# PHÂN TÍCH VÀ THIẾT KẾ THUẬT TOÁN

## LUYỆN TẬP 2

**NHÓM 12:**
Châu Ngọc Huy 19521599
Trần Quốc Thắng 19520951
Trần Công Minh 19521855

# NỘI DUNG

1. ÔN TẬP
2. LUYỆN TẬP

# 1. ÔN TẬP

# 02 LUYỆN TẬP

2.1 Đề Bài
2.2 Abstraction
2.3 Decomposition & Pattern Recognition
2.4 Algorithm Design

# 2.1 Đề bài

Covid19 đem lại thiệt hại nặng nề về người và của cho cả thế giới đặc biệt tại Mỹ. Theo thống kê đến đầu năm 2021, Mỹ có 500 triệu ca tử vong vì Covid19 chiếm 20% số trường hợp trên toàn thế giới, và tổng GDP giảm 3.6%. Không chậm trễ trong việc vực dậy nền kinh tế trong đại dịch, tổng thống Biden đã thông qua gói cứu trợ 1900 tỷ USD.

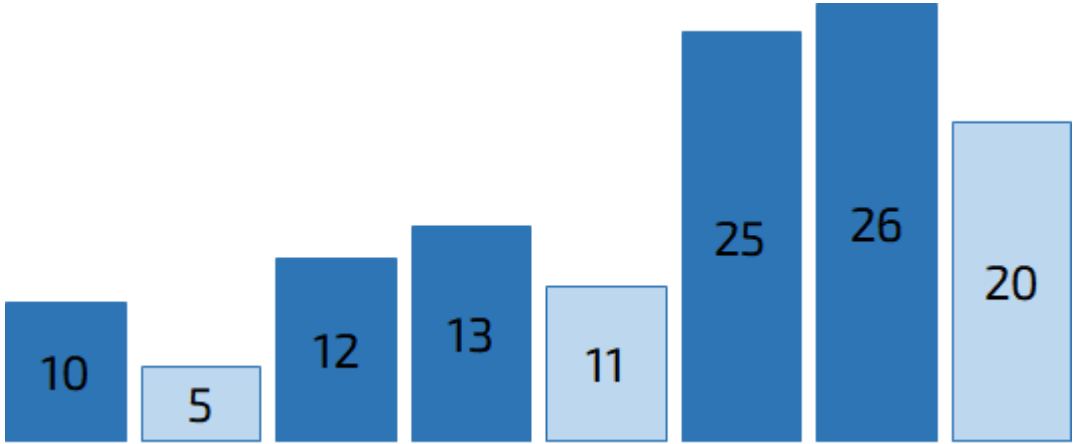
Gói cứu trợ này bao gồm 60% hỗ trợ người dân, 30% cho các doanh nghiệp và 10% dành cho các hoạt động xã hội khác, bao gồm mua vắc xin và các vật tư y tế.

Tuy nhiên gói cứu trợ dành cho doanh nghiệp không chia đều mà tập trung hơn vào các doanh nghiệp thuộc diện “tăng trưởng âm nguy hiểm” (DNG). DNG được định nghĩa là các doanh nghiệp có thiệt hại tăng theo tuần (không liên tiếp) trong đại dịch nhiều. Nếu số tuần thiệt hại tăng càng dài thì mức độ nguy hiểm càng cao và sẽ được ưu tiên hỗ trợ nhiều hơn.

Các doanh nghiệp luôn muốn nhận được sự hỗ trợ cao nhất tương xứng với thiệt hại của họ từ chính phủ. Hãy giúp họ.

# 2.2 Abstraction

Cho dãy số gồm N số $a_1$ $a_2$ … $a_N$, tìm dãy con $b_1$ $b_2$ … $b_K$ tăng dần dài nhất (không liên tiếp) của dãy a.



# 2.3 Pattern Recognition

Nhận thấy rằng phần tử $b_i$ trong dãy con tăng phải lớn hơn phần tử $b_{i-1}$.

Vậy khi thêm phần tử $b_i$ vào dãy con tăng, cần phải xét điều kiện

$b_i > b_{i-1}$.

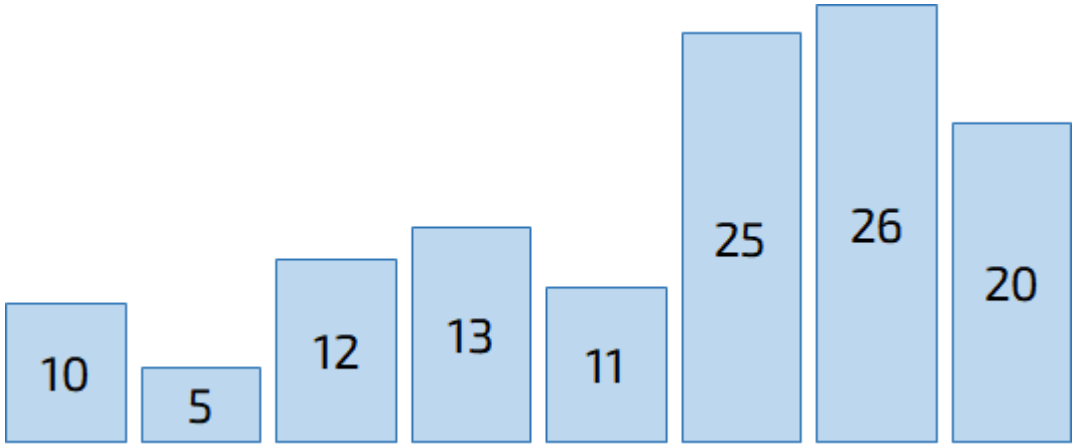
# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam

- Xét lần lượt dãy số từ đầu đến cuối.
- Tại mỗi vị trí i, nếu phần tử $a_i$ lớn hơn phần tử cuối cùng trong dãy con hiện tại thì thêm $a_i$ vào dãy con.

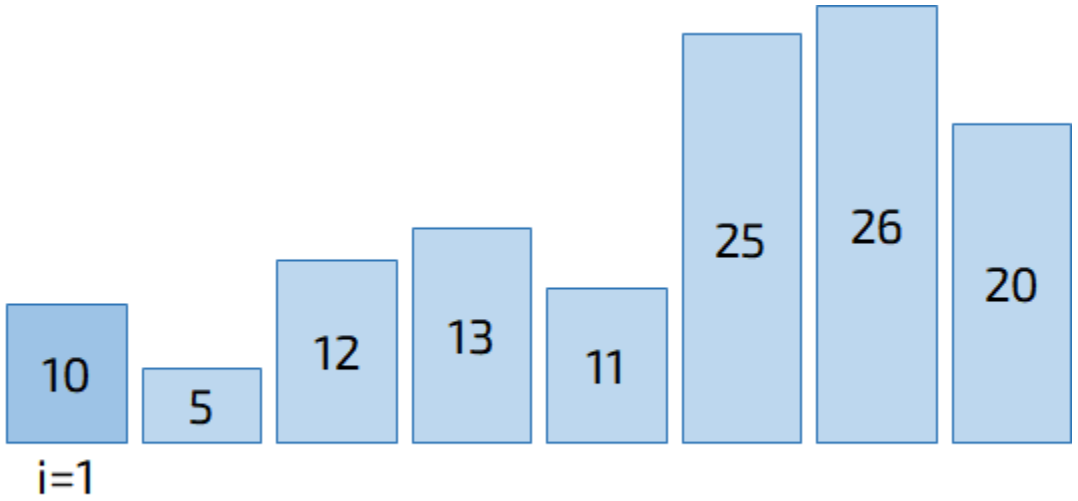
# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



