

3-Link Hopping problem

p → pointer to header
q → pointer to tail

```
while ( p is not equal to q ) {  
    p = p.next  
    q = q.prev  
}  
return p (return q is also correct)
```

4- Matrix Multiplication

A, B → input matrices, C → output matrix
count1, count2, count3 → counters
rowSum → temporary place to store row sums

```
for (count1 = 0 to i - 1)  
    for (count2 = 0 to j - 1) {  
        rowSum = 0  
        for (count3 = 0 to k - 1){  
            rowSum += A[count1,count3] * B[count3,count2]  
        }  
        C[count1,count2] = rowSum  
    }  
}
```

Summation can be done either in the above or below step

5- Master Method problem

- a = 2, b = 2, $\log_2 2 = 1$, $f(n) = \log n$.
Note that we can substitute **log n** with **n** since it is the next element that upper-bounds $\log n$. Hence, Case 1 is valid for $\epsilon = 1$. Consequently $T(n)$ is $\Theta(n)$.
- a = 8, b = 2, $\log_2 8 = 3$, $f(n) = n^2$
Case 1, for $\epsilon = 1$. Hence $T(n)$ is $\Theta(n^3)$
- a = 16, b = 2, $\log_2 16 = 4$, $f(n) = (n \log n)^4$
Case 2, with $k = 4$. Hence $T(n)$ is $\Theta(n^4 \log^5 n)$