Bài tập 1

Nhập môn phân tích độ phức tạp thuật toán

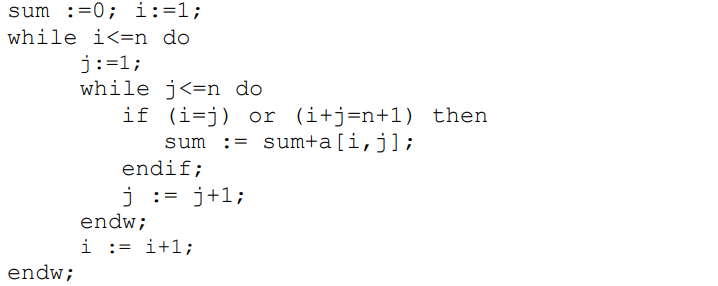
Nguyễn Minh Đức

1712358

Yêu cầu 1

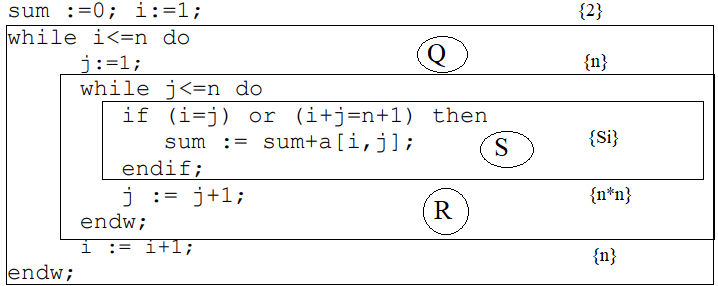
Em chọn câu 3 câu 5

Câu 3



Đếm số phép gán:

Ta phân đoạn thuật toán rồi đếm các phép gán trong mỗi đoạn



Suy ra tổng số phép gán của thuật toán

Tính Si

Vì j luôn được gán bằng 1 rồi chạy tới n rồi lại về 1 nên luôn luôn tìm được 1 giá trị sao cho i=j (và chỉ tìm được 1 giá trị duy nhất khi i không đổi), và cũng luôn tìm được 1 giá trị sao cho i+j=n+1 (cũng chỉ tìm được giá trị duy nhất khi i không đổi). Giả sử n là chẵn, khi đó n+1 là lẻ, nên 2 giá trị trên không thể có cùng j. Suy ra khi n chẵn số, thì cứ sau 1 vòng lặp Q, thì lệnh if trong S được thực hiện thành công 2 lần.

=> Si = 2\*n =>

Còn khi n lẻ thì chỉ tồn tại duy nhất 1 giá trị i để i=j và i+j=n+1

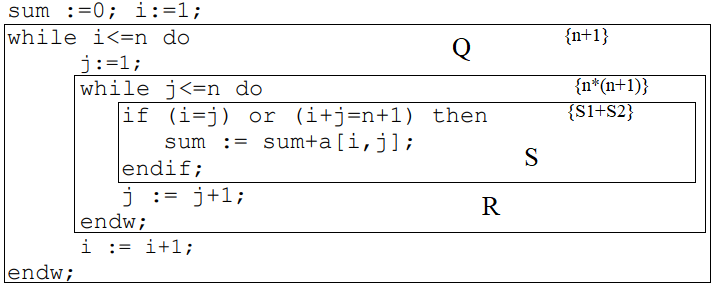
Nên suốt chương trình 2 điều kiện này chỉ trùng nhau 1 lần

=>

Tựu lại ta được

Đếm số phép so sánh:

Cũng phân đoạn thuật toán, rồi đếm số phép so sánh trong từng đoạn

S

Suy ra tổng số phép so sánh của chương trình

Tính S1+S2

Ta thấy rằng nếu điều kiện i=j thỏa thì không cần phải so sánh i+j=n+1.

Số lần i=j thỏa là n lần. Ngoài ra, nếu i=j không thỏa thì ta chắc chắn phải so sánh 2 lần.

