**Bài 3:**

**Câu 1:**

Một chương trình con P (hàm hoặc thủ tục) được gọi là đệ quy nếu:

A. Trong quá trình thực hiện P có phần phải gọi đến chính P.

B. Trong quá trình thực hiện P phải gọi đến P hai lần.

C. Trong quá trình thực hiện P phải gọi đến P hai lần trở lên.

D. Trong quá trình thực hiện P không cần phải gọi đến P.

**Câu 2:**

Cấu trúc chính của một chương trình con đệ quy gồm:

A. Phần cơ sở và phần đệ quy.

B. Phần không cơ sở và phần không đệ quy.

C. Phần cơ sở và phần không có đệ quy.

D. Phần không cơ sở và phần đệ quy.

**Câu 3:**

Cho hàm đệ quy int power(int n)

{ if (n == 0) return 1;

else return 2 \* power(n-1); }.

Giá trị của power(3) được tính bằng:

A. 4

B. 10

C. 8

D. 6

**Câu 4:**

Hàm đệ quy dưới đây thực hiện chức năng gì?

int US(int a, int b) {

if (b == 0)

return a;

else

return US(b, a % b);

}

A. Tính tổng của a và b.

B. Tìm ước chung lớn nhất của a và b.

C. Tìm bội chung nhỏ nhất của a và b.

D. Tính hiệu của a và b.

**Câu 5:**

Hàm đệ quy dưới đây tính gì?

int sum(int n) {

if (n == 0)

return 0;

else

return n + sum(n - 1);

}

A. Tổng các số từ 1 đến n.

B. Tích các số từ 1 đến n.

C. Lũy thừa của n.

D. Tổng các số từ 1 đến n – 1.

**Câu 6:**

Hàm đệ quy sau thực hiện chức năng gì?

int countt(int n) {

if (n <= 0)

return 0;

else {

printf("%d ", n);

return countt(n - 1);

}

}

A. Đếm số lượng các số từ n đến 1 và in ra.

B. Tính tổng các số từ 1 đến n.

C. Tính giai thừa của số nguyên n.

D. In tất cả các số từ n, n-1, ... đến 1.

**Câu 7:**

Độ phức tạp của thuật toán đệ quy tính số hạng thứ n của dãy Fibonacci:

int fibo(int n) {

if (n <= 2)

return 1;

return fibo(n-1) + fibo(n-2);

}

Có kết quả bằng:

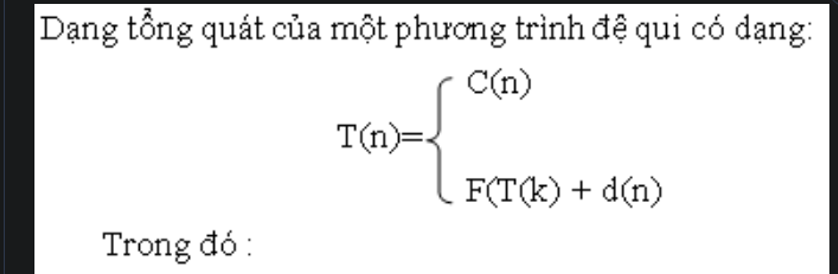
A. O(1)

B. O(2^n)

C. O(n!)

D. O(n^2).

***Câu 8:***



A.

C(n): là thời gian thực hiện chương trình.

F(T(k)) là một đa thức của các T(k).

d(n): là thời gian để phân chia bài toán.

B.

C(n): là thời gian thực hiện chương trình ứng với trường hợp đệ quy dừng.

F(T(k)) là một đa thức của các T(k).

d(n): là thời gian để phân chia bài toán và tổng hợp các kết quả.

