Note Title 1/28/2015

Sequences

$$f(0) = 0$$
$$f(1) = 1$$
$$f(2) = 2$$
$$\vdots$$
$$f(n) = n$$

$f(n) = n \qquad f: N \longrightarrow N$

In this section we will focus on function $f: N \longrightarrow R$

$$f(0) = 0, \quad f(1) = 1, \quad f(2) = 2, \ldots \quad f(n) = n$$

$$f_0 = 0; \quad f_1 = 1; \quad f_2 = 2; \quad \ldots f_n = n$$

?

$$S = f_1 + f_2 + f_3 + \cdots + f_n$$

$$\left.\begin{aligned} S &= 1 + 2 + 3 + \cdots + (n-2) + (n-1) + n \\ S &= n + (n-1) + (n-2) + \cdots 3 + 2 + 1 \end{aligned}\right\} +$$

$$2S = (n+1) + (n+1) + (n+1) + \cdots + (n+1) + (n+1) + (n+1) = n(n+1)$$

$$S = \frac{n(n+1)}{2}$$

Bubble Sort

7					
17	3				
3	17	8			
8		17	15		
15			17	2	
2				17	1
1					17

$$n-1 + (n-2) + (n-3) \ldots + 1$$

$$= 1 + 2 + 3 + \ldots + (n-1)$$

$$= \frac{(n-1)\,n}{2}$$

General formula

$f_1 = a$ $\quad f_2 = a+d$; $f_3 = a+2d$, $\ldots f_n = a+(n-1)d$

? $\quad S = f_1 + f_2 + \ldots + f_n$

$$S = a + (a+d) + (a+2d) + \cdots + a+(n-2)d + a+(n-1)d$$

$$S = a+(n-1)d + a+(n-2)d + \cdots + (a+d) + a$$

$$2S = 2a+(n-1)d + 2a+(n-1)d + \cdots + 2a+(n-1)d + 2a+(n-1)d$$

$$2S = n\left[2a+(n-1)d\right] \Rightarrow S = \frac{n}{2}\left[2a+(n-1)d\right]$$

Eg:

$$2+4+6+8+ \cdots + 100$$

$n=50$

$a=2$

$d=2$

$$= 2\left[1+2+3+ \cdots +50\right]$$

$$= 2 \quad \frac{50(50+1)}{2}$$

$\otimes$

Eg:

$f_1 = a$; $f_2 = ar$; $f_3 = ar^2$ ... ; $f_n = ar^{n-1}$

$f : N \longrightarrow R$ $a \in R$; $r \in R$

$$S = a + ar + ar^2 + \cdots + ar^{n-2} + ar^{n-1}$$

Case (1) $r = 1$; $S = a + a + a + \cdots + a = na$.

Case (2) $r \neq 1$;

$$- \left\{ \begin{aligned} S &= a + \cancel{ar^1} + \cancel{ar^2} \cdots + ar^{n-2} + \cancel{ar^{n-1}} \quad \text{—— (1)} \\ rS &= \qquad \cancel{ar} + \cancel{ar^2} \cdots + \cancel{ar^{n-1}} + ar^n \end{aligned} \right.$$

$$s - rs = a - ar^n$$

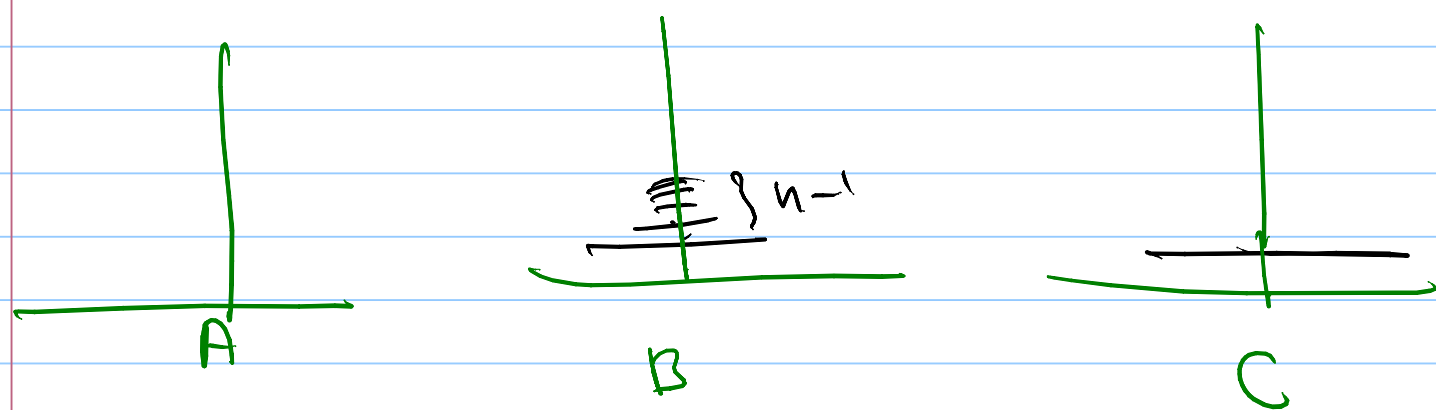
$$s(1-r) = a(1-r^n)$$

$$s = \frac{a(1-r^n)}{(1-r)} \equiv \frac{a(r^n-1)}{(r-1)}$$

$$s = 1 + 2 + 2^2 + \cdots + 2^{n-1} = \frac{1(2^n-1)}{2-1} = 2^n - 1$$

$\otimes$

Towers of Hanoi



Let M_n: be the number of moves required to move n disks from $A \to C$

$$M_n = \underset{(A \xrightarrow[n-1]{} B)}{M_{n-1}} + \underset{(A \xrightarrow[1]{} C)}{1} + \underset{(B \xrightarrow[n-1]{} C)}{M_{n-1}}$$

$$M_n = 2M_{n-1} + 1 \;; \qquad M_1 = 1$$

$$M_2 = 2M_1 + 1 = 3$$

$$M_3 = 2M_2 + 1 = 7$$

$$\otimes$$

$$M_n = 2M_{n-1} + 1 \qquad M_1 = 1$$

$$M_1 = 1$$

$$M_2 = 2M_1 + 1 = 2 \cdot 1 + 1$$

$$M_3 = 2 \cdot M_2 + 1 = 2(2 \cdot 1 + 1) + 1 = 2^2 + 2 + 1$$

$$M_4 = 2M_3 + 1 = 2(2^2 + 2 + 1) + 1 = 2^3 + 2^2 + 2 + 1$$

$$M_n = 2^{n-1} + 2^{n-2} + \dots + 2 + 1 = 2^n - 1$$

x

F

Linear search.

L

7 9 2 -5 23 11 25 . . .

Sorted:

2	4	7	9	9	12	15	17	19	20	25
0	1	2	3	4	5	6	7	8	9	10

MID = (10+0)/2 = 5

"Searching for 25"

Let C_n be the "worst case" number of comparison need to search for an element using binary search.

$$C_n = 1 + C_{\lfloor \frac{n}{2} \rfloor}$$

$$C(n) = 1 + C\left(\left\lfloor \frac{n}{2} \right\rfloor\right)$$

can show $C_n = \lfloor \log_2 n \rfloor + 1$

$\otimes$

Binary search Alg:

Input: Item to search "item"

Sorted array a with n element.

output: "No", otherwise the location of item.

```
F = 0;
L = n-1;
while (F <= L)
{
    M = (F+L)/2
    IF (a[m] == item)
        return m;
```

```
if (a[m] < item)
        F = m+1
else
        L = m-1;
}
cout << "no" << endl;
```