CS 241
Data Organization using C
Counting Characters, Words and Lines

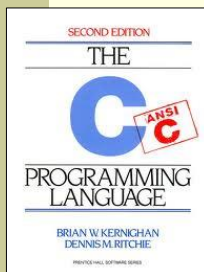Instructor: **Joel Castellanos**
e-mail: joel@unm.edu
Web: http://cs.unm.edu/~joel/

9/19/2019

1

Read: Kernighan & Ritchie

SECOND EDITION
THE
C ANSI C
PROGRAMMING LANGUAGE
BRIAN W. KERNIGHAN
DENNIS M. RITCHIE
PRENTICE HALL SOFTWARE SERIES

- Due Thursday, Sept 19
  - 4.3: External Variables
  - 4.4: Scope Rules
  - 4.5: Header Files
  - 4.6: Static Variables

- Due Tuesday, Sept 24
  - 4.7: Register Variables
  - 4.8: Block Structure
  - 4.9: Initialization

- Due Thursday, Sept 26
  - 4.10: Recursion

2

2

Count Lines of Input

```
1) void main(void)
2) { char c;                    book: int c;
3)    int numberOfLines = 0;    book: int n1 = 0;
4)
5)                1) Reads a character from stdin
6)                2) Copies the character read into c
7)                3) Compares c to EOF  (End Of File code)
8)
9)    while ( ( c = getchar()) != EOF )
10)   {
11)     if (c == '\n') numberOfLines++;
12)   }                book:  leaves out { }
13)
14)   printf("%d\n", numberOfLines);
15)}
```

3

3

Count Lines of Input

```
1) void main(void)
2) {
3)    int numberOfLines = 0;        Why is getchar()
4)    char c = getchar();           called on two different
5)                                  lines of code?
6)    while ( c != EOF )
7)    {
8)      if (c == '\n') numberOfLines++;
9)      c = getchar();
10)   }
11)   printf("%d\n", numberOfLines);
12)}
```

Compile: gcc inputLineCounter.c
Run with input: a.out < inputLineCounter.c
Output: 12

4

4

```
1)  #include <stdio.h>
2)  #define IN 1  /* inside a word */
3)  #define OUT 0 /* outside a word */
4)  void main(void)
5)  { int c, n1, nw, nc, state;
6)    state = OUT;
7)    n1 = nw = nc = 0;
8)    while (( c = getchar()) != EOF) {
9)        ++nc;
10)       if (c == '\n')
11)           ++n1;
12)       if (c == ' ' || c == '\n' || c == '\t')
13)           state = OUT;
14)       else if (state == OUT) {
15)           state = IN;
16)           ++nw;
17)       }
18)   }
19)   printf("%d %d %d\n", n1, nw, nc);
20) }
```

Coding style used
in textbook:

a) Variable names
   too short.

b) Multiple actions
   in one line.

c) Leaves out { }
   when body of
   block is only one
   statement.

d) Terse output.

e) { at end of line.

f) Indent of 5
   spaces.

5

```
1) #include <stdio.h>
2)
3) #define FALSE 0
4) #define TRUE  1
5)
6) void main(void)
7) {
8)   //Body of function shown on next slides.
9) }
```

6

6

3

```
1) void main(void)
2) {
3)    int charCount = 0;
4)    int lineCount = 1;  //Empty files has 1 line.
5)    int wordCount = 0;
6)    int insideWord = FALSE;
7)    char c = getchar();
8)
9)    while (c != EOF)
10)   { //Body of loop on next slide
11)   }
12) }
```

7

7

```
1)  if (charCount == 0)
2)  { printf("%d)", lineCount);
3)  }
4)
5)  if (c == '\n')
6)  {
7)    printf("[%d,%d]\n", charCount, wordCount);
8)    charCount = 0;
9)    wordCount = 0;
10)   insideWord = FALSE;
11)   lineCount++;
12) }
13) else
14)
```

8

8

4

```
1)    else //Char just read is not '\n'
2)    {
3)      charCount++;
4)      printf("%c", c);
5)      if (c == ' ' || c == '\n' || c == '\t')
6)      { insideWord = FALSE;
7)      }
8)      else if (insideWord == FALSE)
9)      { insideWord = TRUE;
10)        wordCount++;
11)      }
12)   }
13)   c = getchar();
14)} //end while (c != EOF)
```
9

9

```
1) void main(void)
2) { //set up variables.     Why does this not work?
3)    while (c != EOF)
4)    { c = getchar();
5)      charCount++;
6)      if (c == '\n') lineCount++;
7)      if (c == ' ' || c == '\n' || c == '\t')
8)      { insideWord = FALSE;
9)      }
10)     else if (insideWord == FALSE)
11)     { insideWord = TRUE;
12)        wordCount++;
13)     }
14)   }
15)   //output
16)}
```
10

10