CSC 180 lab 4: Loops

Mon Oct 1 or Thurs Oct 4, 2001

1. Write a program to output "Hello, world" not once but ten times.

2. Write a program to add the numbers 1+2+3+4+5+6+7+8+9+10, using a loop, and print the result (which will be 55).
(The number 10 should appear only once in the program, and changing that number at that position should change the sequence being summed accordingly; thus printf("%d\n", 1+2+3+4+5+6+7+8+9+10) is not adequate.)

3. As you know, the sum of the integers from 1 to n, inclusive, is $(n^2+n)/2$. Write a program which verifies this formula, for $1 \le n \le 100$. That is, your program will loop over these values, and inside will have a nested loop which performs the addition; then inside the outer loop but outside the inner loop it will have an 'if' statement which compares the sum to the result of this formula. (If that sentence is impenetrable, ignore it for now, and it may or may not be of use later.)

So first your program will verify that $1=(1^2+1)/2$; next it will verify that $1+2=(2^2+2)/2$; next it will verify that $1+2+3=(3^2+3)/2$; and so on.

For each exception it finds, it should output a one-line message, such as ''formula doesn't work for n=35''. (Of course, if your program is correct, there will be no output, thus verifying the formula for $1 \le n \le 100$.)

Prime numbers

A *factor* of an integer is a number which divides it evenly. So *d* is a factor of *p* exactly when *p % d* == 0 (recall the ''remainder'' operator ''%'').

A *prime* number is a positive integer with exactly two factors: itself and 1. The numbers 2, 3, 5, and 7 are prime. 4 is not prime because it is divisible not only by 1 and 4, but also by 2.
9 is not prime because it is divisible not only by 1 and 9, but also by 3.

A number with more than two factors (such as the numbers 4 and 9, as discussed above) is called *composite*.

Prime numbers have a variety of applications; one computer application is in cryptography, especially ''public key'' cryptographic systems.

4. Testing a number to see whether it is prime is a good candidate for a separate function. Its header line will be:

```
int isprime(int p)
```

It should not consult any global variables.

The argument to isprime() must be at least 2. A block comment immediately before the isprime() function should explain this additional information about the function's interface, as well as any other comments you have.

The *isprime* function tests all numbers between 2 and $p-1$ to see whether they divide *p*.
If any of them do, the number is not prime. If none of them does, the number is prime. The function will return 1 (true) if the number is prime and 0 (false) if it is not.

Write this function and test it with the following main program (~ajr/lab4/first10.c on ecf):

```
#include <stdio.h>

int main()
{
    int i;
    extern int isprime(int p);
```

(continued)

```
    printf("Primes up to 10:\n");
    for (i = 2; i <= 10; i = i + 1)
        if (isprime(i))
            printf("%d\n", i);

    return 0;
}
```

The output of this should be 2, 3, 5, 7 (on four lines and without the commas, obviously).

Your isprime() function should be in a separate isprime.c because you will link it with three further main programs below.

It should not do any input and output. That is for the calling function to do. If the isprime() function does not do any input or output, then it is a useful library function, and resembles ‘‘pure’’ functions such as the sqrt() function.

5. Write an alternate main function which repeatedly prompts for a number, uses isprime() to determine its primality, then outputs ‘‘%d is prime’’ or ‘‘%d is composite’’. When you use scanf() to return a number, it will return 1 if it indeed read the one item. When scanf() returns a value other than 1, your loop should exit. This will represent non-numeric input or EOF. When you use the program, then, you can type control-D to end (by signalling EOF), or you can type ’q’ or anything else non-numeric to end, too.

You will write this alternate main function in a new file, and then link that file with your isprime.c to produce this new program.