Complex Conditions: NOT

- **NOT operation:** ! ex
  - evaluated to 1 (true) if ex equals to 0 (false is C terms )
  - otherwise evaluated to 0
  
  ```
  c = 4;
  !c ⇒ 0
  !(c > 3) ⇒ 0
  !(c < 3) ⇒ 1
  !(c > 3 || c < 5) ⇒ 0
  ```
Review: Char Processing Pattern

```c
#include <stdio.h>
main() {

    int ch; /* to store current character */
    /* 1. All necessary preparations */

    /* In the loop, read an input file character by character */
    while ((ch = getchar()) != EOF) {
        /* 2. What we need to do with each character */
    }
    /* 3. What we need to do after all the characters were read */
}
```

Example: Counting non-digits

```c
#include <stdio.h>

void main() {

    char ch;
    int nondig = 0;

    /* Accumulating counters in the loop */
    while ((ch = getchar()) != EOF)
        if (! (ch >= '0' && ch <= '9'))
            nondig++;

    /* Printing results */
    printf("Numbers of non-digits is %d:\n", nondig);
}
```
Example: ABC Counting

```c
#include <stdio.h>
void main() {
    char ch;
    int a = 0, b = 0, c = 0;

    /* Accumulating counters in the loop */
    while ((ch = getchar()) != EOF) {
        if (ch == 'a' || ch == 'A')
            a++;
        else if (ch == 'b' || ch == 'B')
            b++;
        else if (ch == 'c' || ch == 'C')
            c++;
    }

    /* Printing results */
    printf("nNumbers of characters:

a %d; b %d; c %d; 
", a, b, c);
}
```

Example: Word Counting

```c
#include <stdio.h>
#define IN 1 /* inside a word */
#define OUT 2 /* outside a word */
void main () {
    int c, nl, nw, nc, state;
    state = OUT;
    nl = nw = nc = 0;
    while ((c = getchar()) != EOF) {
        ++nc; /* symbol counting */
        if (c == ' ' || c == '
' || c == '	') /* line counting */
            ++nl;
        if (c == ' ' || c == '
' || c == '	') /* word counting */
            state = OUT;
        else if (state == OUT) {
            state = IN; /* count a word when turning from OUT to IN */
            ++nw;
        }
    }
    printf("%d %d %d
", nl, nw, nc);
}
```
switch: multiple selection

switch (expression) {
    case exp1:
        statement11; statement12; ... break;
    case exp2:
        statement21; statement22; ... break;
    case exp3:
        statement31; statement32; ... break;
    ...  
    default:
        statementN1; statementN2; ... break;
}

Example: ABC Counting

/* Accumulating counters in the loop */
while ((ch = getchar()) != EOF)
    switch (ch) {
        case 'a':
        case 'A':
            a++; break;
        case 'b':
        case 'B':
            b++; break;
        case 'c':
        case 'C':
            c++; break;
    }

**switch vs. else-if**

- Switch is a natural construct for multiple selection in the case of integer expression, else-if is more general
- Switch has a more efficient implementation
- Similar constructs exist in other languages
- In C language switch has to be used with break since execution is continuous

**For Loop**

```plaintext
for (ex1; ex2; ex3)
    ex4;
```

- Is simply:
  ```plaintext
  ex1;
  while (ex2) {
    ex4;
    ex3;
  }
  ```
Example: Average Line Length

```c
#include <stdio.h>

void main () {
    int c, nl = 0;
    long nc = 0;
    while((c = getchar()) != EOF) {
        if(c == '\n')
            ++nl;
        ++nc;
    }
    if(nl)
        printf("Average line length: %.2f\n", nc / (float) nl);
}
```

While and for loops

- **While loop**
  ```c
  nc = 0;
  while((c = getchar()) != EOF) {
      if(c == '\n')
          ++nl;
      ++nc;
  }
  ```

- **Equivalent for loop**
  ```c
  for (nc = 0; (c = getchar()) != EOF; ++nc)
      if(c == '\n')
          ++nl;
  ```
Example: Conversion Table F2C

```c
#include <stdio.h>

void main() {
    float fahr, celsius;
    int lower, upper, step;

    lower = 0;   /* lower limit of temperature table */
    upper = 300; /* upper limit */
    step = 20;   /* step size */

    fahr = lower;
    while (fahr <= upper) {
        celsius = (5.0 / 9.0) * (fahr - 32.0);
        printf("%3.0f %6.1f\n", fahr, celsius);
        fahr = fahr + step;
    }
}
```
For vs. While

- For loop could be considered as a compressed form of while
- For is convenient for tasks like counting and table processing
- For is very useful for array processing
- All information about loop control is collected in the header of the loop

Before Next Lecture:

- Do reading assignment
  - Perry: Chapter 15 and Chapter 17
- Run Classroom Examples
- Use KnowledgeTree
- Exercise: word counter with for
- Exercise: counting digits ('0', '1', ..., '9') in the text read from the standard input using switch and for loop