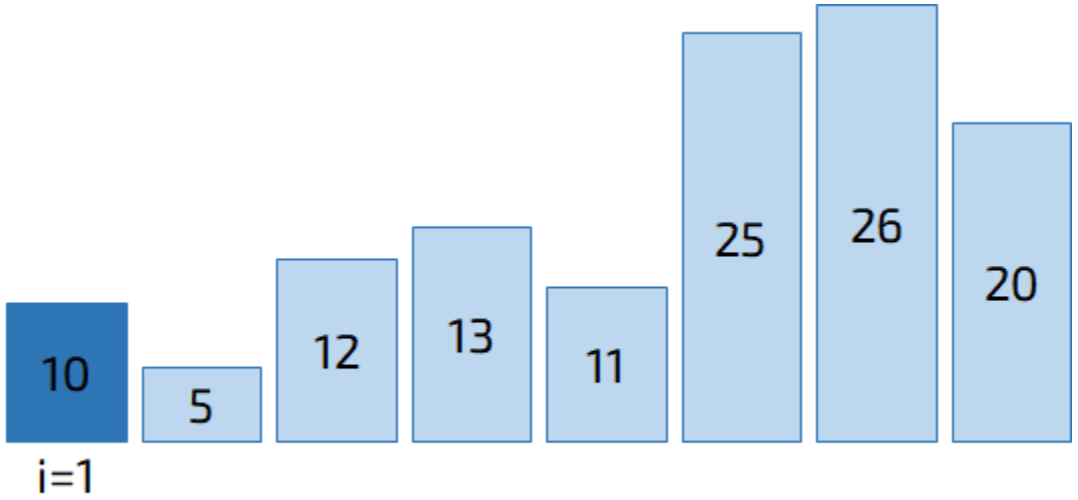
# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



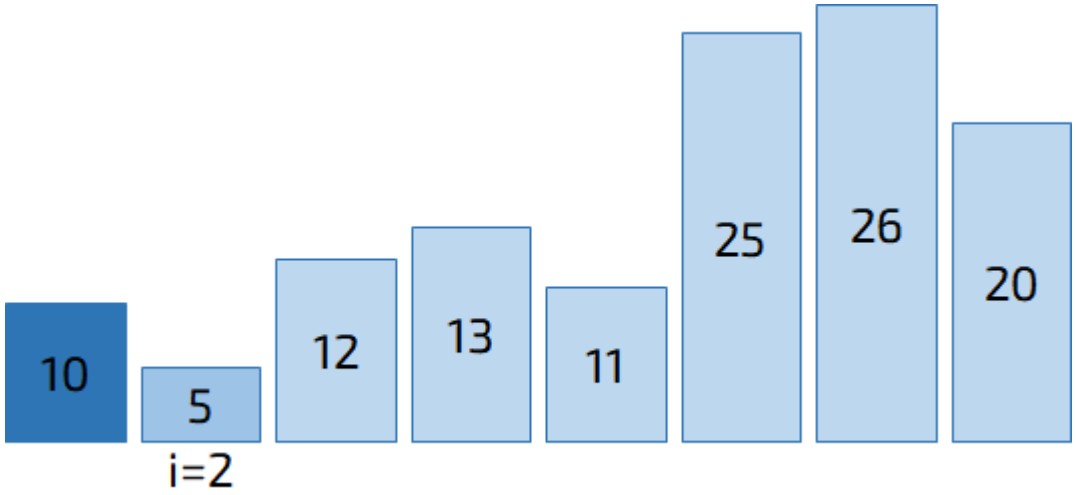
# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



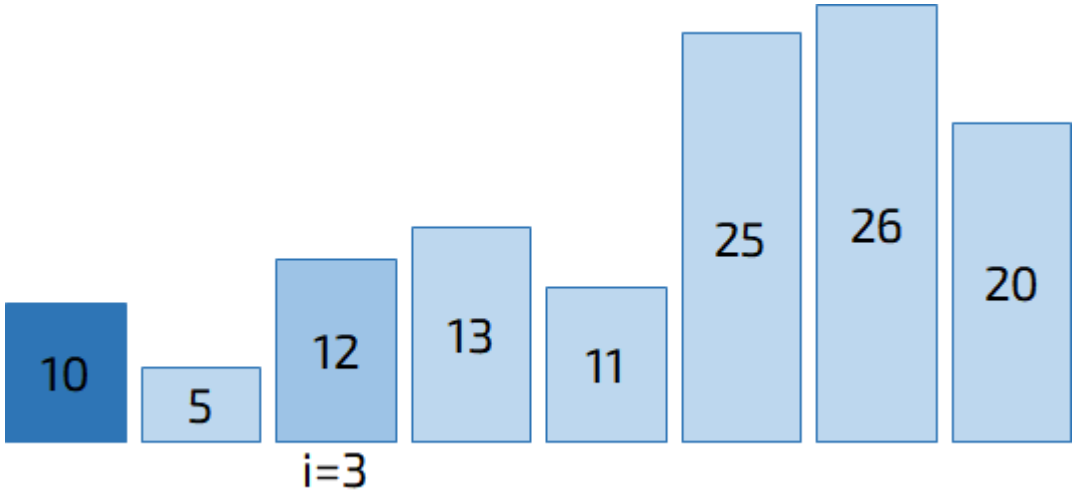
# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



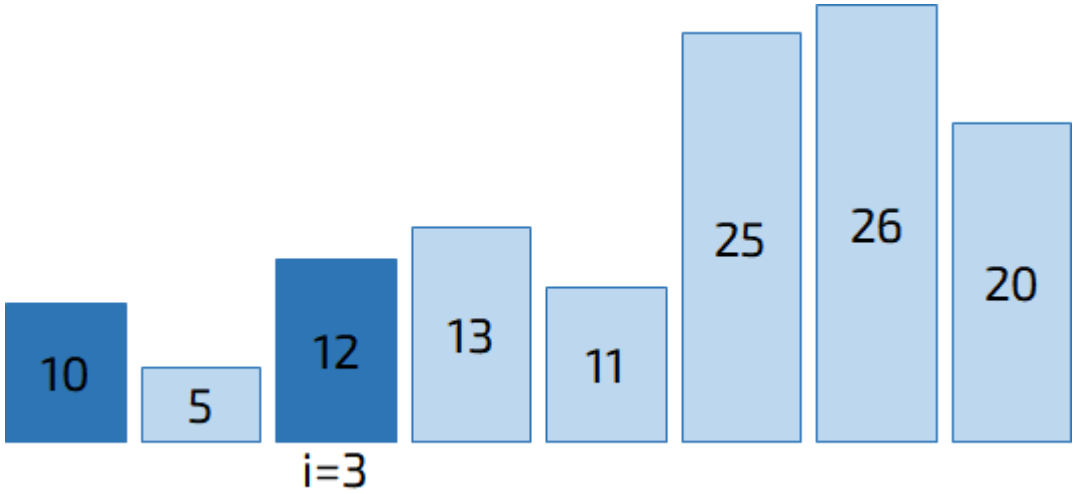
# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



