Bài 1:

206.

T(n) = 1 với n = 1

= 2T(n-1) + n – 1 với $n \geq 2$

$$T(n) = \sum_{i=0}^{n-1} 2^i (n-i-1) = O(2^n)$$

224.

CM : $X(n) \geq 2^{(n-2)} \forall n \geq 1$

Có X(n) = $1 \, nếu \, n \leq 2$

= $\sum_{k=1}^{n-1} X(k) X(n-k) \, ngược \, lại$

n = 1 đúng X(1) = $1 \geq 2^{(1-2)}$

n = 2 đúng

Giả sử đúng với n = i, CM đúng với n = i + 1

TH1 : $i+1 \leq 2 \, đúng$

TH2 : $i+1 > 2$

$$X(i+1) = \sum_{k=1}^{i} X(k) X(i+1-k) = \sum_{k=1}^{i-1} X(k) X(i+1-k) + X(i) X(i+1-i) \geq \sum_{k=1}^{i-1} X(k) X(i-k) + X(i) = X(i) + X(i) \geq 2.2^{(i-2)} = 2^{(i-1)}$$

=> điều phải chứng minh

231.

T(n) = d nếu $n \leq 1$

= aT(n/c) + b với $n > 1$

$$T(n) = O\left(n^{(\log_n b)}\right)$$

= $O\left(n^{(\log_n b)} \log(n)\right)$ khi $a = c^{(\log_n b)}$

= $O\left(n^{(\log_c a)}\right) khi \, a > c^{(\log_n b)}$

Bài 2:

- In biểu diễn nhị phân
    + Độ phức tạp O(log(n))
    + Code trong file bai2.1.py
- Tìm số Fibonacci thứ n
    + Độ phức tạp $2^n$
    + Code trong file bai2.3.py

Bài 3:

- Tìm kiếm phần tử trên mảng được sắp (có/không)
    + Độ phức tạp O(log(n))
    + Giải thuật : Chọn 1 vị trí bất kỳ t. Nếu

$a[t] < value \rightarrow \forall a[i] < value, (i < t) \rightarrow value \in a[t+1, n]$

Ngược lại $value \in a[1, t]$

    + Code trong file bai3.1.py
- Tìm max min
    + Độ phức tạp O(n)
    + Giải thuật : Chia mảng thành 2 nửa, tìm min max trên mỗi nửa và min max cả đoạn là min max trên từng nửa so sánh với nhau

+ Code trong file bai3.2.py

Bài 4:

- Bài toán : Tính giá trị của biểu thức : $a^n$ mod c với a, n, c nguyên dương
- Độ phức tạp : O(log(n))
- Với n nhỏ, ta hoàn toàn có thể giải với độ phức tạp O(n). Tuy nhiên với n lớn thì chi phí là rất đắt. Bởi vậy chúng ta dùng chia để trị, giảm mức chi phí xuống còn O(log(n)).
- Ý tưởng : $a^n$ = aaaa...aaaa (n lần)

$a^n = (a^{n/2})^2$ với n chẵn

hoặc $a^n = a(a^{n/2})^2$ với n lẻ

Có thể tính $a^{n/2}$ 1 lần nên ta sẽ giảm được ½ chi phí với mỗi lần lặp → chi phí còn O(log(n))

- Code giải trong file bai4.py
- Như vậy với phương pháp chia để trị, ta đưa bài toán trên từ độ phức tạp O(n) xuống còn O(log(n)) giảm đáng kể thời gian thực thi (với n = 10^8 thời gian thực thi của thật toán O(n) lên tới 1s còn O(log(n)) thời gian chạy ~0s).