BÀI TẬP LÝ THUYẾT 1

1. (2 điểm) Chứng minh định lý sau

$$f_1(n) \in O\big(g_1(n)\big) \wedge f_2(n) \in O\big(g_2(n)\big) \Rightarrow f_1(n) + f_2(n) \in O\left(\max\{g_1(n), g_2(n)\}\right)$$

2. (2 điểm) Chứng minh:

$$\begin{bmatrix} F(n+1) & F(n) \\ F(n) & F(n-1) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n \text{ với } n \geq 1.$$

3. (2 điểm) Xét giải thuật sắp xếp nổi bọt cải tiến sau:

```
impBubbleSort(a[1 .. n]) {
   flag = true;
   m = 1;
   while (flag) {
      flag = false;
      m++;
      for (j = n; j ≥ m; j--)
         if (a[j - 1] > a[j]) {
            a[j - 1] ⇆ a[j];
            flag = true;
         }
   }
}
```

Hãy phân tích độ phức tạp.

4. (2 điểm) Xét giải thuật sắp xếp chèn trực tiếp:

```
InsertionSort(a[1 .. n]) {
   for (i = 2; i ≤ n; i++) {
      v = a[i];
      j = i - 1;
      while (j ≥ 1) && (a[j] > v) {
         a[j + 1] = a[j];
         j--;
      }
      a[j + 1] = v;
   }
}
```

Hãy phân tích độ phức tạp của giải thuật này.

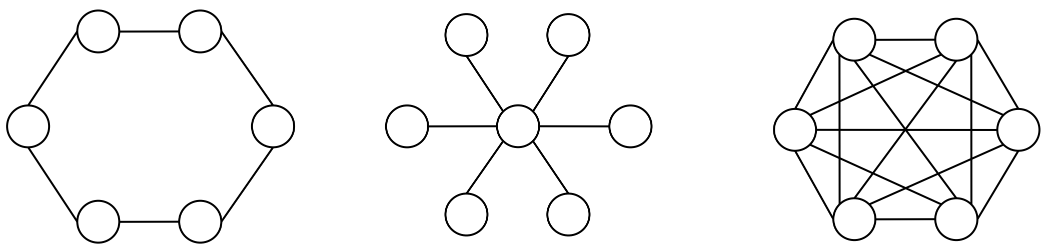
5. (2 điểm) Đánh gía độ phức tạp của giải thuật sau:

```
MaxContSubSum(a[1 .. n]) {
   maxSum = 0;
   for (i = 1; i ≤ n; i++)
      for (j = i; j ≤ n; j++) {
         curSum = 0;
         for (k = i; k ≤ j; k++)
            curSum += a[k];
         if (curSum > maxSum)
            maxSum = curSum;
      }
   return maxSum;
}
```

BÀI TẬP LÝ THUYẾT 2

1. Tính toán độ phức tạp của hàm `Taohoanvi()` được trình bày trong chương 3, phần tìm kiếm vét cạn.

2. Thiết kế của một mạng cục bộ dùng để kết nối các thiết bị văn phòng như máy tính, máy in, máy fax, … thường có một trong ba hình thức sau: dạng vòng (*ring*), dạng sao (*star*), dạng lưới đầy đủ (*fully connected mesh*)



Dạng vòng *Dạng sao* *Dạng lưới đầy đủ*

Gọi *G* là ma trận kề, biểu diễn một mạng cục bộ nêu trên dưới dạng đồ thị vô hướng. Hãy xây dựng một giải thuật với tiếp cận *brute-force* để cho biết *G* thể hiện dạng thức liên kết nào và chi phí của giải thuật này.

3. Bài toán phân hoạch (*partition problem*): Cho n số nguyên dương. Cần phân hoạch chúng thành hai tập con tách rời sao cho tổng các phần tử giữa hai tập là bằng nhau. Tất nhiên, bài toán đưa ra không phải lúc nào cũng có nghiệm. Hãy thiết kế giải thuật tìm kiếm vét cạn (hiệu quả nhất có thể) cho bài toán này.

4. Ma phương (hay hình vuông ma thuật, ma trận kỳ ảo - *magic square*) bậc n là ma trận vuông $n \times n$ chứa các số từ 1 đến n^2 sao cho tổng của mỗi dòng, tổng của mỗi cột và tổng của mỗi đường chéo (chính, phụ) đều bằng nhau. Gọi tổng này là t.

   a. Chứng minh $t = \frac{n(n^2+1)}{2}$

   b. Thiết kế giải thuật tìm kiếm vét cạn cho phép phát sinh mọi ma phương bậc n.

BÀI TẬP LÝ THUYẾT 3

1. (4 điểm) Bài toán tìm đồng thời số nhỏ nhất và lớn nhất trong mảng một chiều có n phần tử.

   a. Cho rằng chiến lược thiết kế Chia để trị được sử dụng. Phân tích độ phức tạp (không sử dụng định lý chủ) cho các trường hợp sau:

   - Phép so sánh là thao tác cơ sở.
   - Phép so sánh `if (l >= r - 1)` cũng được xem là thao tác cơ sở.

   b. Chỉ ra một tiếp cận có cùng độ phức tạp với Chia để trị nhưng không sử dụng kỹ thuật lập trình đệ qui. Phân tích độ phức tạp khi xem phép so sánh là thao tác cơ sở.