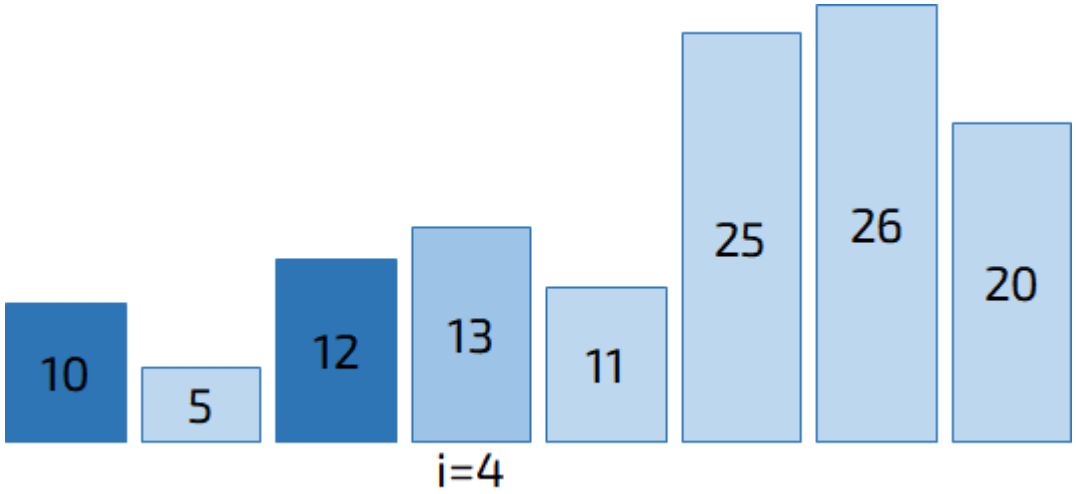
# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



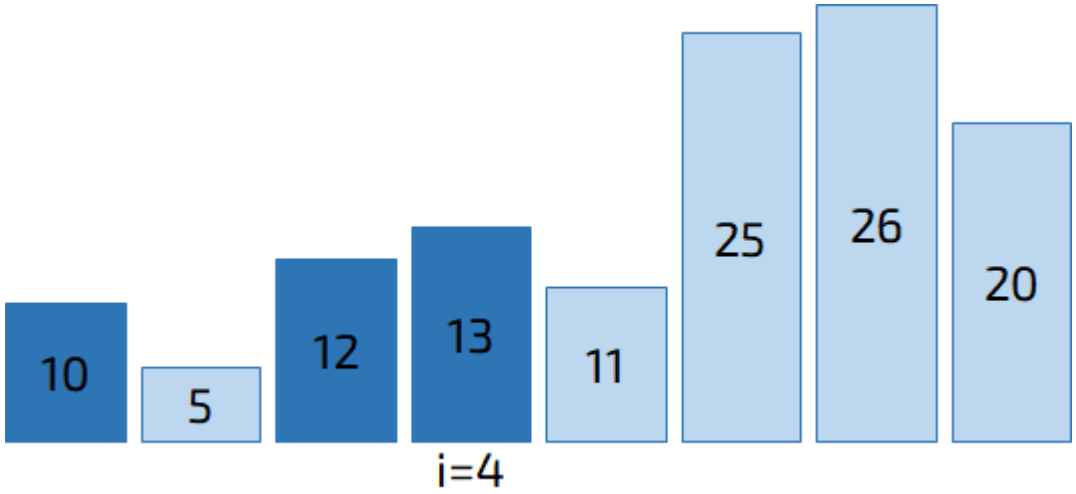
# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



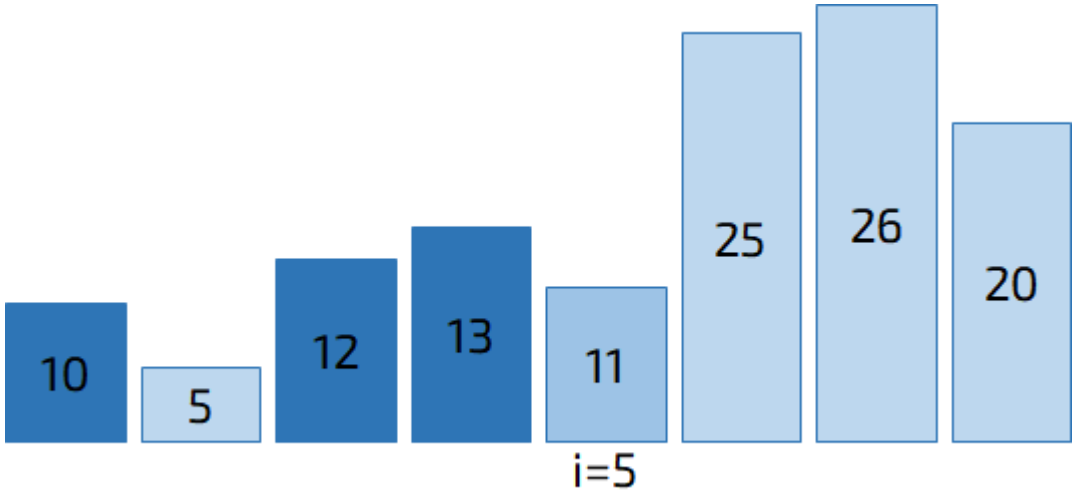
# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



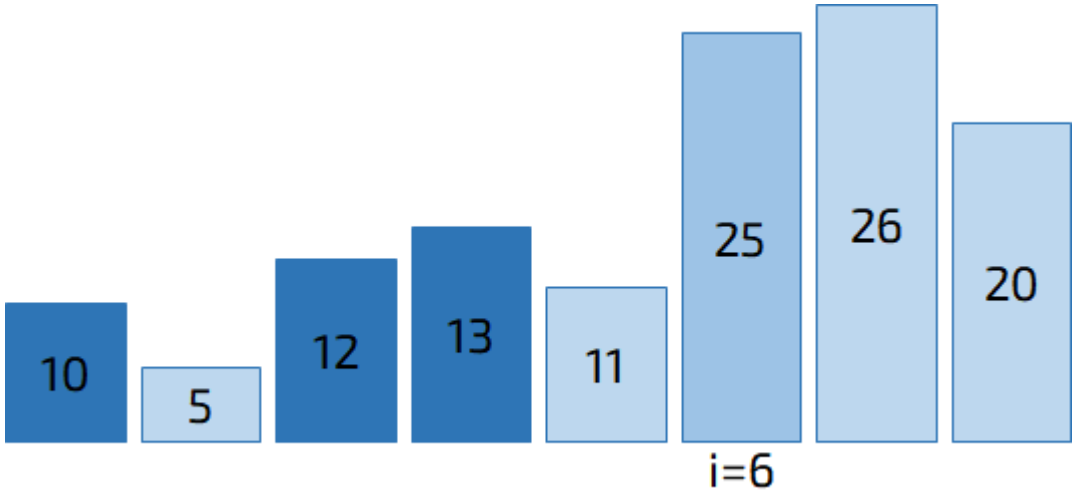
# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



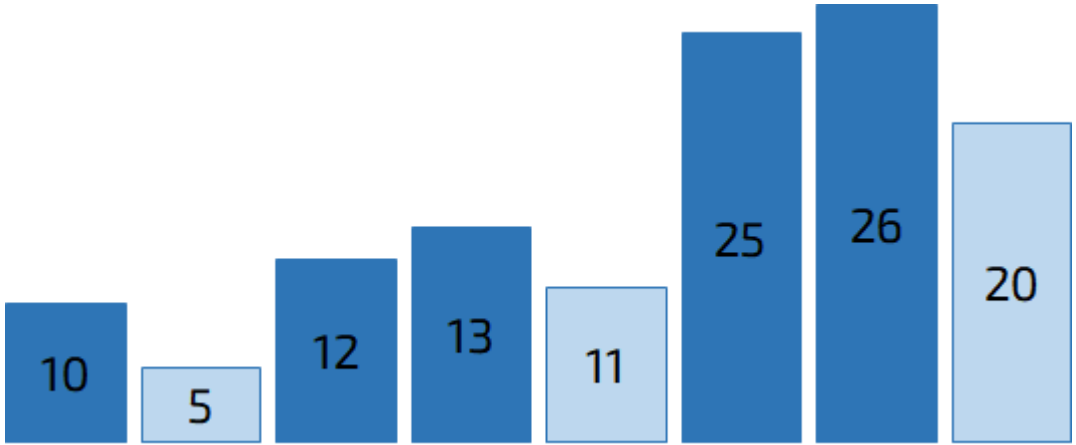
# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



