BÀI TẬP 1 - ĐỘ PHỨC TẠP TÍNH TOÁN CỦA THUẬT TOÁN

Bài 1:

Cho dãy A gồm n phần tử số nguyên: A[1], A[2], …, A[n]. Hãy viết các thuật toán sau (hàm/ chương trình con), từ đó xác định độ phức tạp tính toán của các thuật toán mà bạn đã viết:

1. Tìm giá trị lớn nhất của dãy A.

2. Đảo ngược dãy A này. Ví dụ: Nếu dãy A=(5, 2, 4), thì sau khi đảo ngược ta có:
A=(4, 2, 5).

3. Kiểm tra dãy A có đối xứng hay không. Ví dụ: Dãy A=(4,2,4) hoặc A=(4,1,1,4) đều là các dãy đối xứng.

4. Đếm số các cặp nghịch đảo trong A. Biết rằng, một cặp nghịch đảo trong A là cặp (A[i], A[j]) sao cho i<j và A[i]>A[j]. Ví dụ: Dãy (5, 2, 4) có 2 cặp nghịch đảo là:
(5, 2) và (5, 4).

5. Đếm các cặp trong A sao cho có tổng bằng một số nguyên x cho trước. Ví dụ: Cho dãy A=(3, 1, 2, 4, 2) và x = 5; khi đó có 3 cặp trong A có tổng bằng x là:
(A[1], A[3]), (A[1], A[5]) và (A[2], A[4]).

6. In ra dãy D gồm n phần tử số nguyên D[1], D[2], …, D[n], với: D[i] là số các phần tử ở bên phải của phần tử A[i] thuộc dãy A và có giá trị nhỏ hơn hoặc bằng A[i], với mọi i nguyên thuộc [1, n]. Ví dụ: Dãy A=(20, 1, 2, 0, 15, 1) thì dãy D=(5, 2, 2, 0, 1, 0).

Bài 2:

Xác định độ phức tạp thuật toán của hai thuật toán tính e^x như sau. Từ đó cho biết thuật toán nào là tối ưu hơn?

Thuật toán 1

```
float TT1(float  x, int n)
{
float  S = 1;
float  p = 1;
for (int i = 1; i <= n; i++){
      for (int j = 1;  j <= i;  j++){
            p = p*x/ j;
```

Thuật toán 2

```
float  TT2(float  x, int n)
{
float  S = 1;
float  p = 1;
for (int i = 1; i <= n; i++) {
      p = p*x/ i;
      S += p;
```

```
                S += p;
        }
}
return S;
}
```

```
}
return S;
}
```