Washington State University Vancouver

Systems Programming - CS 360

# Assignment 2 - Due: 11:09AM Feburary 11

*Instructor:*
Ben McCamish

February 4, 2019

# Overall Assignment - 100 points

Write a program (in C) targeted at the Linux platform which checks whether a specified word, passed as an argument to the executable, exists in a dictionary or not. The specified dictionary is *webster* and on the website.

Example of webster dictionary format:

| line | offset | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8143 | 130288 | g | o | u | r | m | e | t | | | | | | | | | \n |
| 8144 | 130304 | g | o | u | t | | | | | | | | | | | | \n |
| 8145 | 130320 | g | o | v | e | r | n | | | | | | | | | | \n |
| 8146 | 130336 | **g** | **o** | **v** | **e** | **r** | **n** | **a** | **n** | **c** | **e** | | | | | | **\n** |
| 8147 | 130352 | g | o | v | e | r | n | e | s | s | | | | | | | \n |
| 8148 | 130368 | g | o | v | e | r | n | o | r | | | | | | | | \n |
| 8149 | 130384 | g | o | w | n | | | | | | | | | | | | \n |
| 8150 | 130400 | g | r | a | b | | | | | | | | | | | | \n |

- Format is 1 word per line
- Lines are in ascending sorted order
- Each line is 16 characters long
- Use binary search

Marks are shown below. Any requirement marked with "(Required)" will result in a score of 0 if not implemented.

# Program Interface (Required)

```
./ok search_term

Where: search_term is the desired word you are searching for.
The program should return 'yes' if the word exists in the dictionary
or 'no' if it does not.
```

Example dictionary 'tiny' (included in website):

| line | offset | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | a | a | r | d | v | a | r | k | | | | | | | | \n |
| 1 | 16 | b | e | a | r | | | | | | | | | | | | \n |
| 2 | 32 | c | a | t | | | | | | | | | | | | | \n |
| 3 | 48 | d | o | g | | | | | | | | | | | | | \n |
| 4 | 64 | e | l | e | p | h | a | n | t | | | | | | | | \n |
| 5 | 80 | f | i | s | h | | | | | | | | | | | | \n |
| 6 | 96 | g | u | p | p | i | e | | | | | | | | | | \n |
| 7 | 112 | h | o | r | s | e | | | | | | | | | | | \n |

Example use with DEBUG turned on ('#' represents debug output):

```
% ok dog
# word wanted="dog                "
# search range: bottom=0, top=8
# middle=4, word have="elephant       "
# test: want < have
# search range: bottom=0, top=4
# middle=2, word have="cat            "
# test: want > have
# search range: bottom=3, top=4
# middle=3, word have="dog            "
# test: want = have
yes
```

# 1 Specifications and Restrictions

- (Required) You may only use the i/o methods covered in class to interface with the file. This includes `lseek`, `read`, `write`, `open`, and `close`. However, feel free to use other libraries for debugging (such as `printf`) or for string comparison (such as `strcmp`).
- (Required) Makefile containing at least 3 rules:
  1. `all:` (compiles everything together and produces an executable)
  2. `clean:` (removes all object and temporary files)
  3. `run:` (command for running your executable that works with the submitted code)
- (50 points) I will test 10 different words on your program using a variety of dictionaries. Each word will be worth 5 points. You must truncate all entered words to 16 characters, since my dictionary does not contain anything longer then that.
- (10 points) You must `#DEFINE` at least two variables, one that specifies the dictionary's location and name and another that specifies the word length in the dictionary. Your code should operate around these two such that if I were to supply a dictionary with line length 10, your code would still function correctly.
- (40 points) Error catching. You must catch errors and print out an appropriate error message containing the `errno` and the message produced by that error. This means you will need to `errno.h` and `string.h`, libraries at least.

# What to turn in (in a zip on Blackboard):

- ok.c (no header files)
- Makefile
- README.txt