2. (3 điểm) Phân tích giải thuật Merge sort (không sử dụng định lý chủ) khi xem

   a. Phép so sánh là thao tác cơ sở.

   b. Phép chuyển dời là thao tác cơ sở.

3. (3 điểm) Giả sử cần sắp xếp mảng theo thứ tự tăng dần bằng giải thuật Merge sort. Lúc này, với sự điều chỉnh thích hợp ở mã nguồn, trường hợp tốt nhất có thể xảy ra với chi phí tuyến tính. Hãy cho biết:

   a. Phân bố dữ liệu phải thoả điều kiện gì?

   b. Mã nguồn đã điều chỉnh như thế nào?

BÀI TẬP LÝ THUYẾT 4

1. Bài toán "Tìm cặp điểm gần nhất", sử dụng tiếp cận Chia để trị

   Khi đi tìm cặp điểm trong vùng S đối xứng qua trục ℓ, một ước lượng thô được phát biểu như sau:

   "Tồn tại tối đa 6 điểm trong vùng tìm kiếm diện tích $2\delta \times \delta$".

   Hãy chứng minh phát biểu này.

2. Phân tích độ phức tạp của bài toán "Đổi tiền xu" khi được xử lý bằng kỹ thuật Chia để trị.

3. Nhân hai số nguyên dương n bit (cho rằng, n là lũy thừa của 2).

Gọi x và y là hai số nguyên n bit. Gọi x_L, x_R lần lượt là số nguyên tương ứng với $^n/_2$ bit bên trái và $^n/_2$ bit bên phải của x. Tương tự, ta cũng có y_L, y_R. Khi đó:

$$x = 2^{n/2} x_L + x_R$$

$$y = 2^{n/2} y_L + y_R$$

Ví dụ: Cho rằng $x = 135_{10} = 10000111_2$. Như vậy, $x_L = 8_{10} (= 1000_2), x_R = 7_{10} (= 0111_2)$ và:

$$x = 2^{n/2} x_L + x_R = 2^{8/2} \times 8_{10} + 7_{10}$$

Tích $x \times y$ có thể được viết như sau:

$$x \times y = (2^{n/2} x_L + x_R) \times (2^{n/2} y_L + y_R) = 2^n \times x_L y_L + 2^{n/2} \times (x_L y_R + x_R y_L) + x_R y_R$$

Nhận thấy rằng,

$$x_L y_R + x_R y_L = (x_L + x_R) \times (y_L + y_R) - (x_L y_L + x_R y_R)$$

nên số phép nhân giảm chỉ còn 3. Do đó, $T(n) \in \Theta(n^{1.585})$.

Viết hàm bằng ngôn ngữ C/C++ thực hiện ý tưởng trên với chi phí $\Theta(n^{1.585})$. Mẫu của hàm là:

```
int Multiplication(int x, int y);
```

BÀI TẬP LÝ THUYẾT 5

1. Bài toán Josephus
   a. Chứng minh công thức sau:
   $$J(2^k + i) = 2i + 1, i \in [0, 2^k - 1]$$
   b. Chứng minh phát biểu: "Đảo vòng 1 bit sang trái đối với biểu diễn nhị phân của n sẽ cho ra $J(n)$".
2. Bài toán Chọn (*Selection problem*)

   Hoàn chỉnh việc đánh giá độ phức tạp của giải thuật trong trường hợp trung bình.
3. Giải thuật sau có nhiệm vụ kiểm tra tính liên thông của đồ thị vô hướng, vốn được biểu diễn bởi ma trận kề. Nó sẽ trả về $true$ nếu đây là đồ thị liên thông, ngược lại thì trả về $false$.

```
Connected(A[1 .. n][1 .. n]) {
  if (n == 1)  return true; // Đồ thị có 1 đỉnh thì liên thông
  else
    if (!Connected(A[1 .. n - 1][1 .. n - 1]))   return false;
    else {
      for (i = 1; i ≤ n – 1; i++)
        if (A[n][i])   return true;
      return false;
    }
}
```

Liệu giải thuật có thực thi đúng nhiệm vụ của nó hay không? Nếu cho rằng đúng thì hãy chỉ ra độ phức tạp trong trường hợp xấu nhất. Nếu cho rằng sai thì giải thích tại sao?

4. Tìm kiếm

   Cho mảng $A = \{a_1, a_2, \dots, a_n\}$ thoả (i) $\forall i, j \in [1, n]: a_i \neq a_j$ khi $i \neq j$, (ii) $\forall i \in [1, n]: a_i \in [1, n + 1]$ và (iii) $\forall i \in [1, n - 1]: a_i < a_{i+1}$. Hãy thiết kế giải thuật chi tiết (dưới dạng mã giả) có nhiệm vụ tìm con số vắng mặt sao cho hiệu quả nhất; đồng thời, phân tích độ phức tạp của giải thuật.