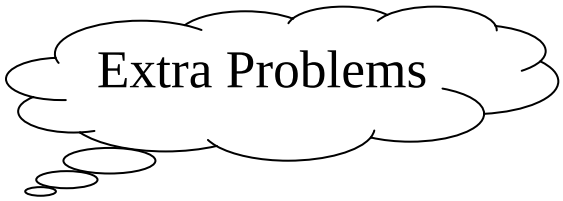

1) What would the following function do?

```
void example(int n)
  {
    int i;
    for (i=0; i<n; i++)
       cout << '*';
    cout << endl;
  }
```

How would you call this function in a program? How would you use this function in producing the following output on the screen?

```
*
**
***
****
```

2) What would be the output from the following programs?

a)

```
void change(void)
 {
   int x;
   x = 1;
 }
void main()
 {
   int x;
   x = 0;
   change();
   cout << x << endl;
 }
```

b)

```
void change(int x)
  {
    x = 1;
  }
void main()
  {
    int x;
    x = 0;
    change(x);
    cout << x << endl;
  }
```

3) Write a function prototype for a function that takes two parameters of type `float` and returns **true** (1) if the first parameter is greater than the second and otherwise returns **false** (0).
4) Write a function prototype for a function that takes two parameters of type `int` and returns **true** if these two integers are a valid value for a sum of money in pounds and pence. If not valid then **false** should be returned.

5) A function named `ex1` has a local variable named `i` and another function `ex2` has a local variable named `i`. These two functions are used together with a main program which has a variable named `i`. Assuming that there are no other errors in the program will this program compile correctly? Will it execute correctly without any run-time errors?
6) What would be the output from the following program?

```
void change(int& y)
  {
    y = 1;
  }
void main()
  {
    int x;
    x = 0;
    change(x);
    cout << x << endl;
  }
```

7) Write a function prototype for a function that takes two parameters of type `int` and returns **true** if these two integers are a valid value for a sum of money in pounds and pence. If they are valid then the value of the sum of money should also be returned in pence (without affecting the input value of pence). If not valid then **false** should be returned.

Exercises:

1) Write the prototype for a function named perimeter ( ), which returns an unsigned long int and that takes two parameters, both unsigned short int.
2) Write the definition of the function perimeter ( ) described in Exercise 1. The two parameters represent the length and width of a rectangle. Have the function return the perimeter (2*(length + width)) of the rectangle. Write a main ( ) to test the function and interact with the user.
3) Trace the following program to find what is wrong with it. Fix the error(s) in the program and show its output.

```
# include <iostream.h>
void MyFunc ( unsigned short int x ) ;
int main ( )
 {
            unsigned short int x = 5, y;
            y = MyFunc ( int ) ;
            cout << " x: " << x << " y: " << y << " \n " ;
            return 0;
   }
 void MyFunc ( unsigned short int x )
     {
        return ( 4 * x  );
     }
```

4) Trace the following program to find what is wrong with it. Fix the error (s) in the program and show its output.

```
# include <iostream.h>
int ABS( int ) ;
int main ( )
{
        int x = -5, y ;
        y = ABS ( x ) ;
        cout << "\n x: " << x << " y: " << y << " \n " ;
        x = 5 ;
        y = ABS ( x )
        cout << " x: " << x << " y: " << y ;
        return 0 ;
}
int ABS ( num ) ;
{
        if ( num < 0 )
            num = - num ;
        return ( num ) ;
}
```

5) Write a program that prints out the larger of two numbers entered from the keyboard. Use a function to do the actual comparison of the numbers.
6) Write a function called hrsToSec( ) that takes three int values ; for hours, minutes and seconds as arguments and returns the equivalent time in seconds (type long). Create a program to exercise this function by repeatedly obtaining a time value in hours, minutes and seconds from the user (format 12:59:59), calling the function and displaying the value of seconds it returns.
7) Raising a number n to a power p is the same as multiplying n by itself p times. Write a function called power ( ) that takes a double value for n and an int value for p, and returns the result as a double value. Use a default argument of 2 for p, so that if this argument is omitted, the number will be squared. Write a main ( ) function that gets values from the user to test this function.
8) Start with the power ( ) function of Exercise 7, which works only with type double. Create a series of overloaded functions with the same name that, in addition to double, also work with types char, int, long and float. Write a main ( ) program that exercises these overloaded functions with all argument types .
9) Write a function that, when you call it, displays a message telling you how many times it has been called: I have been called 3 times, or whatever. Write a main ( ) program that calls this function at least 10 times. Try implementing this function in two different ways. First, use a global variable to store the count. Second use a static variable. Which is more appropriate ? Why can't you use an automatic variable ?
10) Write a program that asks for a number and a power. Write a recursive function that takes the number to the power. Thus if the number is 2 and the power is 4, the function will return 16.
11) Try to solve the Fibonacci problem mentioned in this chapter using another method.
12) Write a function which draws a line of `n` asterisks, `n` being passed as a parameter to the function. Write a **driver** program (a program that calls and tests the function) which uses the function to output an `m x n` block of asterisks, `m` and `n` entered by the user.

13) Extend the function of the previous exercise so that it prints a line of `n` asterisks starting in column `m`. It should take two parameters `m` and `n`. If the values of `m` and `n` are such that the line of asterisks would extend beyond column 80 then the function should return **false** and print nothing, otherwise it should output **true** and print the line of asterisks. Amend your driver program so that it uses the function return value to terminate execution with an error message if `m` and `n` are such that there would be line overflow.

    Think carefully about the test data you would use to test the function.

14) Write a function which converts a sum of money given as an integer number of pence into a floating point value representing the equivalent number of pounds. For example 365 pence would be 3.65 pounds.
15) Write a function

```
void floattopp(float q, int& L, int& P)
```

    which converts the sum of money `q` in pounds into `L` pounds and `P` pence where the pence are correctly rounded. Thus if `q` was 24.5678 then `L` should be set to 24 and `P` should be set to 57. Remember that when assigning a real to an integer the real is truncated. Thus to round a real to the nearest integer add 0.5 before assigning to the integer.
    Write a simple driver program to test the function. Think carefully about the boundary conditions.