Chapter 1: Introduction

1.3 Because of round-off errors, it is customary to specify the number of decimal places that should be included in the output and round up accordingly. Otherwise, numbers come out looking strange. We assume error checks have already been performed; the routine *Separate* is left to the reader. Code is shown in Fig. 1.1.

1.4 The general way to do this is to write a procedure with heading

```
void ProcessFile( const char *FileName );
```

which opens *FileName,* does whatever processing is needed, and then closes it. If a line of the form

```
#include SomeFile
```

is detected, then the call

```
ProcessFile( SomeFile );
```

is made recursively. Self-referential includes can be detected by keeping a list of files for which a call to *ProcessFile* has not yet terminated, and checking this list before making a new call to *ProcessFile.*

1.5 (a) The proof is by induction. The theorem is clearly true for $0 < X \le 1$, since it is true for $X = 1$, and for $X < 1$, $\log X$ is negative. It is also easy to see that the theorem holds for $1 < X \le 2$, since it is true for $X = 2$, and for $X < 2$, $\log X$ is at most 1. Suppose the theorem is true for $p < X \le 2p$ (where p is a positive integer), and consider any $2p < Y \le 4p$ ($p \ge 1$). Then $\log Y = 1 + \log (Y / 2) < 1 + Y / 2 < Y / 2 + Y / 2 \le Y$, where the first inequality follows by the inductive hypothesis.

(b) Let $2^X = A$. Then $A^B = (2^X)^B = 2^{XB}$. Thus $\log A^B = XB$. Since $X = \log A$, the theorem is proved.

1.6 (a) The sum is 4/3 and follows directly from the formula.

(b) $S = \frac{1}{4} + \frac{2}{4^2} + \frac{3}{4^3} + \cdots$. $4S = 1 + \frac{2}{4} + \frac{3}{4^2} + \cdots$. Subtracting the first equation from the second gives $3S = 1 + \frac{1}{4} + \frac{2}{4^2} + \cdots$. By part (a), $3S = 4/3$ so $S = 4/9$.

(c) $S = \frac{1}{4} + \frac{4}{4^2} + \frac{9}{4^3} + \cdots$. $4S = 1 + \frac{4}{4} + \frac{9}{4^2} + \frac{16}{4^3} + \cdots$. Subtracting the first equation from the second gives $3S = 1 + \frac{3}{4} + \frac{5}{4^2} + \frac{7}{4^3} + \cdots$. Rewriting, we get $3S = 2\sum_{i=0}^{\infty} \frac{i}{4^i} + \sum_{i=0}^{\infty} \frac{1}{4^i}$. Thus $3S = 2(4/9) + 4/3 = 20/9$. Thus $S = 20/27$.

(d) Let $S_N = \sum_{i=0}^{\infty} \frac{i^N}{4^i}$. Follow the same method as in parts (a) - (c) to obtain a formula for S_N in terms of S_{N-1}, S_{N-2}, ..., S_0 and solve the recurrence. Solving the recurrence is very difficult.

```
double RoundUp( double N, int DecPlaces )
{
        int i;
        double AmountToAdd = 0.5;

        for( i = 0; i < DecPlaces; i++ )
                AmountToAdd /= 10;
        return N + AmountToAdd;
}

void PrintFractionPart( double FractionPart, int DecPlaces )
{
        int i, Adigit;

        for( i = 0; i < DecPlaces; i++ )
        {
                FractionPart *= 10;
                ADigit = IntPart( FractionPart );
                PrintDigit( Adigit );
                FractionPart = DecPart( FractionPart );
        }
}

void PrintReal( double N, int DecPlaces )
{
        int IntegerPart;
        double FractionPart;

        if( N < 0 )
        {
                putchar('-');
                N = -N;
        }
        N = RoundUp( N, DecPlaces );
        IntegerPart = IntPart( N ); FractionPart = DecPart( N );
        PrintOut( IntegerPart );        /* Using routine in text */
        if( DecPlaces > 0 )
                putchar('.');
        PrintFractionPart( FractionPart, DecPlaces );
}
```

Fig. 1.1.

1.7 $$\sum_{i=\lfloor N/2 \rfloor}^{N} \frac{1}{i} = \sum_{i=1}^{N} \frac{1}{i} - \sum_{i=1}^{\lfloor N/2-1 \rfloor} \frac{1}{i} \approx \ln N - \ln N/2 \approx \ln 2.$$

1.8 $2^4 = 16 \equiv 1 \ (mod\ 5)$. $(2^4)^{25} \equiv 1^{25} \ (mod\ 5)$. Thus $2^{100} \equiv 1 \ (mod\ 5)$.

1.9 (a) Proof is by induction. The statement is clearly true for $N = 1$ and $N = 2$. Assume true for $N = 1, 2, ..., k$. Then $\sum_{i=1}^{k+1} F_i = \sum_{i=1}^{k} F_i + F_{k+1}$. By the induction hypothesis, the value of the sum on the right is $F_{k+2} - 2 + F_{k+1} = F_{k+3} - 2$, where the latter equality follows from the definition of the Fibonacci numbers. This proves the claim for $N = k + 1$, and hence for all N.

(b) As in the text, the proof is by induction. Observe that $\phi + 1 = \phi^2$. This implies that $\phi^{-1} + \phi^{-2} = 1$. For $N = 1$ and $N = 2$, the statement is true. Assume the claim is true for $N = 1, 2, ..., k$.

$$F_{k+1} = F_k + F_{k-1}$$

by the definition and we can use the inductive hypothesis on the right-hand side, obtaining

$$F_{k+1} < \phi^k + \phi^{k-1}$$
$$< \phi^{-1}\phi^{k+1} + \phi^{-2}\phi^{k+1}$$
$$F_{k+1} < (\phi^{-1} + \phi^{-2})\phi^{k+1} < \phi^{k+1}$$

and proving the theorem.

(c) See any of the advanced math references at the end of the chapter. The derivation involves the use of generating functions.

1.10 (a) $\sum_{i=1}^{N} (2i-1) = 2\sum_{i=1}^{N} i - \sum_{i=1}^{N} 1 = N(N+1) - N = N^2$.

(b) The easiest way to prove this is by induction. The case $N = 1$ is trivial. Otherwise,

$$\sum_{i=1}^{N+1} i^3 = (N+1)^3 + \sum_{i=1}^{N} i^3$$
$$= (N+1)^3 + \frac{N^2(N+1)^2}{4}$$
$$= (N+1)^2 \left[\frac{N^2}{4} + (N+1) \right]$$
$$= (N+1)^2 \left[\frac{N^2 + 4N + 4}{4} \right]$$
$$= \frac{(N+1)^2(N+2)^2}{2^2}$$
$$= \left[\frac{(N+1)(N+2)}{2} \right]^2$$
$$= \left[\sum_{i=1}^{N+1} i \right]^2$$