# 2.3 Pattern Recognition

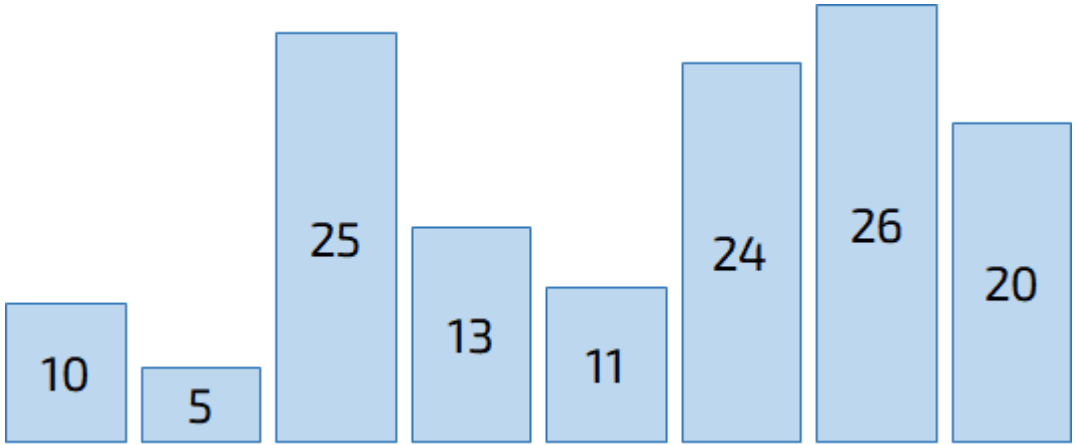
## Hướng tiếp cận Tham lam

| Độ phức tạp tính toán | Độ phức tạp bộ nhớ |
|---|---|
| O(N) | O(N) |

=> Có trường hợp chưa tối ưu

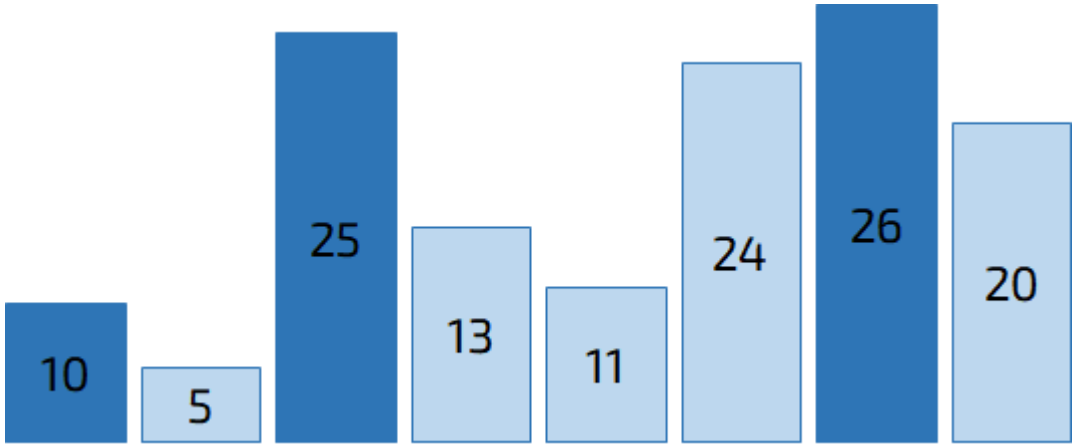
# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



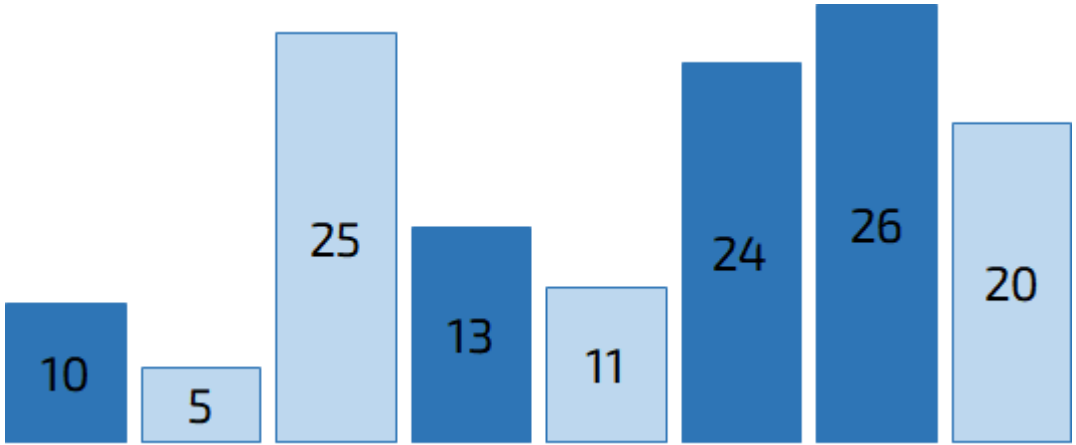
# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam



# 2.3 Pattern Recognition

## Hướng tiếp cận Tham lam

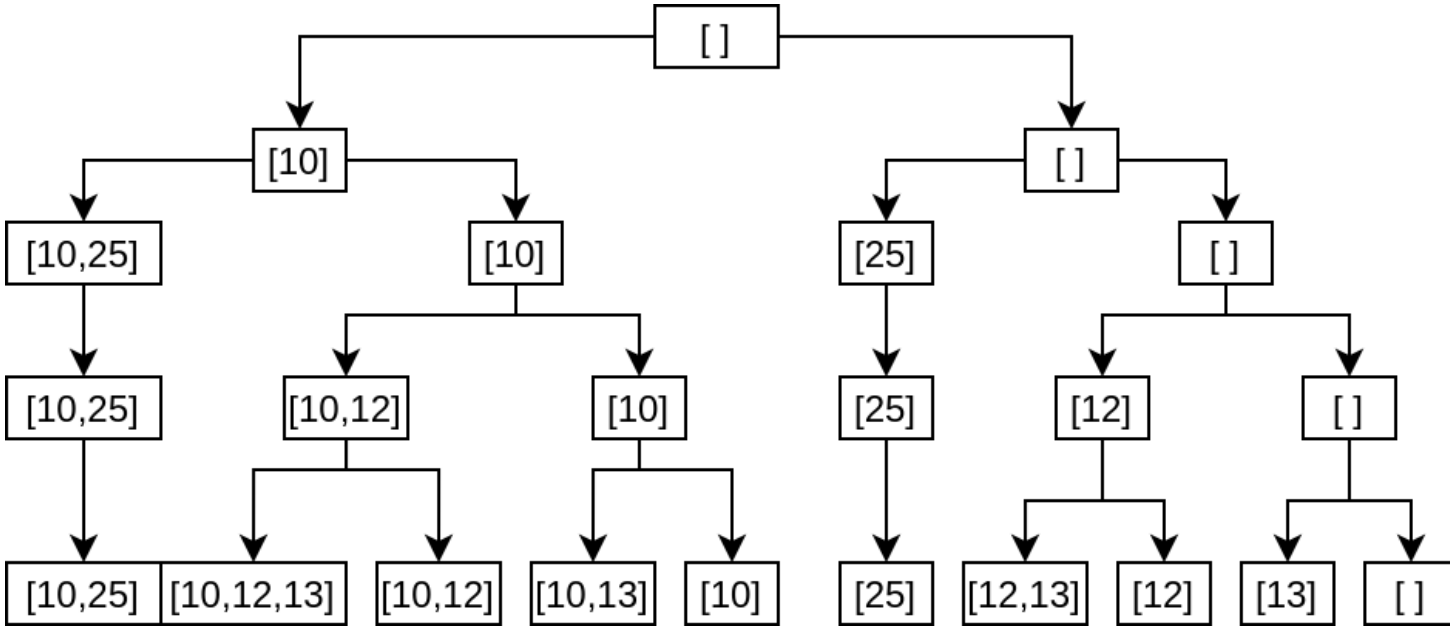
- **Nhận xét:** Tham lam không có khả năng “nhìn xa trông rộng”.
- Việc lựa chọn **thêm** hay **không thêm** phần tử vào dãy sẽ ảnh hưởng đến kết quả tối ưu toàn cục.

