PHÂN TÍCH THIẾT KẾ THUẬT TOÁN

GIẢNG VIÊN: **NGUYỄN THANH SƠN**

**** Nhóm 7:***

Đỗ Trọng Khánh - 19521676

Trịnh Tuấn Nam - 19521874

Nguyễn Dương Hải - 19521464

Giới thiệu phương pháp thiết kế thuật toán

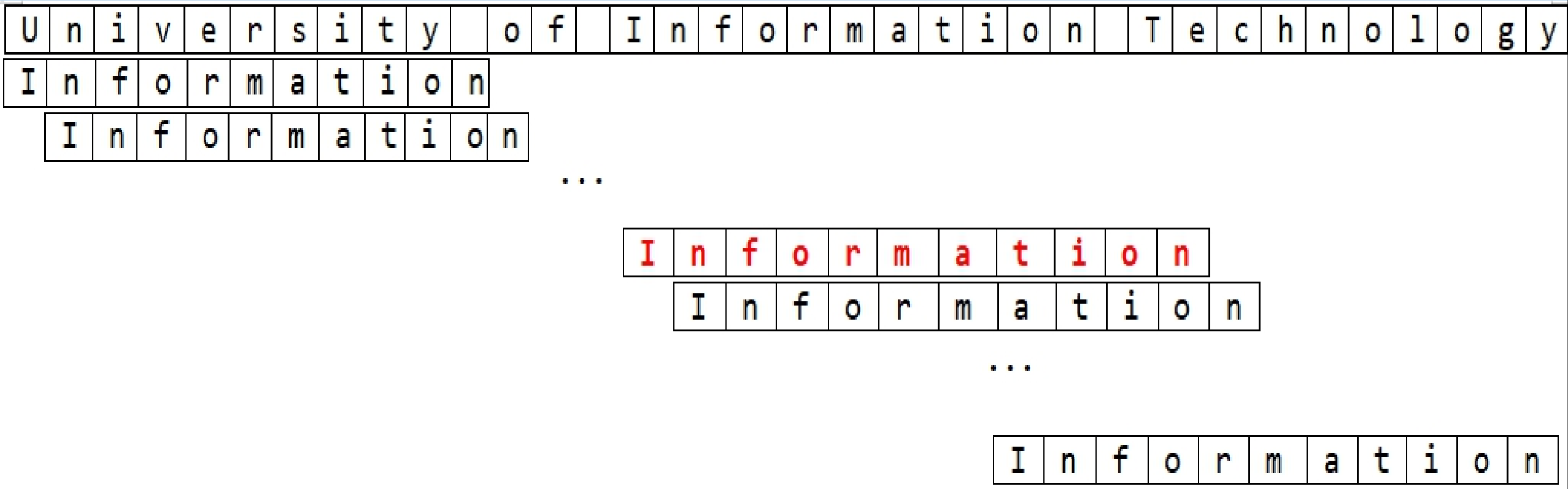
COMPLETE SEARCH - BRUTE FORCE

VÍ DỤ 1: **Cho 2 chuỗi A và B. Tìm tất cả các vị trí mà chuỗi A xuất hiện trong chuỗi B.**

+ Chuỗi A = "Information"

+ Chuỗi B = "University of Information Technology"

```
a = 'Information'
b = 'University of Information Technology'
for i in range(len(b) - len(a) + 1):
    if (b[i] == a[0]):
        for j in range(len(a)):
            if (b[i+j] != a[j]):
                break
            if (j == len(a)-1):
                print(i)
```


University of Information Technology
Information
Information
...
Information
Information
...
Information

BRUTE-FORCE

Định nghĩa:

Vét cạn là thuật toán sẽ **chạy tất cả các trường hợp có thể có** để giải quyết một vấn đề (bao gồm cả trường hợp đúng và các trường hợp sai hay còn gọi là trường hợp dư thừa).

Vét cạn là cách tiếp cận đầu tiên cho các bài toán và là một cách dễ nhận thấy nhất, cho dù nó có thể chưa phải là cách tốt nhất.

* Không có pha tiền xử lý.

Đặc điểm: * Quá trình so sánh thực hiện theo bất kỳ thứ tự nào.

* Độ phức tạp thuật toán là O(m.n)

Khi nào nên sử dụng:

- Tìm kiếm Brute-force thường được sử dụng khi kích thước vấn đề có giới hạn.

- Phương pháp này cũng được sử dụng khi tính đơn giản của công việc ưu tiên hơn yêu cầu tốc độ.

VÍ DỤ 2:

Vừa gà vừa chó
Bó lại cho tròn
Ba mươi sáu con
Một trăm chân chẵn.
Hỏi có bao nhiêu con gà,
bao nhiêu con chó ?

(22 gà - 14 chó)

```
n = 36
chan = 100
for ga in range(1, n):
    for cho in range(1, n):
        if (ga+cho == n) and (2*ga + 4*cho == chan):
            print(ga, cho)
```

ƯU ĐIỂM:

* Luôn cho ra kết quả **chính xác** cho mọi bài toán
* Sử dụng cách tiếp cận **đơn giản nhất** có thể để giải quyết vấn đề
* **Ít tốn không gian nhớ** do *không* sử dụng bộ nhớ phụ để lưu trữ

NHƯỢC ĐIỂM:

* **Độ phức tạp và thời gian thực thi** tỉ lệ thuận với **kích thước** của không gian tìm kiếm

INPUT:

* Tập dữ liệu A
* Tập nghiệm b

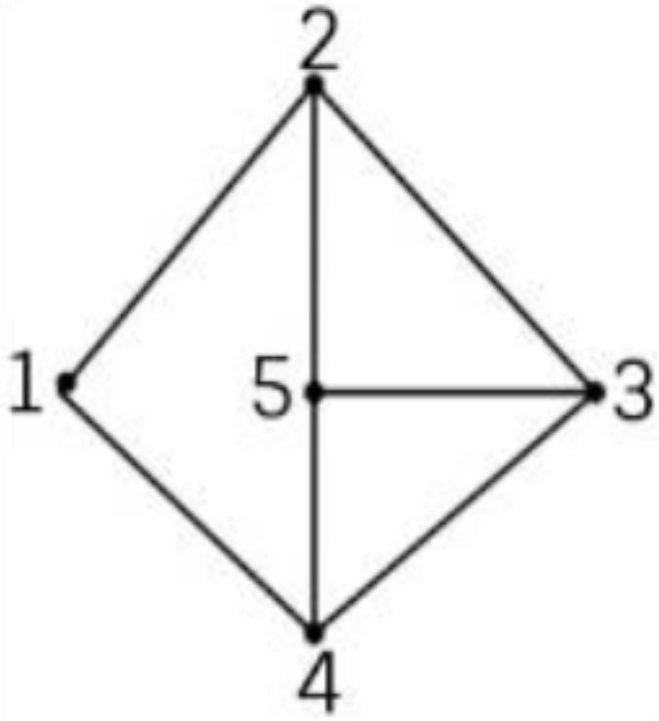
OUTPUT:

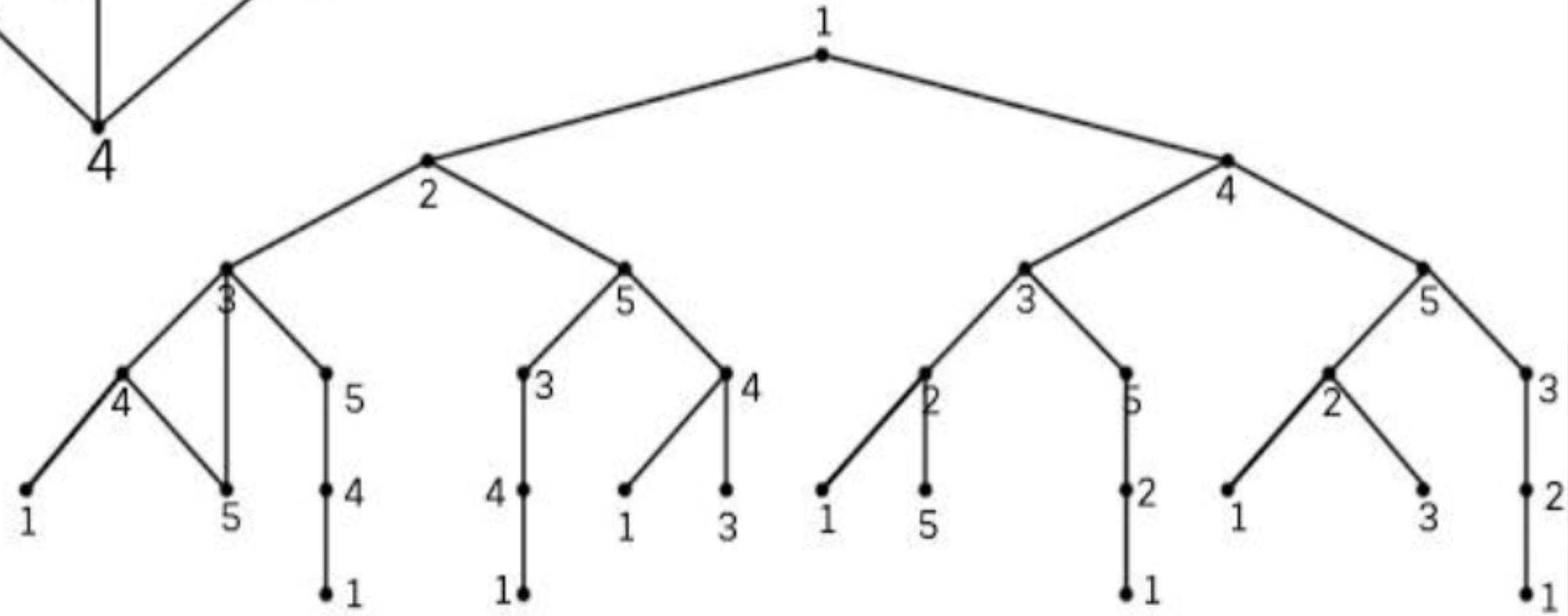
* O*utput* (A, b): sử dụng nghiệm b của A sao cho phù hợp với bài toán

ACTION:

```
b ← first(A)
while b ≠ Λ do
    if valid(A, b) then
        output(A, b)
    b ← next(A, b)
end while
```

Đường đi Hamilton





QUESTION 1:

Cho một mảng số nguyên A = [−2, 4, 1, −8, 6, −5].
Tìm dãy con liên tiếp có tổng các phần tử là lớn nhất.

```
def max_subarray(A):
    max_so_far = max_ending_here = A[0]
    for x in A[1:]:
        max_ending_here = max(x, max_ending_here + x)
        max_so_far = max(max_so_far, max_ending_here)
    return max_so_far
```

**** QUESTION 2:***

Cho một mảng số nguyên A bất kì.
Tìm tích lớn nhất của 3 phần tử bất kì trong mảng A.

Ví dụ:

INPUT	OUTPUT
9 5 10 3 7 1	630
-6 8 -1 1	48

TÀI LIỆU THAM KHẢO:

SÁCH: Anany Levitin, Introduction to the Design and Analysis of Algorithms, 3rd Edition, 2014

WEBSITE: https://www.geeksforgeeks.org/

INTERNET:

THANK YOU =))