Lab 03-02-2020

Submission deadline: By the end of the lab on `March 4, 2020`

Problem:

From the Canvas File page, please download the File `veggies.zip`, and unzip it. You should now have a directory named veggies. Use `cd` to go into this directory. You should Find a Makefile, the program `reader.c`, and two text files named `veggieking.txt`, and `vegetables.txt`. First, use the command `less` or your preferred editor to view the contents of `veggieking.txt` and `vegetables.txt`. Then, use make to compile the reader program. This is a starting program that will read the file whose name is specified on the command line. What it does is read each string in the file and clean it, removing any punctuation and making all characters lower case. It then prints out each cleaned string on a new line. Try running

```
reader veggieking.txt
```

to see what its output looks like.

Your job is to modify the program so that instead of printing out each string in the file, it counts the number of each of occurrence of the following vegetable names that are found in the file: carrot, potato, spinach, cauliflower, broccoli, eggplant. The program should read the input file, and then print out how many of each of these vegetable names occur in the file. For example, a run and its output should look something like this:

```
reader vegtables.txt
```

```
1 carrot
4 potato
1 spinach
1 cauliflower
1 broccoli
1 eggplant
```

Run your program on `veggieking.txt` to see if it works correctly. Then run it again on `vegetables.txt` to make sure that it works for any input filename.

Submission:

Please `zip` your `Makefile`, `reader.c`, `vegetables.txt`, and `veggieking.txt` and submit the zip file.

Useful data structure and functions:

- `int tolower(int argument)`: Convert a character to lower case. The character is stored in integer form in C programming. It is defined in `ctype.h` header file.

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

int main()
{
    char c = 'H';
    printf("%c", tolower(c));
    return 0;
}

```

- `int strcmp (const char* str1, const char* str2)`: The `strcmp()` function compares two strings and returns 0 if both strings are identical. It is defined in the `string.h` header file.

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

int main()
{
    char* str1 = "tiger";
    char* str2 = "tiger";
    char* str3 = "cat";
    int result;

    // comparing strings str1 and str2
    result = strcmp(str1, str2);
    printf("strcmp(str1, str2) = %d\n", result);

    // comparing strings str1 and str3
    result = strcmp(str1, str3);
    printf("strcmp(str1, str3) = %d\n", result);

    return 0;
}
```

- `int isalpha(int argument)`: The isalpha() function checks whether a character is an alphabet or not. If a character passed to isalpha() is an alphabet, it returns a **non-zero** integer, if not it returns 0.

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

int main()
{
    char c;
    c = 'A';
    printf("\nResult when uppercase alphabet is passed: %d", isalpha(c));

    c = 'a';
    printf("\nResult when lowercase alphabet is passed: %d", isalpha(c));

    c='+';
    printf("\nResult when non-alphabetic character is passed: %d",
isalpha(c));
```

```
15
16      return 0;
17  }
```

- `typedef` and `enum`: This combination provides a convenient way to restrict the value scope of a variable and avoid typing `enum` every time when declaring variables. Specifically, `enum` is mainly used to assign names to integral constants, the names make a program easy to read and maintain.

```c
#include <stdio.h>

int main()
{
    typedef enum Color_ {RED, YELLOW, BLUE} Color;
    Color pen_colors[100];

    pen_colors[0] = RED;
    pen_colors[1] = BLUE;
    pen_colors[2] = RED;
    pen_colors[3] = YELLOW;

    char* color_map[3] = {"red", "yellow", "blue"};  // map integer to string

    for(int i=0; i<4; i++){
        printf("The pen %d has the color: %s. \n", i, color_map[pen_colors[i]]);
    }

    return 0;
}
```