Chapter 3, then Chapter 4

Formatting Output

```
printf( "hello\n ");

printf( "formatting string", var1, var2, ... );

placeholder
%d          integer
%f          float or double
%c          char
%s          string


%6d         6 is the field width
            right-justify is the default

%-6d        - means left-justify

%6.2f       6 is field width
            2 is the number of digits to the right of the decimal point. It adds zeros if
            necessary.

printf("%6.2f\n", 11.0);

The output will be _11.00

If your field width is too small, it prints anyway.

printf("%6.2f\n", 3275.6);

The output will be 3275.60
```

Printing strings

```
%s
%10s        10 is field width, right-aligned

%-10s     left-aligned
```

Loops

```
   1)    while
   2)    for
   3)    do-while

while ( condition ) {

   // body
   // statements that are performed
   // repeatedly as long as condition
   // is true

}

The condition can be numeric.  0 is false, anything else is true

Usually there is some variable declaration before the loop and the variable(s) are
initialized before the loop.


   initialization
```

```
   while (condition) {

       // body

   } // end while
```

Sample Problem with a While Loop

```
   // print the numbers from 1 to 100
   // right-aligned neatly, one per line
   // using a while loop

   int i = 1;
   while (i <= 100) {

     printf("%3d\n", i);
     i++;
   } // end while
```

2. for loop

In C, you have to declare the loop control variable before the loop.

Same problem, numbers 1 through 100, one per line, right aligned.

```
initialization, declaration
for( initialization; condition; increment ) {

    // body

} // end for
```

increment must be a complete statement
increment is done after the body
initialization, condition, and increment can be empty

The Same Problem with a For Loop

```
int i;
for ( i = 1; i <= 100; i++ ) {

   printf("%3d\n", i);

} // end for

// when I reach this point i = 101
```

Sentinel Loop

```
   // Special type of while loop

// Sentinel loop
// A sentinel is a special value in the
// input.  It tells us to stop inputting.

1.    get the first input value
2.    while ( input != sentinel ) {
3.       process the input
4.       get the next input value
5.    } // end while
```

Sample Problem with a Sentinel Loop

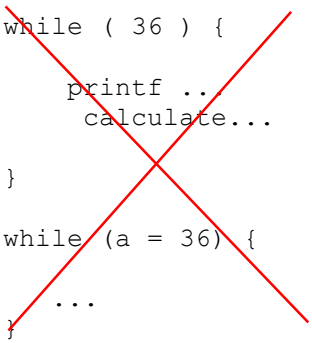
Input integers and print each integer, the square and the cubes (in a nice table) until the user types 0 for the input value.

```
int num;
printf("Enter a number (type 0 to quit): \n");
scanf("%d", &num);

while ( num != 0 ) {

   printf("%10d%10d%10d\n", num, num*num, num*num*num);
   printf("Enter a number (type 0 to quit): \n");
   scanf("%d", &num);

} // end while
```

Infinite Loop

If a loop never stops, it's called an infinite loop.

271/462: don't deliberately write an infinite loop.



```
while ( 36 ) {

    printf ...
     calculate...

}

while (a = 36) {

   ...
}
```

Two symptoms of an infinite loop

1. output fills the screen and doesn't stop
2. no output at all, the program seems frozen

Press Ctrl-C on Linux terminal. Or use the Stop/End/X button in your IDE.

Break Statement

You can stop a loop by putting a break statement inside the body.

```
// print the numbers from 1 to 100
int i = 1;
while ( i ) {
   printf("%3d\n", i);
   i++;
   if (i > 100) // bad programming practice
      break;
}
```

Functions (Java methods)

Function header: return_type function_name (parameters)

```
int main ( void ) {

}

int main ( ) {

}
```

The function_name must follow the same rules for naming identifiers as choosing variable names.

Two styles:

1. underscore style

   ```
   my_function
   ```

2. camel case

   ```
   myFunction
   ```

void can be used in the return type. void means that the function doesn't return a value.

Sample Problem with a Function

Write a C function that accepts two parameters (arguments): 2 sides of a right triangle, and returns the length of the hypotenuse.

Precondition: Assume that the two parameters contain positive numbers.

```
#include <stdio.h>
#include <math.h>

double hypo ( double a, double b ) {

   double c;
   c = sqrt( a * a + b * b );
   return c;

} // end hypo

int main(void) {

   // input the two sides
   double side1, side2;
   printf("Type side 1 length:\n");
   scanf("%lf", &side1);
   printf("Type side 2 length:\n");
   scanf("%lf", &side2);

   printf("%.2f\n", hypo( side1, side2 ) );
} // end main
```