# 2.3 Pattern Recognition

## Hướng tiếp cận Quay lui

A = [10, 25, 12, 13]

Tham lam: B = [10, 25]



# 2.3 Pattern Recognition

## Hướng tiếp cận Quay lui

| Độ phức tạp tính toán | Độ phức tạp bộ nhớ |
| --- | --- |
| O(2^n) | O(N) |

# 2.3 Pattern Recognition

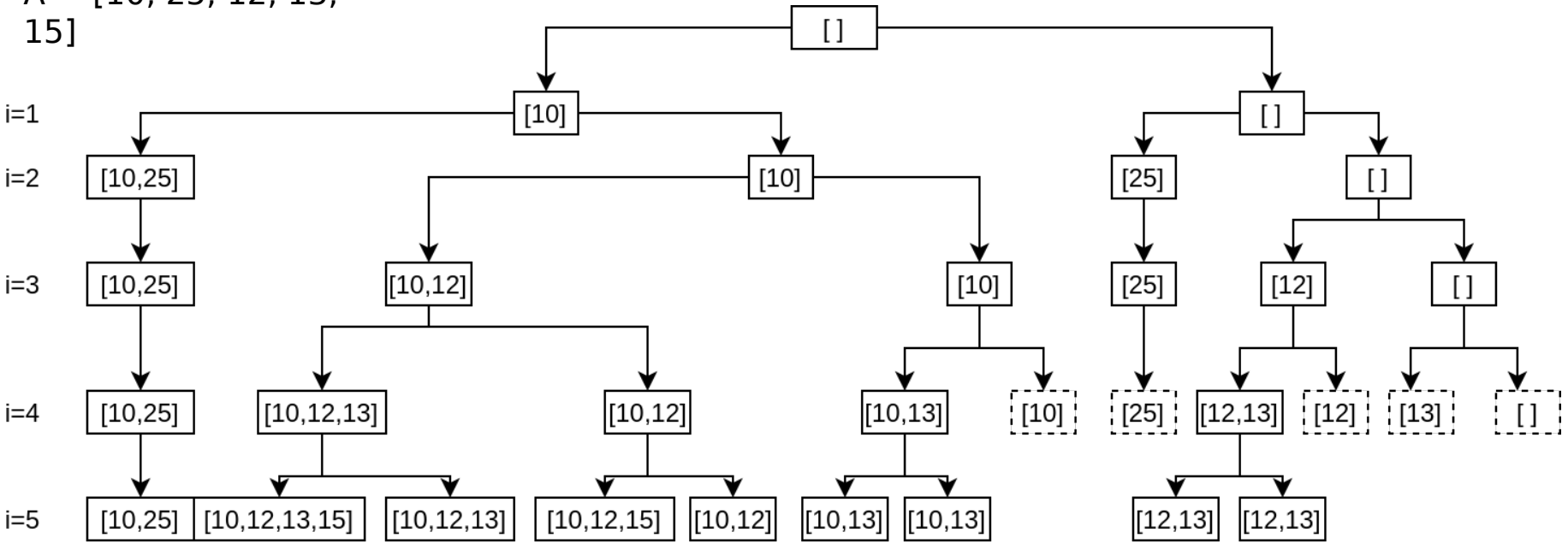
Giả sử đang xét đến phần tử thứ i của dãy N phần tử, dãy con tăng đang xét có k phần tử và dãy con tăng dài nhất mà thuật toán tìm được hiện tại có độ dài k* phần tử.

**k - i + N < k*** (tổng số phần tử dãy tăng dần và số phần tử chưa xét)
⇒ Dãy hiện tại chắc chắn không thể trở thành đáp án tối ưu
⇒ Bỏ qua branch đang xét.

# 2.3 Pattern Recognition

## Hướng tiếp cận Nhánh cận

A = [10, 25, 12, 13, 15]

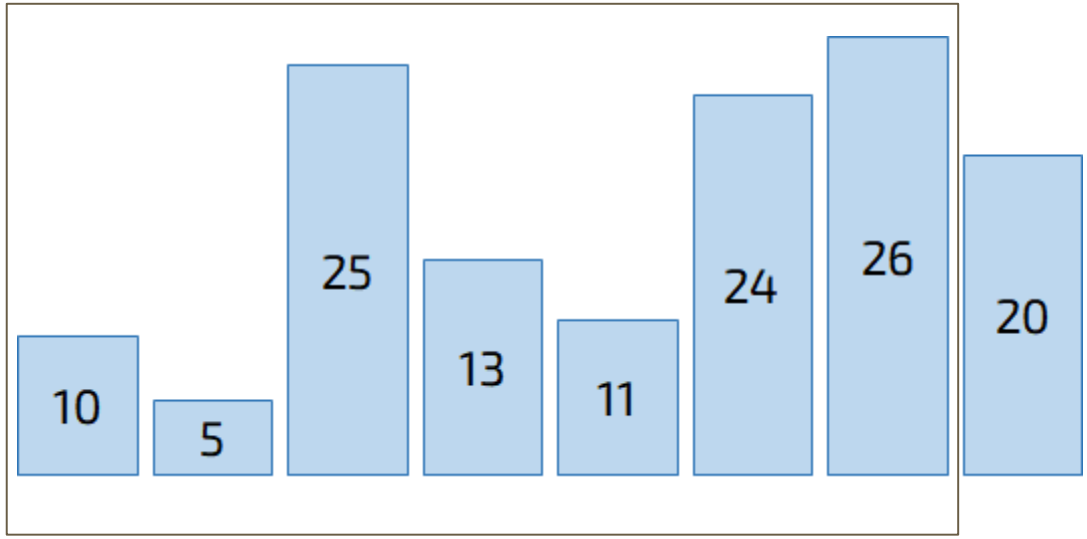


# 2.3 Pattern Recognition

## Hướng tiếp cận Nhánh cận

| Độ phức tạp tính toán | Độ phức tạp bộ nhớ |
| --- | --- |
| $O(2^N)$ | O(N) |

# 2.3 Pattern Recognition




Nếu tại mỗi phần tử liền trước đều biết được độ dài chuỗi con tăng (tính tới thời vị trí hiện tại)
=> có thể tìm được độ dài tối đa đến phần tử hiện tại
=> Bài toán con gối nhau?

