**EX1**

//3

sum = 0;

for (i = 0; i < n; i++)

for (j = i + 1; j <= n; j++)

for (k = 1; k < 10; k++)

sum = sum + i \* j \* k;

T(5)=O(1)

T(4)=O(1)

T(45)= O(1)\*O(1)= O(1)

T(3)= O(n)

T(345)= O(n)\*O(1)=O(n^2)

T(2)=O(n)

T(2345)=O(n)\*O(n)=O(n^3)

T(1)=O(1)

T(12345)=O(1)\*O(n^3)=O(n^3)

Sai 3 ý (1)

//5

sum = 0;

thisSum = 0;

for (i = 0; i < n; i++) {

thisSum += b[i];

if (thisSum > sum)

sum = thisSum;

else

thisSum = sum;

}

T(6)=O(1)

T(8)=O(1)

T(5678)=O(1)

T(4)=O(1)

T(45678)=O(1)+O(1)=O(1)

T(3)=O(n)

T(345678)=O(n)\*O(1)=O(n)

T(1)=O(1)

T(2)=O(1)

T(12345678)=O(1)+O(1)+O(n)=O(n)

Đúng (2)

**Ex2:**

//3

sum = 0; {2 g}

i = 1;

while(i<=n) { {n+1 ss}

j = n-i; {1 g}

while(j<=i) {

sum = sum+j; {2 g}

j=j+1;

}

i=i+1; {1 g}

}

Ở vòng lặp while trong chỉ được thực hiện khi i >= n/2, cứ mỗi lần lặp, khoảng cách của j và i tăng lên 2 đơn vị.

Thiếu điều kiện để giải

Số phép so sánh: n + 1 +

Số phép gán: 2 + n\*( 2 +

Quá tắt

(1.5)

//4

s = 0; {2 g}

i =1;

while(i<=n){ {n+1 ss}

j = 1; {1 g}

while(j<=i\*i) {

s = s + 1; {2 g}

j = j + 1;

}

i = i + 1; {1 g}

}

Vòng lặp while trong sẽ được thực hiện với số lần từ 1 -> i^2, vì i tăng 1 đơn vị trong vòng while ngoài, số lần while trong có thể tính là

1 + 4 + 9 +..+ n^2= n\*(n+1)\*(n+2)/6

* Số phép so sánh: (n+1) + ( n\*(n+1)\*(n+2)/6 + n) *//n cuối cùng là mỗi lần while trong đều cần lần so sánh cuối để kết thúc vòng lặp*

*Thiếu*

* Số phép gán: 2 + 2\*n + n\*(n+1)\*(n+2)/6

Thiếu \*2, 2n+1 không phải n+2

(1.75)

//5

sum = 0; {2 g}

i = 1;

while (i<=n) { {n+1 ss}

j = n - i\*i; {1 g}

while (j<=i\*i) {

sum = sum + i\*j; {2 g}

j=j+1;

}

i=i+1; {1 g}

}

Câu lệnh trong vòng lặp while thứ 2 chỉ được thực thi khi 2\*i2-n >= 0

Số phép so sánh: (n+1)\*(

Số phép gán: 2 + 2 \* n \* + 2 \*

Trình bày chưa rõ, tắt (1.5)