을 종료한다.

• 수행 예:
(a) "helpless" 탐색

```
TYPE A KEY: helpless
total number of corrected results = 1

< 0>: helpless penalty:  0.000, substi:0,  delet:0,  inser:0,  transp:0
```

(b) "hld", "hlod" 탐색

```
TYPE A KEY: hld
total number of corrected results = 10045

< 0>: had penalty:  1.100, substi:1,  delet:0,  inser:0,  transp:0
< 1>: hid penalty:  1.100, substi:1,  delet:0,  inser:0,  transp:0
< 2>: Old penalty:  1.100, substi:1,  delet:0,  inser:0,  transp:0
< 3>: old penalty:  1.100, substi:1,  delet:0,  inser:0,  transp:0
< 4>: held penalty:  1.300, substi:0,  delet:1,  inser:0,  transp:0
< 5>: hold penalty:  1.300, substi:0,  delet:1,  inser:0,  transp:0
< 6>: hae penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 7>: ham penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 8>: han penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 9>: has penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
```

```
TYPE A KEY: hlod
total number of corrected results = 4422

< 0>: hood penalty:  1.100, substi:1,  delet:0,  inser:0,  transp:0
< 1>: clod penalty:  1.100, substi:1,  delet:0,  inser:0,  transp:0
< 2>: plod penalty:  1.100, substi:1,  delet:0,  inser:0,  transp:0
< 3>: hold penalty:  1.900, substi:0,  delet:0,  inser:0,  transp:1
< 4>: hand penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 5>: hard penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 6>: he'd penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 7>: head penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 8>: heed penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 9>: held penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
```

(c) "hapy" 탐색

```
TYPE A KEY: hapy
total number of corrected results = 5428

< 0>: hazy penalty:  1.100, substi:1,  delet:0,  inser:0,  transp:0
< 1>: happy penalty:  1.300, substi:0,  delet:1,  inser:0,  transp:0
< 2>: hay penalty:  1.600, substi:0,  delet:0,  inser:1,  transp:0
< 3>: hace penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 4>: hack penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 5>: hail penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 6>: hair penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 7>: hajj penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 8>: half penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 9>: hall penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
```

(d) "grateful", "rateful" 탐색

```
TYPE A KEY: grateful
total number of corrected results = 1

< 0>: grateful penalty:  0.000, substi:0,  delet:0,  inser:0,  transp:0
```

```
TYPE A KEY: rateful
total number of corrected results = 371

< 0>: fateful penalty:  1.100, substi:1,  delet:0,  inser:0,  transp:0
< 1>: hateful penalty:  1.100, substi:1,  delet:0,  inser:0,  transp:0
< 2>: Grateful penalty:  1.300, substi:0,  delet:1,  inser:0,  transp:0
< 3>: grateful penalty:  1.300, substi:0,  delet:1,  inser:0,  transp:0
< 4>: Careful penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 5>: careful penalty:  2.200, substi:2,  delet:0,  inser:0,  transp:0
< 6>: tasteful penalty:  2.400, substi:1,  delet:1,  inser:0,  transp:0
< 7>: wasteful penalty:  2.400, substi:1,  delet:1,  inser:0,  transp:0
< 8>: graceful penalty:  2.400, substi:1,  delet:1,  inser:0,  transp:0
< 9>: wrathful penalty:  2.400, substi:1,  delet:1,  inser:0,  transp:0
```

• 제출물: 프로그램(.c 화일 1개로 할 것), 실행 결과를 보여주는 실행창 화면.

*** **가이드파일과 함께 빈 부분을 수정하여 제출하도록 한다.**

• 참고 사항:

```c
#define penalty_substitution 1.1    // 이하 4 줄은 오류의 벌점임.
#define penalty_deletion 1.3
#define penalty_insertion 1.6
#define penalty_transposition 1.9
#define Maximum_corrections 50000   // 교정 결과 테이블 크기
#define Max_error_count 2           // 이 갯수의 오류 처리까지만 수행함.

typedef struct node *nodeptr;
typedef struct node {
    char ch; // character in this node
    long int bp;  // a byte position in the master file
    nodeptr right ;  // a horizontal pointer
    nodeptr below ;  // a vertical poiner
} nodetype;

typedef struct record {
   char word[50];
   int  widx;
   int  wfreq;
} ty_rec ;

typedef struct correct_result {
   char word [100]; // the result of correction
   int ns, nd, ni, nt; // number of substitution, deletion, insertion, transposistion
   float  penalty; // penalty score
} ty_correction;

ty_correction cwords [Maximum_corrections];
int nres = 0;
int found_perfect_match = 0;
nodeptr ROOT_TRIE = NULL; // Pointer to ROOT_TRIE node of the total trie.
```