**Câu 9:**

Cho hàm đệ quy:

int US(int a, int b) {

if (b == 0)

return a;

else

return US(b, a % b);

}

Giá trị của US(28, 20) được tính bằng:

A. 2

B. 4

C. 6

D. 8.

**Câu 10:**

Cho hàm đệ quy:

int sum(int n) {

if (n == 0)

return 0;

else

return n \* n + sum(n - 1);

}

Gọi hàm sum(3) trả về giá trị:

A. 20

B. 30

C. 14

D. 16.

**Câu 11:**

Độ phức tạp của thuật toán đệ quy thực hiện tìm kiếm giá trị x trong dãy có n phần tử theo phương pháp nhị phân:

int bsearch(int a[], int L, int R, int x) {

if (L > R)

return -1;

int k = (L + R) / 2;

if (x == a[k])

return k;

if (x > a[k])

return bsearch(a, k + 1, R, x);

return bsearch(a, L, k - 1, x);

}

Có kết quả bằng:

A. O(n log n)

B. O(log n)

C. O(n)

D. O(n^2).

**Câu 12:**

Thuật toán có phương trình đệ quy sau, (T(1)=1):

T(n) = 2T(n/2) + n

Được giải theo định lý Master có nghiệm là:

A. O(n)

B. O(n^2)

C. O(log n)

D. O(n log n).

**Câu 13:**

Thuật toán có phương trình đệ quy sau, (T(1)=1):

T(n) = 16T(n/4) + n

Được giải theo định lý Master có nghiệm là:

A. O(n)

B. O(n^2)

C. O(n log n)

D. O(log n).

**Câu 14:**

Thuật toán có phương trình đệ quy sau, (T(1)=1):

T(n) = 8T(n/2) + n

Được giải theo định lý Master có nghiệm là:

A. O(n^3)

B. O(n^2)

C. O(n^2 log n)

D. O(n).

**Câu 15:**

Với phương trình đệ quy T(n) = 8T(n/2) + n^3, hãy xác định bộ giá trị của (a, b, k) và nghiệm tính được theo định lý Master:

A. (8, 2, 3); O(n log n^2)

B. (8, 1, 3); O(n^3)

C. (8, 2, 3); O(n^3 log n)

D. (8, 1, 3); O(log n^3).

**Câu 16:** Khi đánh giá độ phức tạp của chương trình con đệ quy tính n! (n>=0), phương trình đệ qui được thành lập

có dạng:

A: c1 là hằng số

C1 nếu n=0

T(n-1)+T(n-2) nếu n>0

B: c1, c2 là hằng số

C1 nếu n=0

2T(n-1)+C2 nếu n>0

C 2T(n/2) + C2 nếu n>0

D T(n-1)+C2 nếu n>0

C1, C2 là các hằng số

**Câu 17:** Giá trị của power (2, 3) với đoạn chương trình sau, được tính bằng

int power(int base, int exp) {

if (exp = = 0) return 1;

else return base \* power(base, exp-1);

}

A 6

B 8

C 4

D 2

**Câu 18:** Cho Hàm đệ quy

int fibo(int n) {

if (n <= 2) return 1;

else return fibo(n-1) + fibo(n-2);

}

giá trị của fibo(5) được tính bằng:

A 8

B 3

C 13

D 5

**Câu 19:** Thuật toán có phương trình đệ quy sau, (T(1)=1)

T(n) = T(5n/9) + 1

được giải theo định lý Master có nghiệm là:

A O(n2)

B O(nlogn)

C O(logn)

D O(n)

**Câu 20:** Thuật toán có phương trình đệ quy sau, (T(1)=1)

T(n)=3T(n/3)+n

được giải theo định lý Master có nghiệm là:

A O(n2)

B O(logn)

C O(nlogn)

D O(n)

**Câu 21:** Thuật toán có phương trình đệ quy sau, (T(1)=1)

T(n)=4T(n/2)+n2

được giải theo định lý Master có nghiệm là:

A O(n2)

B O(nlogn)

C O(n2logn)

D O(n)

**Câu 22:**

Độ phức tạp của thuật toán đệ quy thực hiện sắp xếp các phần tử của một dãy có

𝑛 phần tử theo chiều tăng dần bằng phương pháp Quicksort trong trường hợp tốt nhất được xác định bởi phương trình sau:

Quicksort (int a[], int L, int R) {

k = Partition(a, L, R);

Quicksort(a, L, k-1);

Quicksort(a, k, R);

}

Độ phức tạp của thuật toán trong trường hợp tốt nhất là:

1. 𝑂(𝑛)
2. O(nlogn)
3. O(n^2)
4. O(2^n)

Câu 23: Phần cơ sở của thuật toán đệ qui bao gồm

A

các trường hợp dừng mà có thể trực tiếp

giải quyết được ngay

B

các trường hợp chưa dừng lại mà có thể

giải quyết trong trường hợp đặc biệt

C

các trường hợp chưa dừng nhưng có thể

trực tiếp giải quyết được

D

các trường hợp dừng nhưng chưa thể

trực tiếp giải quyết được

**Câu 23: Đặc điểm của chương trình con đệ qui :**

A

1. Trong chương trình con đệ quy có lời gọi đến chính nó

2.Mỗi lan có loi goi thì kích thuớc cua bai toan đa thu nho hơn trướ

3.Có một trường hợp đặc biệt và gọi đệ qui được tiếp tục

B

1. Trong chương trình con đệ quy có lời gọi đến chính nó

2.Mỗi lan co loi goi thì kích thuoc cua bai toan được thu nho đi một

nứa

3.Có một trường hợp đặc biệt, là trường hợp dừng của thuật toán và gọi

đệ qui cũng kết thúc

C

1. Trong chương trình con đệ quy có lời gọi đến chính nó

2.Mỗi lần có lời gọi thì kch thước cua bai toán được thu nhỏ đi một

nửa

3.Có một trường hợp đặc biệt và gọi đệ qui được tiếp tục

D

1. Trong chương trình con đệ quy có lời gọi đến chính nó

2.Mỗi lần có lời gọi thì kích thước của bài toán đã thu nhỏ hơn trước

3.Có một trường hợp đặc biệt, là trường hợp dừng của thuật toán và gọi

đệ qui cũng kết thúc

**câu 24**: Hàm đệ quy facto được định nghĩa như sau:

int facto(int n) {

if (n == 0) return 1;

else return n \* facto(n-1);

}

A

Bình phương của

B

Tổng các số từ 1 đến n

C

Trung bình cộng của các số từ 1 đến n

D

Giai thua cua n

**Câu 25**: Hàm đệ quy sau thực hiện chức năng gì?

int sum(int n) {

if (n <= 0) return 0;

else

if (n % 2 = = 0) return n + sum(n- 2);

else return sum(n - 1);

