```c
/*================================================================
 * This example is intended as a first example of code
 * in the C programming language.  It illustrates:
 *
 *   1. The structure of a C program:
 *        -- comments (block and inline)
 *        -- #include's
 *        -- Prototypes
 *        -- The  ** main **  function
 *        -- Other functions
 *
 *   2. The input/compute/output pattern
 *        -- Input via scanf
 *        -- Output via printf
 *        -- Declaring and using variables, including types
 *
 *   3. Definite and indefinite loops (FOR and WHILE statements)
 *
 *   4. Defining and calling functions
 *        -- including functions that have parameters
 *        -- and functions that return a value
 *
 *   5. Notations for:
 *        -- Ending a statement (with a semicolon)
 *        -- Delineating a block of code
 *             -- left-curly-brace   {   to begin the block
 *             -- right-curly-brace  }   to end the block
 *             -- style: left-curly-brace is placed at the end of the
 *                       line where the block begins, and matching
 *                       curly-brace is placed on a line by itself
 *                       where the block ends, indented to match the
 *                       code that follows
 *
 * Examples to follow will extend the above to include:
 *
 *   6. Conditionals
 *   7. Indefinite loops (WHILE statements)
 *   8. Structures, #define and typedef
 *   9. Pointers, arrays and strings
 *
 * The main functions in this file are:
 *   main
 *   temperature_convert
 *   square_root_table
 *   chaos
 *   sum_cosines
 *   first_cosine
 *
 * Authors: David Mutchler, Delvin Defoe, John Zelle,
 *          many others before them, and now you.  May, 2012.
 ================================================================
 */

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

#define TRUE 1
#define FALSE 0

void temperature_convert();
void square_root_table(int n);
void chaos(float constant, int how_many_to_print);
float sum_cosines(int n);
int first_cosine(float threshold);

void print_chaos_intro();
void print_chaos_header(float constant);
void print_chaos_footer(float constant);

/*****************************************************************
 * main - Runs four input/compute/output examples:
 *   a. Input a temperature in Celsius & print its Fahrenheit
equivalent.
 *   b. Print a table of square roots.
 *   c. Generate chaotic numbers from an initial "seed".
 *   d. Return the sum of cosines of integers from 1 to n.
 *   e. Return the first positive integer whose cosine is > a
threshold.
 *****************************************************************/
```

```c
int main() {
	float sum1, sum2;
	float threshold1, threshold2;
	int first1, first2;
	int n1, n2;

	// Example 1: Temperature convert.
	temperature_convert();

	// Example 2:  Square root table.
	square_root_table(20);

	// Example 3:  Chaos.
	chaos(3.9, 10);
	chaos(2.0, 25);

	// Example 4: Sum cosines from 1 to n.
	n1 = 10;
	sum1 = sum_cosines(n1);
	n2 = 10000;
	sum2 = sum_cosines(n2);

	printf("\n");
	printf("The sum of the cosines of 1, 2, ... %i is %f.\n", n1,
sum1);
	printf("The sum of the cosines of 1, 2, ... %i is %f.\n", n2,
sum2);

	// Example 5:  1st pos integer whose cos is bigger than ...
	threshold1 = 0.99;
	threshold2 = 0.99999;

	first1 = first_cosine(threshold1);
	first2 = first_cosine(threshold2);

	printf("\n");
	printf("The first integer whose cosine is bigger than %f is
%i.\n", threshold1, first1);
	printf("The first integer whose cosine is bigger than %f is
%i.\n", threshold2, first2);

	return EXIT_SUCCESS;
}
```

```c
/*****************************************************************
 * temperatureConvert: A classic input-compute-output example:
 *   -- Inputs a temperature in Celsius.
 *   -- Computes the equivalent temperature in Fahrenheit.
 *   -- Prints the result (temperature in Fahrenheit).
*****************************************************************/
void temperature_convert() {
	float celsius, fahrenheit;

	printf("What is the Celsius temperature? ");
	fflush(stdout);
	scanf("%f", &celsius);

	fahrenheit = ((9.0 / 5.0) * celsius) + 32;
	printf("Celsius temperature %4.1f is %4.1f degrees
Fahrenheit.\n", celsius, fahrenheit);
}

/*****************************************************************
 * printSquareRootTable(int n):
 *   A classic definite-loop example that:
 *    -- Prints a table of the square roots of 1 .. n,
 *       where n is the function's parameter.
*****************************************************************/
void square_root_table(int n) {
	int k;

	printf("\n");
	printf("  k  square root of k\n");
	printf("  -  ----------------\n");

	for (k = 1; k <= n; k = k + 1) {
		printf("%3i  %9.3f\n", k, sqrt(k));
	}
}

/*****************************************************************
 * chaos(int constant, int howManyToPrint):
 *   A classic input-compute-output example that also illustrates
 *   definite loops.  It computes and prints a chaotic sequence
 *   of numbers, as a function of a number input from the user.
 *   The 1st param is the constant to use in the chaotic equation.
 *   The 2nd parameter specifies how many chaotic numbers to print.
 *****************************************************************/
```

```
void chaos(float constant, int how_many_to_print) {
	float seed, x;
	int k;

	print_chaos_intro();

	printf("Enter a number between 0 and 1: ");
	fflush(stdout);
	scanf("%f", &seed);

	print_chaos_header(constant);
	x = seed;
	for (k = 0; k < how_many_to_print; k = k + 1) {
		printf("%9i %17.10f\n", k, x);
		x = constant * x * (1 - x);
	}

	printf("    Final %17.10f\n", x);
	print_chaos_footer(constant);
}

void print_chaos_intro() {
	printf("\n");
	printf("This function illustrates a mathematical function");
	printf(" that may be 'chaotic'.\n");
}

void print_chaos_header(float constant) {
	printf("\n");
	printf("The numbers that I will now print are generated\n");
	printf("from the number that you just entered.\n");
	printf("Depending on the 'constant' in the chaotic
equation,\n");
	printf("they may (or may not) jump around as if they are
'chaotic'.\n");
	printf("\n");
	printf("The 'constant' in the equation for THIS run is
%f.\n", constant);
	printf("Iteration     Chaotic number\n");
}

void print_chaos_footer(float constant) {
	printf("Look at the above numbers, which were generated
using\n");
	printf("%5.2f as its constant.  Do the numbers appear to jump
around?\n",
			constant);
	printf("If so, that constant (apparently) yields chaos!\n");
}

/*****************************************************************
 * sum_cosines(int n):
 *   Returns the sum of cosines of integers from 1 to n, inclusive.
 *****************************************************************/
float sum_cosines(int n) {
	int k;
	float sum; // Using double here would (slightly) increase
accuracy.

	sum = 0;
	for (k = 1; k <= n; k = k + 1) {
		sum = sum + cos(k);
	}

	return sum;
}

/*****************************************************************
 * first_cosine(float threshold):
 *   Returns the smallest positive integer whose cosine
 *      is greater than the given threshold.
 *****************************************************************/
int first_cosine(float threshold) {
	int k;

	k = 1;
	while (TRUE) {
		if (cos(k) > threshold) {
			break;
		}
		k = k + 1;
	}

	return k;
}
```