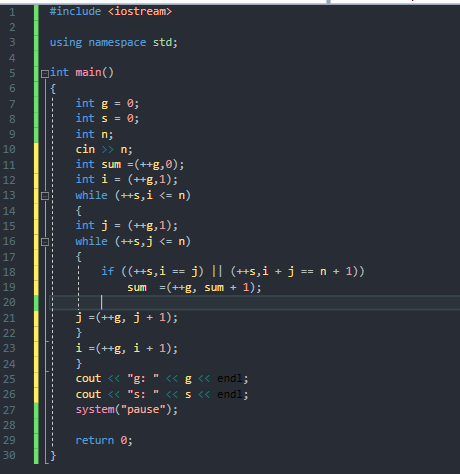
=>

=>

So với chương trình thực nghiệm

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Tính toán lý thuyết | | Chương trình thực nghiệm | |
| n | G | S | G | S |
| 10 | 142 | 311 | 142 | 311 |
| 21 | 526 | 1345 | 526 | 1345 |
| 100 | 10402 | 30101 | 10402 | 30101 |
| 1001 | 1006006 | 3007005 | 1006006 | 3007005 |

source code

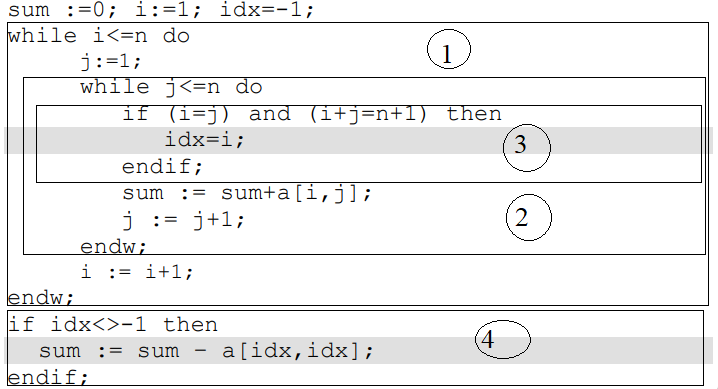


Vậy tính toán lý thuyết của hoàn toàn trùng khớp với chương trình thực nghiệm.

=> Độ phức tạp thuật toán trên là

Câu 5

Phân đoạn thuật toán, ta được



Số phép gán(g) và so sánh(ss):

Ngoài các vòng lặp: g = 3, ss = 0

(1): g = 2n, ss =n + 1

(2): g = 2n2, ss = n(n + 1) = n2 + n

(3):

i chạy từ n về 1 và j từ 1 lên n nên luôn có 1 giá trị của j để j = i.

Tương tự ta cũng có 1 giá trị của j để i + j = n + 1.

Nếu i = j và i + j = n + 1 -> i = (n + 1) / 2 -> n lẻ. Ta có:

Nếu n lẻ:

g = 1, ss = n2 + 1

Nếu n chẵn:

g = 0, ss = n2

(4):

Nếu n lẻ, (4) -> idx = i -> idx <> -1:

g = 1, ss = 1

Nếu chẵn, (4) -> idx = -1:

g = 0, ss = 1

1. + (2) + (3) + (4) + (5):

n lẻ:

g = 3 + 2n + 2n2 + 1 + 1 = 2n2 + 2n + 5

ss = 0 + n + 1 + n2 + n + n2 + 1 + 1= 2n2 + 2n + 3

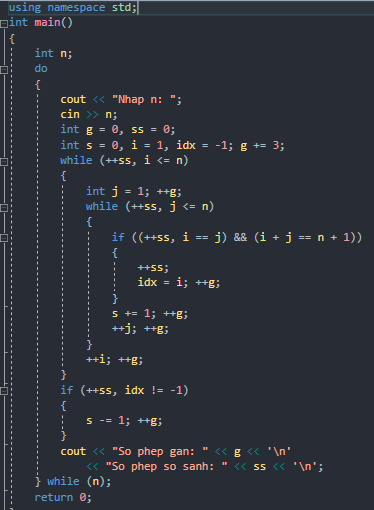
n chẵn:

g = 3 + 2n + 2n2 = 2n2 + 2n + 3

ss = 0 + n + 1 + n2 + n + n2 + 1 = 2n2 + 2n + 2

Vậy nếu n lẻ sẽ có 2n2 + 2n + 5 phép gán và 2n2 + 2n + 3 phép so sánh, nếu n chẵn sẽ có 2n2 + 2n + 3 phép gán và 2n2 + 2n + 2 phép so sánh.