A

Tính tổng các số nguyên tố từ 1 đến n

B

Tính tổng các số chẵn từ 1 đến n

C

Tính tổng tất cả các số từ 1 đến n

D

Tính tổng các số lẻ từ 1 đến n

**Bài 1**

Câu 1: Thuật toán là

A

một day các bước, mỗi bước mô tả chính xác các phép

toán hoặc hành động cần thực hiện để giải quyết một

vấn đề

B

một day hữu hạn cac bước, mỗi bước mô ta chính xác

các phép toán hoặc hành động cần thực hiện để giải

quyết một vấn đề

C

một day hữu hạn cac bước, mô ta chính xac các phep

toán hoặc hành động để giải quyết một vấn đề

D

một day hữu hạn cac bưoc, mo ta chính xac các phep

toán hoặc hành động cần thực hiện để giải quyết một

vấn đề

câu2 : Giải thuật là

A

cách giải quyết bài toán cho kết quả gần đúng (chấp

nhận được) đỡ phức tạp và có hiệu quả hơn

B

cách giải quyết bài toán cho kết quả có hiệu quả hơn

C

cách giải quyết bài toán đảm bảo các đặc trưng của

thuật toán

D

cách giải quyết bài toán cho kết quả đúng

**câu 3:** Tính hữu hạn của thuật toán là

A

thuật toán bao giờ cũng phải dừng lại sau một số hữu

hạn bước

B thuat toan sẽ dừng lại sau mot số lần lặp các bước

C

thuật toán sẽ dừng lại sau một số bước thực hiện

D

thuật toán bao giờ cũng phải dừng lại sau một số vô

hạn bước thực hiện

**câu 4:** Giá về thời gian trên máy Turing là:

A

Thời gian để thực hiện các bước chuyển hình trạng từ

hình trạng đầu đến hình trạng cuối

B

Thời gian để thực hiện bước chuyển hình trạng đầu

C

Thời gian để thực hiện các bước chuyển hình trạng

trung gian

D

Thời gian để thực hiện bước chuyển hình trạng cuối

**Câu5**: Với máy xử lý thuật toán bằng ngôn ngữ tựa ALGOL, đơn vị nhớ là:

A

Một cho nhớ để chứa một dữ liệu ra

B

Một chỗ nhớ để chứa một dữ liệu

C

Một chỗ nhớ để chứa một dữ liệu vào

D

Một chỗ nhớ để chứa một kí hiệu

**Câu 6 :** Với bài toán: Xác định giá trị lớn nhất trong dãy có n số nguyên X=(x1, X2, ... ,Xn}, n là số nguyên dương. Hãy xác định kích thước của bài toán theo quan niệm thứ nhất:

A n2

B n+1

C nlogn

D n

**Câu 7:** Xác định Input, Output cho bài toán tìn kiếm tuần tự giá trị k trong dãy n số nguyên khác nhau X1, X2, ... , Xn

A

Input : số nguyên dương n, dãy n số nguyên khác nhau

X1, X2, ... , Xn, Số nguyên k

Output : Vị trí i mà xi = k hoac thông báo không tìm thấy

số nguyên k trong dãy

B

Input : day n số nguyên khác nhau X1, X2, ... , Xn, Số

nguyên k

Output : Vị trí i mà x = k hoac thông báo không tìm thấy

số nguyên k trong dãy

C

Input : day n số nguyên khác nhau x1, X2, ... , Xn, Số

nguyên k

Output : Vị trí i mà xị = k

D

Input: số nguyên dương n, day n số nguyên khác nhau

X1, X2, ... , Xn

Output : Vị trí i mà Xi = k hoac thông báo không tìm thấy

số nguyên k trong dãy

câu 8 : Xác định Input của bài toán : Hoán đổi giá trị của 2 biến số nguyên x và y và dùng biến trung gian số nguyên z

A

Hai biến số nguyên y, z

B

Hai biến số nguyên x, z

C

Hai biến số nguyên x, y

D

Ba biến số nguyên x, y, z

**Câu 9**: Phương pháp giả mã dùng để biểu diễn thuật toán là

A

mượn một ngon ngu lap trình nao đo đe viết chương

trình

B

mượn các cú pháp của một ngôn ngữ lập trình nào đó

để thể hiện thuật toán

C

dùng cấu trúc của một ngôn ngữ lập trình bậc cao để

viết chương trình

D

dùng cấu trúc của ngôn ngữ lập trình bậc thấp để mô

tả thuật toán

**câu 10**: Khi biểu diễn thuật toán bằng lưu đồ khối (sơ đồ khối), hình chữ nhật có ý nghĩa gì?

A

Thực hiện thao tác kiểm tra du liệu theo điều kiện để

phân nhánh thuật toán

B

Thực hiện thao tác nhập và xuất dữ liệu

C

Thực hiện thao tác xử lý du liệu (gán, các phép tính cơ

bản)

D

Thực hiện thao tác ghi và nhập dữ liệu

**Câu 11**: Với bài toán : Xác định giá trị lớn nhất trong dãy n số nguyên X={x1, X2, ... ,Xn}, n là số nguyên dương.

Hãy chọn biểu diễn Input, Output đúng :

A

Input : Dãy số nguyên X= {X1, X2, ... , Xn}, n nguyên dương.

Output: Tìm giá trị lớn nhất Max của dãy X

B

Input : Dãy số nguyên X= {x1, X2, ... , Xn}

Output: Tìm số giá trị lớn nhất

C

Input : Day số nguyên X= {x1, X2, ... , Xn}, n

Output: Tìm giá trị lớn nhất

D

Input : Day số nguyên X= {x1,X2, ... ,xn}

Output: Tìm gia trị lớn nhất Max của X

Câu 11 : Cho day số nguyên có n phần tử :X1, X2, ... , Xn. Neu thuật toán tìm thấy và đưa ra chỉ số thứ i đầu tiên thỏa man xi là số chan thì với

