**Câu 1:**

* Phép gán:

Số phép gán cho vòng lặp m:

Lần 0 thì thực hiện 1 phép gán.

Lần 1 thì thực hiện 2 phép gán.

Lần 2 thì thực hiện 3 phép gán.

…

Lần m thì thực hiện (m+1) phép gán.

Số phép gán cho vòng lặp n:

Lần 0 thì thực hiện m+1(for m)\*0+1 phép gán.

Lần 1 thì thực hiện m+1(for m)\*1+1 phép gán.

Lần 2 thì thực hiện m+1(for m)\*2+1 phép gán.

…

Lần n thì thực hiện m+1(for m)\*n+1 phép gán.

T(m,n)=(m+1)\*n+1;

Với m=n thì:

=>> T(n)=(n+1)\*n+1=O(n2)

* Phép so sánh:

Số phép so sánh cho vòng lặp m:

Lần 0 thì thực hiện 1 phép so sánh.

Lần 1 thì thực hiện 2 phép so sánh.

Lần 2 thì thực hiện 3 phép so sánh.

…

Lần m thì thực hiện (m+1) phép so sánh.

Số phép so sánh cho vòng lặp n:

Lần 0 thì thực hiện m+1(for m)\*0+1 phép so sánh.

Lần 1 thì thực hiện m+1(for m)\*1+1 phép so sánh.

Lần 2 thì thực hiện m+1(for m)\*2+1 phép so sánh.

…

Lần n thì thực hiện m+1(for m)\*n+1 phép so sánh.

T(m,n)=(m+1)\*n+1;

Với m=n thì:

=>> T(n)=(n+1)\*n+1=O(n2)

=>> Độ phức tạp của giải thuật trên là O(n2)

**Câu 2:**

* **Phép gán:**

Số phép gán cho vòng lặp A=for(j=0;j<i;j++)

Với i=0 thì có 1 phép gán.

Với i=1 thì có 3 phép gán.

Với i=2 thì có 5 phép gán.

…

Với i=k thì có 2k+1 phép gán.

Số phép gán cho vòng lặp for (i = 0;i < n; i++)

Với n=0 thì có 1 phép gán và 0\*A

Với n=1 thì có 1 phép gán và 1\*A

Với n=2 thì có 1 phép gán và 2\*A

…

Với n=k thì có k+1 phép gán và k\*A

=>>(k+1) + k(2k+1).

=>>2k2+2k+1

* **Phép so sánh:**

Số phép so sánh cho vòng lặp m:

Với i=0 thì thực hiện 1 phép so sánh.

Với i=1 thì thực hiện 2 phép so sánh.

Với i=2 thì thực hiện 3 phép so sánh.

…

Với i=k thì thực hiện (k+1) phép so sánh.

Số phép so sánh cho vòng lặp n:

Với n=0 thì thực hiện 1 phép so sánh và 0 vòng lặp for(i)

Với n=1 thì thực hiện 2 phép so sánh và 1 vòng lặp for(i)

Với n=2 thì thực hiện 3 phép so sánh và 2 vòng lặp for(i)

…

Với n=k thì thực hiện k+1 phép so sánh và k vòng lặp for(i)

=>>(k+1) + k\*(k+1).

=>>k2+2k+1

=>> Độ phức tạp của giải thuật trên là O(n2)

**Câu 3:**

Trong vòng lặp for thứ nhất: i chạy từ 0 đến (n-1)=>> T(n)=O(n)

Trong vòng lặp for thứ hai: i chạy từ 0 đến (n2-1)=>>T(n)=O(n2)

=>> Độ phức tạp của đoạn code là O(n2) .(chọn độ phức tạp cao nhất của 2 vòng lặp for liên tiếp nhau ).

**Câu 4:**

F(n) có 2n-1 lần gọi đệ lại đệ qui .

* O(2n-1) = O(2n)

Câu 5: Đánh giá độ phức tạp của dãy FIBONACCI

F(n) có 2n-1 lần gọi đệ lại đệ qui .

* O(2n-1) = O(2n)