Chapter 1

Revision of Some Basic Maths Concepts

Mathematical Foundations

Series and summation:

$$1 + 2 + 3 + \ldots\ldots\ N \quad = \quad N(N+1)/2 \qquad \text{(arithmetic series)}$$

$$1 + r + r^2 + r^3 + \ldots\ldots\ldots r^{N-1} \quad = \quad (1- r^N)/(1-r), \qquad \text{(geometric series)}$$

$$\cong 1/(1-r)\ , \qquad r < 1, \text{ large } N$$

Sum of squares:

$$1 + 2^2 + 3^2 + \ldots\ldots\ldots N^2 = N(N + 1)(2N + 1)/6$$

Properties of a log Function

$\log_x a = b$ iff $x^b = a$

(we will use base 2 mostly, but may use other bases occasionally)

Will encounter log functions again and again!

$\log n$ bits needed to encode n messages.

$\log (ab) = \log a + \log b$

$\log (a/b) = \log a - \log b$

$\log a^b = b \log a$

$\log_b a = \log_c a / \log_c b$

$a^{\log n} = n^{\log a}$

$$a^{mn} = (a^m)^n = (a^n)^m$$

$$a^{m+n} = a^m a^n$$

$$(2\pi n)^{0.5} (n/e)^n \leq n! \leq (2\pi n)^{0.5} (n/e)^{n + (1/12n)}$$

Proof By Induction

Prove that a property holds for input size 1 (base case)

Assume that the property holds for input size 1,…n.

Show that the property holds for input size n+1.

Then, the property holds for all input sizes, n.

Prove that the sum of $1+2+\ldots..+n = n(n+1)/2$

$1(1+1)/2 = 1$

Thus the property holds for $n = 1$ (base case)

Assume that the property holds for $n=1,\ldots,m$,

Thus $1 + 2 +\ldots..+m = m(m+1)/2$

We will show that the property holds for $n = m + 1$, that is $1 + 2 + \ldots.. + m + m + 1 = (m+1)(m+2)/2$

This means that the property holds for n=2 since we have shown it for n=1

Again this means that the property holds for n=3 and then for n=4 and so on.

Now we show that the property holds for $n = m + 1$, that is $1 + 2 + \ldots.. + m + m + 1 = (m+1)(m+2)/2$

assuming that $1 + 2 + \ldots..+m = m(m+1)/2$

$$1 + 2 + \ldots..+m + (m+1) = m(m+1)/2 + (m+1)$$

$$= (m+1)(m/2 + 1)$$

$$= (m+1)(m+2)/2$$

Sum of Squares

Now we show that

$1 + 2^2 + 3^2 + \ldots\ldots\ldots n^2 = n(n + 1)(2n + 1)/6$

$1(1+1)(2+1)/6 = 1$

Thus the property holds for $n = 1$ (base case)

Assume that the property holds for $n=1,..m$,

Thus $1 + 2^2 + 3^2 + \ldots\ldots\ldots m^2 = m(m + 1)(2m + 1)/6$

and show the property for $m + 1$, that is show that

$1 + 2^2 + 3^2 + \ldots\ldots\ldots m^2 + (m+1)^2 = (m+1)(m + 2)(2m + 3)/6$

$$1 + 2^2 + 3^2 + \ldots\ldots\ldots m^2 + (m+1)^2 = m(m + 1)(2m + 1)/6 + (m+1)^2$$

$$=(m+1)[m(2m+1)/6 + m+1]$$

$$= (m+1)[2m^2 + m + 6m + 6]/6$$

$$= (m+1)(m + 2)(2m + 3)/6$$

Fibonacci Numbers

Sequence of numbers, $F_0 \quad F_1 , \; F_2 \; , \; F_3 \; ,\ldots\ldots.$

$F_0 \; = \; 1, F_1 \; = \; 1,$

$F_i \; = F_{i-1} \; + \; F_{i-2} \; ,$

$F_2 \; = \; 2, F_3 \; = \; 3, F_4 \; = \; 5, F_5 \; = \; 8$

Will prove that $F_{n+1} < (5/3)^{n+1}$,

$F_2 < (5/3)^2$

Let the property hold for 1,…k

Thus $F_{k+1} < (5/3)^{k+1}$? $F_k < (5/3)^k$

$F_{k+2} = F_k + F_{k+1}$,

$< (5/3)^k + (5/3)^{k+1}$

$= (5/3)^k (5/3 + 1)$

$< (5/3)^k (5/3)^2$

Proof By Counter Example

Want to prove something is not true!