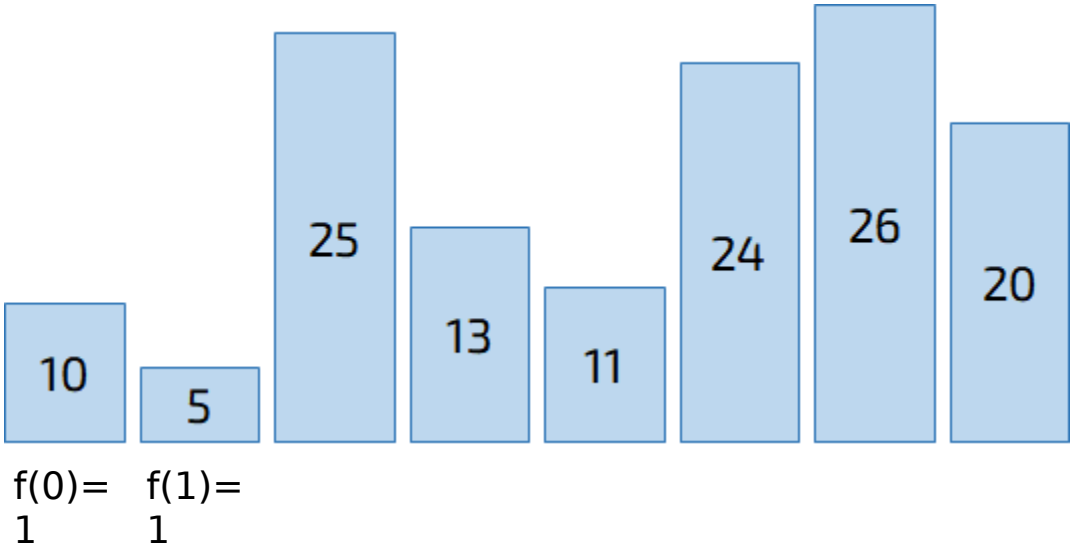
# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động

- Gọi $f_i$ là dãy con tăng dài nhất có phần tử cuối cùng là $a_i$.
- Ta có $f_i = \max(f_j) + 1$, $\forall\ j<i$ và $a_j<a_i$.
- $f_0 = 1$.

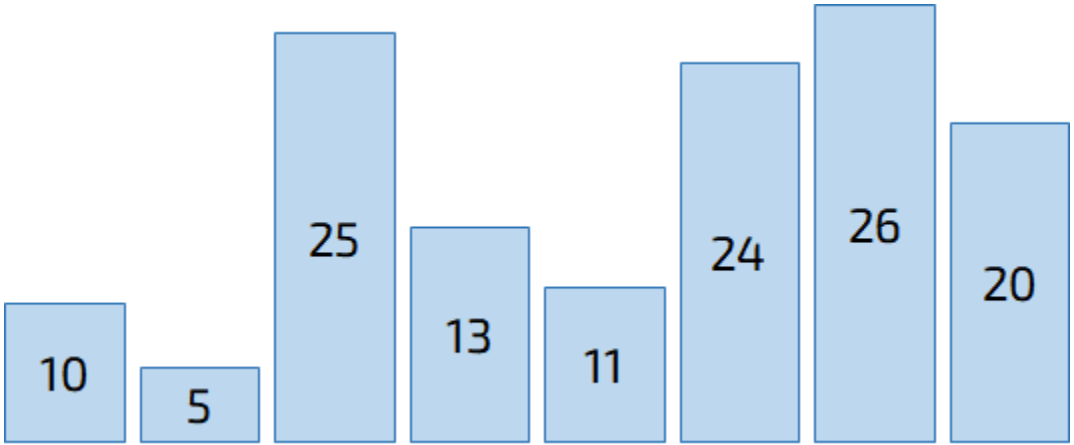
# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động



# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động

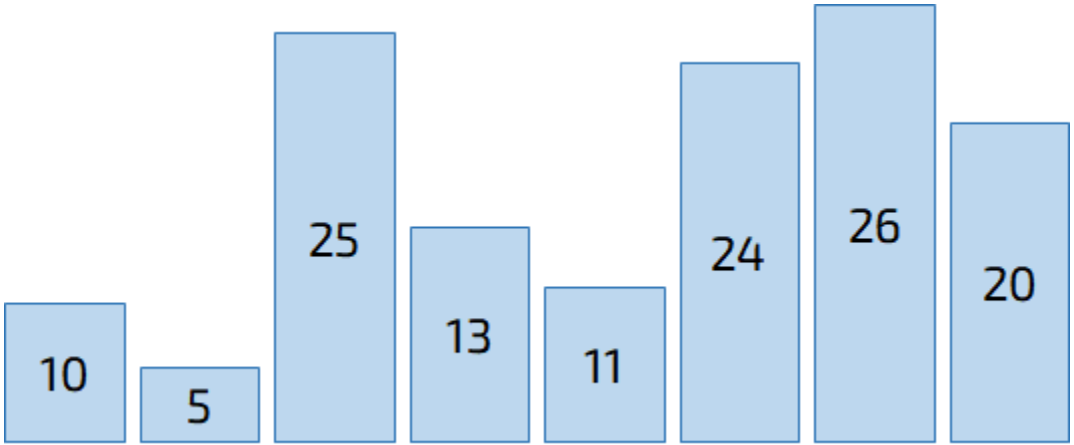



f(0)= 1  f(1)= 1

f(2)=f(0) + 1= 2

# 2.3 Pattern Recognition

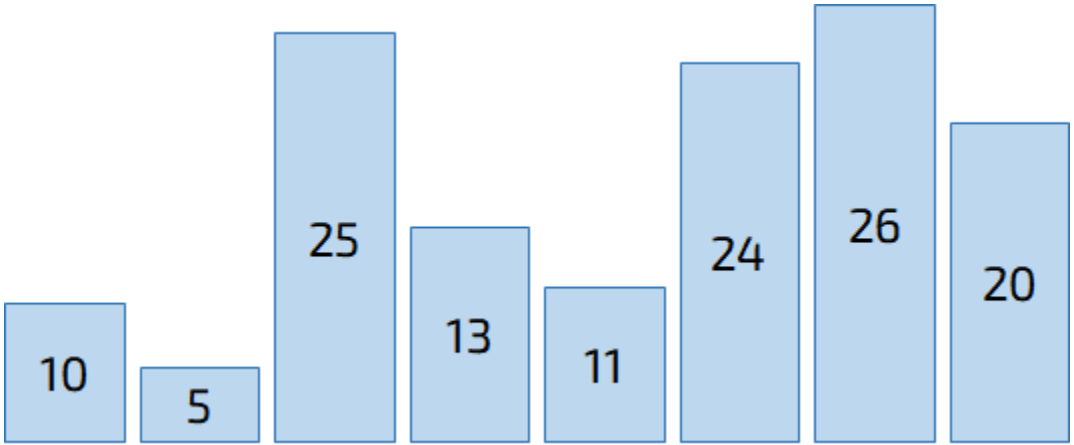
## Hướng tiếp cận Quy hoạch động




f(0)= f(1)= f(2)=
1 1 2
f(3) = f(0) + 1 = 2

# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động

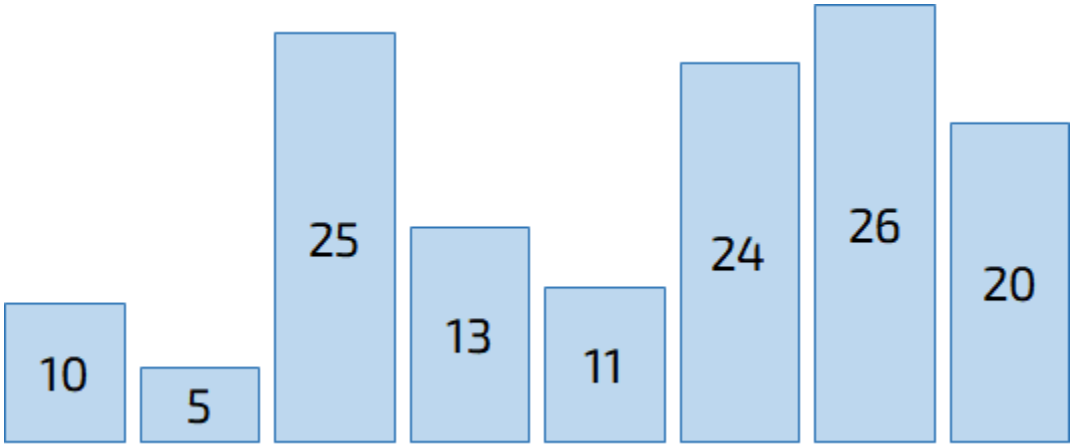



f(0)= 1 f(1)= 1 f(2)= 2 f(3)= 2

f(4) = f(0) + 1 = 2

# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động

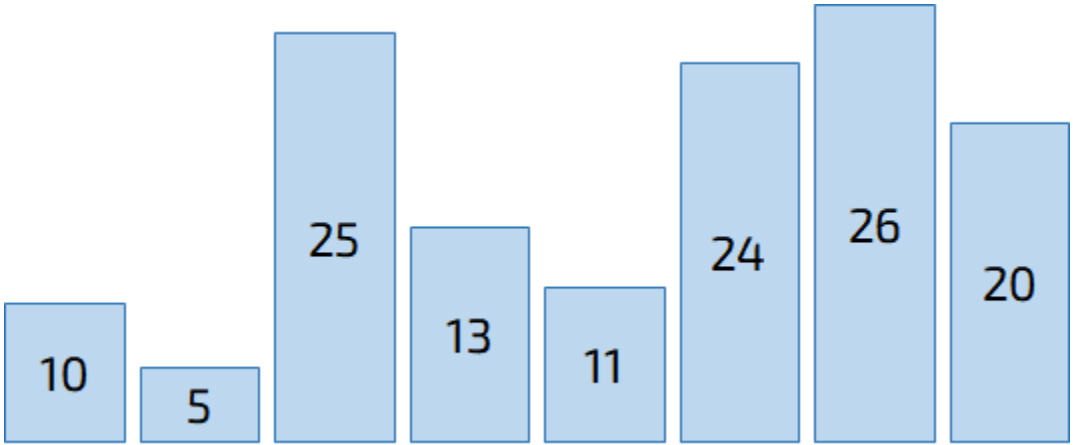



f(0)= 1 f(1)= 1 f(2)= 2 f(3)= 2 f(4)= 2

f(5) = f(3) + 1 = 3

# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động

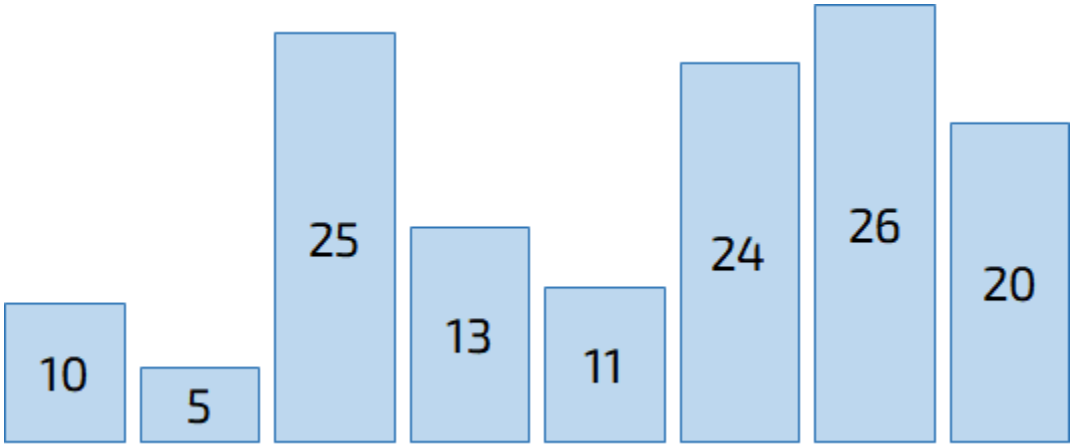



f(0)= 1 f(1)= 1 f(2)= 2 f(3)= 2 f(4)= 2 f(5)= 3

f(6)= f(5) + 1 = 4

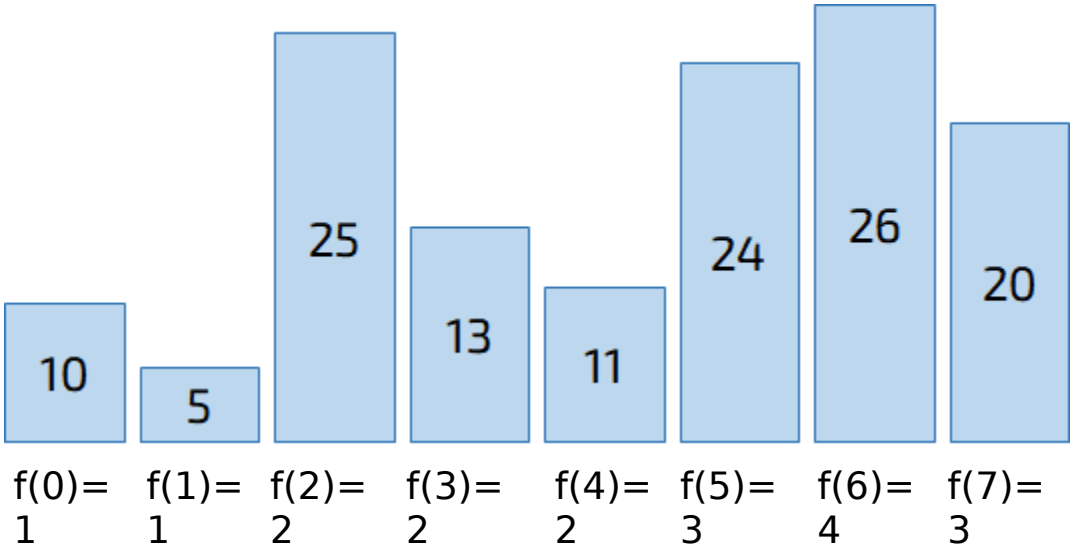
# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động



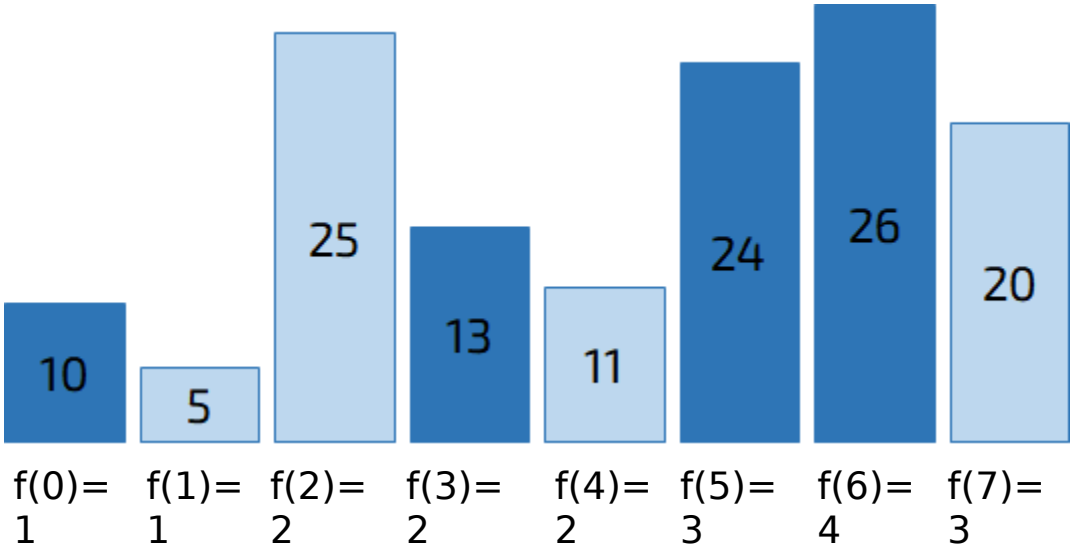
# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động



# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động



# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động

| Độ phức tạp tính toán | Độ phức tạp bộ nhớ |
|---|---|
| $O(N^2)$ | $O(N)$ |

⇒ Thuật toán đã đủ tốt, nhưng có thể tốt hơn?

# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động

*Cây Fenwick*

- ***Pattern Recognition:***

  Trong bước tìm max($f_j$), có thể dùng cấu trúc dữ liệu

  **Cây Fenwick** để truy vấn nhanh hơn.

# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động

### *Cây Fenwick*

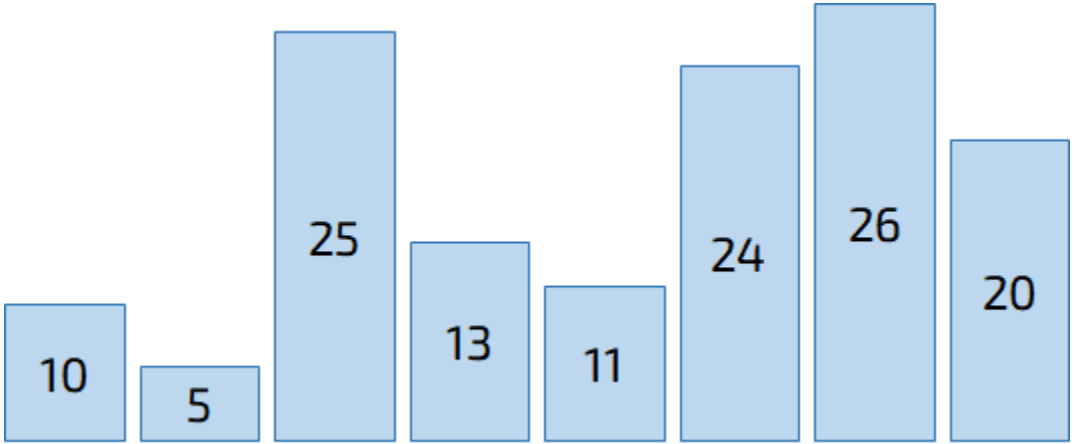
- ***Nhận xét:*** Cây Fenwick không còn hiệu quả khi các phần tử trong dãy a quá lớn.
- ***Pattern Recognition:***

  Chuẩn hóa các giá trị $a_1$, $a_2$, …, $a_N$ về khoảng [1, N] vẫn đảm bảo thứ tự lớn bé.

# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động

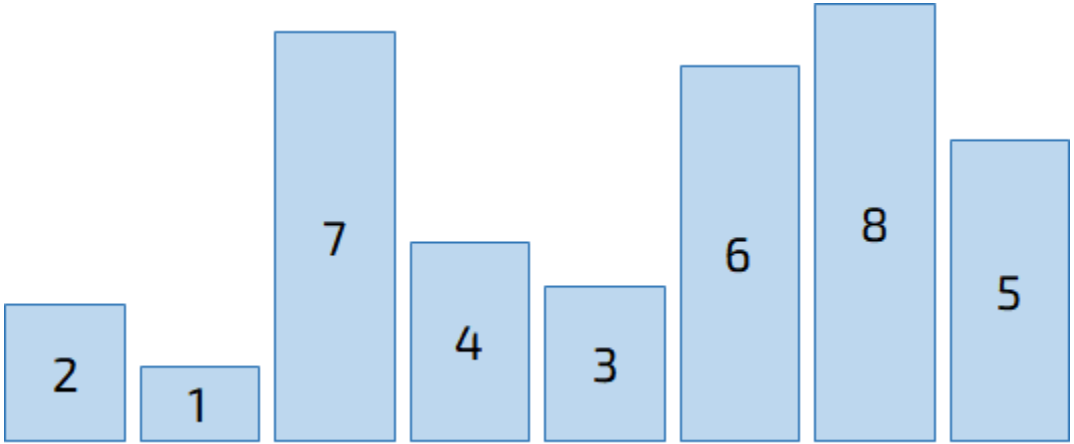
*Cây Fenwick + chuẩn hóa*



# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động

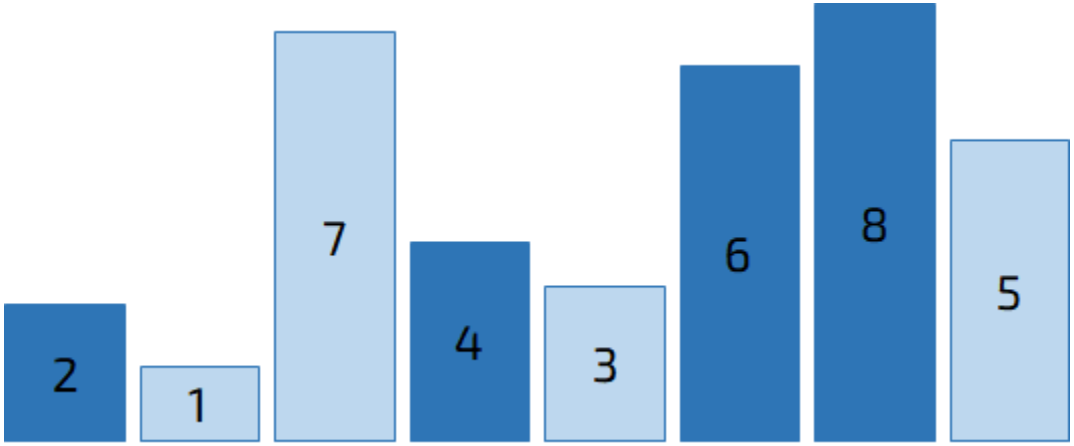
*Cây Fenwick + chuẩn hóa*



# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động

*Cây Fenwick + chuẩn hóa*



# 2.3 Pattern Recognition

## Hướng tiếp cận Quy hoạch động

*Cây Fenwick + chuẩn hóa*

| Độ phức tạp tính toán | Độ phức tạp bộ nhớ |
|---|---|
| O(NlogN) | O(N) |

# 2.4 Algorithm Design

# ANY QUESTION

# 2.5 Homework

## TÌM SỐ ĐẢO

Mực nước biển ngày càng tăng là một vấn đề khó khăn mà cả thế giới phải đối mặt trong những năm tiếp theo. Bởi vì Trái Đất Ngày một nóng lên khiến băng ở cực Bắc và cực Nam tan ra. Dự kiến đến năm 2100, nước biển sẽ dâng lên khoảng 30 - 130 cm, đe dọa hệ san hô và các khu vực thấp của thế giới. Các quốc đảo và những thành phố lớn như New York, Los Angeles, Mumbai, Sydney, Rio de Janeiro... sẽ chìm dưới nước. Vì thế, chúng ta cần phải đếm số lượng khu vực đất liền chưa bị chìm dưới nước để có thể đưa ra những sự giải pháp kịp thời.

Giả sử bản đồ là một ma trận hai chiều chỉ có giá trị là 0 hoặc 1. Số 1 tượng trưng cho khu vực đất liền và số 0 tượng trưng cho khu vực nước biển. Hai hay nhiều số 1 nằm kề nhau (9 vị trí xung quanh) chỉ được coi là một khu vực. Hãy tính số lượng khu vực đất liền có trong bản đồ.

```
Input : mat[][] = {{1, 1, 0, 0, 0},
                   {0, 1, 0, 0, 1},
                   {1, 0, 0, 1, 1},
                   {0, 0, 0, 0, 0},
                   {1, 0, 1, 0, 1}

Output : 5
```

# Tham khảo


- https://medium.com/swlh/a-visual-guide-to-solving-the-longest-increasing-subsequence-problem-dabbee570551
- https://vnoi.info/wiki/algo/data-structures/fenwick.md
- https://github.com/khanh-moriaty/CS112.L11.KHTN/tree/master/seminar
- Sema Rani (2016) LIS using backtracking and branch-and-bound approaches in CSI Transactions on ICT

# THANKS FOR LISTENING

Github repository
https://github.com/drakiez92/CS112.L21.KHTN_N12