điều kiện nào thuật toán sẽ dừng :

A

X¡ là số chẵn

B

i>=n

C

Xi là số lẻ

D

i<n

câu 12: Cho day số nguyên có n phần tử X1, X2, ... , Xn và số nguyên k. Nếu thuật toán tìm thấy và đưa ra chỉ số thứ i đầu tiên thỏa mãn Xi =k

thì với điều kiện nào thuật toán sẽ dừng :

A

i>=n

B

Xi =k

C

i<n

D

Xj <>k

**Câu 13:** Độ phức tạp dữ liệu vào của bài toán theo quan niệm thứ nhất là

A

số lượng dữ liệu được xử lý của bài toán

B

số lượng dữ liệu đã được sử dụng vào của bài toán

số lượng dữ liệu trung gian của bài toán

D

số lượng dữ liệu vào của bài toán

câu 14: Lưu đồ khoi dùng để biểu diễn thuật toán là

A

một hệ thống các nút (nút giới hạn, nút thao tác, mũi

tên) được nối với nhau bởi các cung (mũi tên)

B

một hệ thống các nút (nút giới hạn, nút thao tác, nút

điều kiện) có hình dạng khác nhau theo qui ước, thể

hiện các chức năng khác nhau và được nối với nhau

bởi các cung (mũi tên)

C

một hệ thống các nút (nút giới hạn, nút thao tác, nút

điều kiện, mũi tên) thể hiện các chức năng khác nhau

và không được nối với nhau

D

một hệ thống các nút (nút giới hạn, nút điều kiện, mũi

tên) có hình dạng khác nhau theo qui ước, thể hiện các

chức năng khác nhau

**câu 15:** Với máy xử lý thuật toán bằng ngôn ngữ tựa ALGOL, đơn vị nhớ là:

A

Một chỗ nhớ để chứa một kí hiệu

B

Một chỗ nhớ để chứa một dữ liệu

C

Một chỗ nhớ để chứa một dữ liệu vào

D

Mot chỗ nhớ đe chứa một dữ liệu ra

Câu 16: Với bài toán: Xác định giá trị lớn nhất trong dãy n số nguyên X={x1, X2, ... ,Xn), n là số nguyên dương.

Hãy xác định kích thước của bài toán theo quan niệm thứ 2:

A

Σ[1og2xi]+log2n+n

B

Σ [1og2xi] +log2n+n+1

C

Σ [1og2 xi] +log211

D

Σ[logg2 xi]

i=1

câu 16: Biểu diễn thuật toán theo ngôn ngữ tự nhiên là

A

sử dụng ngôn ngữ thường ngày để lập chương trình

B

sử dụng ngôn ngữ chữ viết thường ngày

C

sử dụng ngôn ngữ chữ viết để ve thuật toán

D

sử dụng ngôn ngữ chữ viết thường ngày để liệt kê các

bước của thuật toán

câu 17 : Tính hiệu quả của thuật toán được đánh giá dựa trên các tiêu chuẩn:

A

Thời gian cần thiết để chạy chương trình

B

Dung lượng bộ nhớ của máy tính và thời gian chạy

chương trình

C

Dung lượng bộ nhớ cần có

D

Dung lượng bộ nhớ cần có và thời gian cần thiết để

chạy chương trình

cau 18: Trong biểu diễn một bài toán trên máy tính, Input là

A

Các dữ liệu vào của bài toán

B

Dữ liệu trong tính toán của bài toán

C

Dữ liệu trung gian của bài toán

D

Mot Dữ liệu vào của bài toán

**Bài 2**

**Câu 1** : Khi nói thời gian thực hiện của một chương trình là T(n) = Cn (C là hằng số) thì có nghĩa là

A

chương trình đó cần Cn dữ liệu vào

B

chương trình đó cần Cn dữ liệu tính toán

C

chương trình đó cần Cn chỉ thị thực thi

D

chương trình đó cần Cn dữ liệu ra

**caau2:** Cách đánh giá thời gian thực hiện thuật toán độc lập với máy tính và các yếu tố liên quan tới máy tính sẽ dẫn tới khái niệm gọi là