Give an example to show that it does not hold!

Is $F_N < N^2$?

No, $F_{11} = 144$

However, if you were to show that $F_N < N^2$ then you need to show for all N, and not just one number.

Proof By Contradiction

Suppose, you want to prove something.

Assume that what you want to prove does not hold.

Then show that you arrive at an impossibility.

Example: The number of prime numbers is not finite!

Suppose the number of primes is finite, k.

The primes are $P_{1,}$ $P_{2\ldots\ldots}$ P_k

The largest prime is P_k

Consider the number $N = 1 + P_1 + P_2 + \ldots\ldots + P_k$

N is larger than P_k Thus N is not prime.

So N must be product of some primes.

However, none of the primes $P_{1,}$ $P_{2\ldots\ldots}$ P_k

divide N exactly. So N is not a product of primes.

(contradiction)

Recursion

A subroutine which calls itself, with different parameters.

Need to evaluate factorial(n)

$$\text{factorial}(n) = n\underline{.(n-1)\ldots 2.1}$$

$$= n * \text{factorial}(n-1)$$

Suppose routine factorial(p) can find factorial of p for all p < m. Then factorial(m+1) can be solved as follows:

$$\text{factorial}(m+1) = (m+1) * \text{factorial}(m)$$

Anything missing?

```
Factorial(m)
{
    If m = 1, Factorial(m) = 1;
    Else  Factorial(m) = m * Factorial(m-1)
}
```

Basic rules of Recursion :

- There should be a base case for which the subroutine does not call itself.
- For the general case: the subroutine does some operations, calls itself, gets result and does some operations with the result
- The subroutine should progressively move towards the base case.

Printing numbers digit by digit

- We wish to print out a positive integer, *n*. Our routine will have the heading printOut(n). Assume that the only I/O routine available printDigit(m) will take a single-digit number and output it.

```
void printOut( int n )                // Print nonnegative n
{
        if( n >= 10 )
                printOut( n / 10 );
        printDigit( n % 10 );
}
```

See proof of algorithm correctness in the textbook.

Recursion Versus Iteration

- Cumulative product of the first n numbers (n>0)

prod = 1

For j=1 to m

prod $\rightarrow$prod $*$j

- Write its recursive version.
- In general, iteration is more efficient than recursion because of maintenance of state information.

Towers of Hanoi

- Source peg, Destination peg, Auxiliary peg
- k disks on the source peg, a bigger disk can never be on top of a smaller disk
- Need to move all k disks to the destination peg using the auxiliary peg, without ever keeping a bigger disk on the smaller disk.

- We know how to move 1 disk from source to destination.
- For two disks, move the top one to the auxiliary, bottom one to the destination, then first to the destination.
- For three disks,
  - move top two disks from source to auxiliary, using destination.
  - Then move the bottom one from the source to the destination.
  - Finally move the two disks from auxiliary to destination using source.

- We know how to solve this for k=1
- Suppose we know how to solve this for k-1 disks.
- We will first move top k-1 disks from source to auxiliary, using destination.
- Will move the bottom one from the source to the destination.
- Will move the k-1 disks from auxiliary to destination using source.

```
towerOfHanoi(k, source, auxiliary, destination)
  {
     If k=1 move disk from source to destination; (base
            case)
     else
        {
            towerOfHanoi(top k-1, source, destination,
                                        auxiliary);
             Move the kth disk from source to destination;
            towerOfHanoi(k-1, auxiliary, source,
                                        destination);
        }
}
```