【HW-04】Term Counting by Linked List

解題紀錄

題目敘述

簡介 (Intro)

鏈結串列 (Linked List) 是一種線性資料結構 (Linear Data Structure)，
其透過節點 (Node) 間的「串接」組合而成 (i.e., 節點內有「指向下一個節點」的指標)。

不同於陣列是連續的記憶體分配 (Contiguously Allocated)，
節點「通常」僅在需要時，由 **malloc** 等函式分配空間。

作業 (Homework)

請由標準輸入讀入單詞，建構對應的鏈結串列 (Linked List)，
並以此統計單詞數量 (Term Counting)，例如輸入：

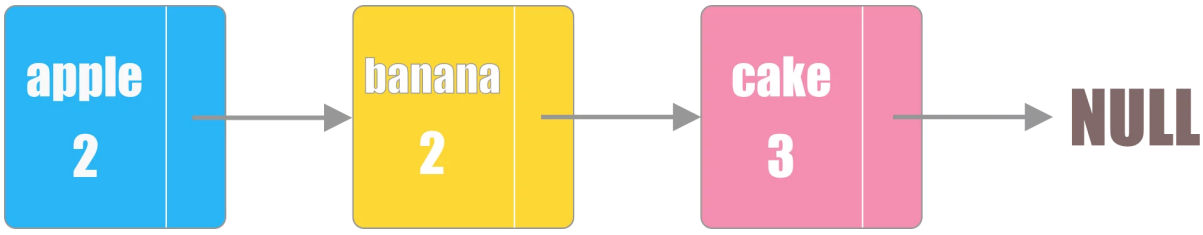
```
apple
banana
apple
apple
banana
cake
cake
-apple
cake
```

輸出 (順序根據單詞出現時間印出)：

```
2 apple
```

2 banana
3 cake

圖例：



[註]：
儘管此次作業並無要求依詞頻次數排序節點，
仍請已完成此次作業之同學進行該練習。

實作細節 (Implementation details)

節點 (Node) 僅是邏輯上的意涵，不代表一定要一個「struct Node」，
因此，你可以自行實作串列 **節點/資料的表示法**，例如：

資料本身即是「節點」

```
1  struct term {
2      char *str;
3      size_t cnt;
4      struct term *next;
5  };
```

另外有個「struct Node」搭配 void pointer 指向你的資料，也是常見的做法：

```
1  struct node {
2      void *data;
3      struct node *next;
4  };
5
6  struct term {
7      char *str;
8      int count;
9  };
```

助教我個人偏愛的則是：「將節點塞入資料中」

```
struct node {
    struct node *next;
};

struct term {
    char *str;
    int cnt;
    struct node entry;
};
```

若你習慣寫 C++, Java... 等語言，你可能會想要定義一個「struct list」來幫你管理串列：

```
struct term {
    char *str;
    int cnt;
};

struct node {
    struct term term;
    struct node *next;
};

struct list {
    struct node *head;
    size_t capacity;
    size_t size;
};
```

除了這些表示法的差異外，鏈結串列仍有相當多種變形，
作業**不限定**鏈結串列的種類，除了單向鏈結串列以外，
你也可以自行實作雙向、環狀、Dummy Node...etc.

提示 (Hint)

1.
鏈結串列**不像**陣列擁有**隨機存取** (i.e., 利用中括號以及索引 找到目標元素)，
欲找到「目標節點 」須搭配 **迴圈** 來逐個尋訪，例如：

```
Term *ptr = head;
while (ptr) {
```

```
3       printf("%s: %zu\n", ptr->str, ptr->cnt);
4       ptr = ptr->next;
5   }
```

必要時，也可能需要**兩個**指標來進行迴圈的迭代，
以達到「尋找最後一個節點」等需求 (當然，若你使用指標的指標，就不需要了)。

2.
節點「通常」會由 malloc 分配位於 **Heap** 的記憶體空間，
請謹記：**有 malloc 就要有 free**，
完成輸出後，請確實釋放對應的記憶體空間。

3.
使用 fgets 讀取輸入時，時常需要去除掉結尾的換行字元 (\n)，
然而它不一定會存在，因此使用 strlen 的方式來去除時，
請記得加上判斷：

```
1  size_t len = strlen(str);
2  if (str[len - 1] == '\n')
3      str[len - 1] = '\0';
```

你也可以使用其他方式：

```
1  /* 法二 */
2  char *ptr = strchr(str, '\n');
3  if (ptr) *ptr = '\0';
4
5  /* 法三 */
6  str[strcspn(str, "\n")] = '\0';
```

輸入說明

每行一個單詞，長度不超過 1024，

開頭若有「減號」則該詞次數減一

輸出說明

照單詞出現順序，印出其出現次數，冒號後有一個空白，參考 format：

```
"%s: %d\n"
```

輸入/輸出範例 1

執行參數
無

輸入

```
apple
banana
apple
apple
banana
cake
cake
-apple
cake
```

輸出

```
apple: 2
banana: 2
cake: 3

```

輸入/輸出範例 2

執行參數
無

輸入

```
Avenged Sevenfold
Extreme
Trivium
Guns N' Roses
Green Day
Alter Bridge
Avenged Sevenfold
-Green Day
Avenged Sevenfold
Guns N' Roses
```

```
10  Guns·N'·Roses
11  -Trivium
12  Alter·Bridge
13  Trivium
14  Extreme
15  Trivium
16
```

輸出

```
1  Avenged·Sevenfold:·3
2  Extreme:·2
3  Trivium:·2
4  Guns·N'·Roses:·2
5  Green·Day:·0
6  Alter·Bridge:·2
7
```

作答限制

基礎限制

執行時間上限	記憶體上限	開檔上限
1 秒	32 MB	0 個
指標	陣列	全域變數
✓	✓	✓

作業已過繳交期限，查看解題紀錄