Soure code thực nghiệm



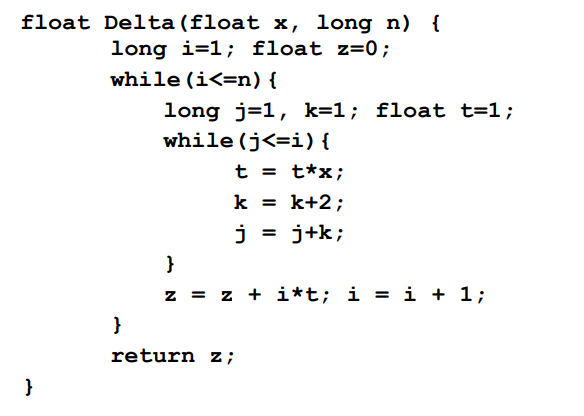
Kết quả cho thấy tính toán lý thuyết hoàn toàn trùng khớp với thực nghiệm

Từ công thức trên suy ra độ phức tạp thuật toán là O(

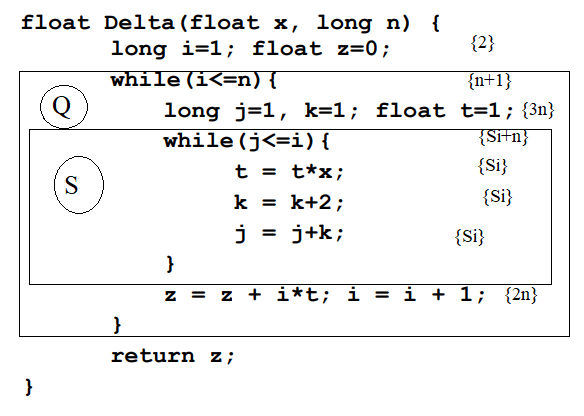
Yêu cầu 2:

Em chọn câu 9 câu 10

Câu 9



Phân đoạn chương trình, ta được:



Số phép gán trong chương trình

Số phép so sánh trong chương trình

Tính Si:

Ta thấy rằng k luôn là 1 dãy số lẻ liên tục 1,3,5,7,.. (do k=k+2 và k ban đầu bằng 1).

=> j luôn được tăng thêm 1 số lẻ liên tục lần trước khi lớn hơn i.

=> Vào vòng lặp cuối cùng thì

Giả sử giá trị k trong vòng lặp cuối bằng k0 và Sj là số lần lặp khi i không đổi thì

=>

Số lần lặp sẽ là số tự nhiên lớn nhất sao cho bình phương của nó nhỏ hơn hoặc bằng i.

Tổng lại ta có

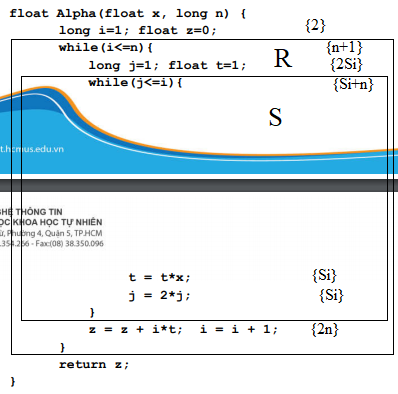
=>

=>

Vậy thuật toán có độ phức tạp O(

Câu 10:

Phân đoạn chương trình, ta được:



Số phép gán trong chương trình

Số phép so sánh trong chương trình

Tính Si:

Để ý rằng j luôn bắt đầu các lũy thừa của 2 liên tục cho tới khi lớn hơn i.

Gọi số lần lặp của S khi i chưa thay đổi là Sj => j sẽ được nhân 2 Sj lần

=>

Từ đó suy ra số lần lặp Sj là số tự nhiên lớn nhất sao cho 2 mũ số đó nhỏ hơn hoặc bằng j và bằng .

Tổng lại ta có

=>

=>

Vậy thuật toán có độ phức tạp