A

độ phức tạp của dữ liệu vào thuật toán

B

độ phức tạp không gian của thuật toán

C

độ phức tạp dữ liệu ra của thuật toán

D

độ phức tạp tính toán của thuật toán

**câu 3:** Nếu T1(n) và T2(n) là thời gian thực hiện 2 chương trình P1, P2 và T1(n)=O(f(n)), T2(n)=O(g(n)). Thời gian thực hiện của 2 chương trình

nối tiếp nhau là:

A

T(n)=max(f(n),g(n))

B

T(n)=O(f(n)\*g(n))

C

T(n)=O(max(f(n),g(n))

D

T(n)=O(f(n))\*O(g(n))

**Câu 4** Nếu độ phức tạp của lenh là O(1) thì độ phức tạp của đoạn chương trình sau:

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

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

{lệnh

được xác định bằng

A

O(nlogn)

B

O(1)

C

O(n)

D

0(n^2)

**Câu 5:** Nếu độ phức tạp của lệnh 1 và lệnh 2 đều là O(1) thì độ phức tạp của đoạn chương trình sau:

for (i=1 ; i <= n ; i++) {lệnh 1}

for (j=1 ;j <= m ; j++) {lệnh 2}

được xác định bằng:

A

0(m)

B

0(n\*m)

C

O(max(n,m))

D

0(n)

**Câu 6 :** Nếu độ phức tạp của lệnh là O(1) thì độ phức tạp của đoạn chương trình sau :

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

for (j=1 ; j <= m; j++) {lệnh}

for (k=1 ; k <= h ; k++) {lệnh}

được xác định bằng:

A

O(n+m\*h)

B

O(n\*max(m,h))

C

O(n\*(m+h))

D

O(n\*m\*h)

**Câu 7**: Nếu T1(n) và T2(n) là thời gian thực hiện 2 đoạn chương trình P1, P2 và T1(n)=O(f(n)), T2(n)=O(g(n)). Thời gian thực hiện 2 đoạn

chương trình lồng nhau:

A

T(n)=f(n)\*g(n)

B

T(n)=O(max(f(n),g(n))

C

T(n)=O(f(n)\*g(n))

D

T(n)=max(f(n),g(n))

**Câu 8:** Khi đánh giá độ phức tạp của thuật toán, hàm nào dưới đây có tốc độ tăng trưởng chậm nhất?

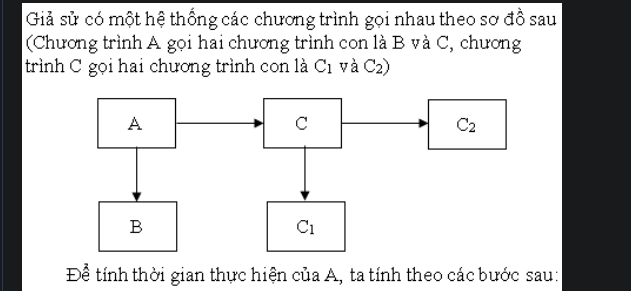
A O(nlogn)

B O(n)

C 0(n2)

D O(logn)

**Câu 9 :**



A

1. Tính thời gian thực hiện của B, C1

2. Tính thời gian thực hiện của C2

3. Tính thời gian thực hiện của A

B

1. Tính thời gian thực hiện của C1 và C2

2. Tính thời gian thực hiện của B

3. Tính thời gian thực hiện của A

C

1. Tính thời gian thực hiện của B

2. Tính thời gian thực hiện của C1, C2

3. Tính thời gian thực hiện của A

D

1. Tính thời gian thực hiện của B, C1 và C2

2. Tính thời gian thực hiện của C

3. Tính thời gian thực hiện của A

**Câu 10:** Xác định độ phức tạp cho đoạn chương trình sau:

S = 1; p = 1;

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

p=p\*x/ i;

S = s+ p;

}

A

O(logn)

B

0(n)

C

O(nlogn)

D

0(n2)

**Câu 11:** Xác định độ phức tạp cho đoạn chương trình sau:

if (m<n) p = m; else p = n;

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

c[i]=a[i] + b[i];

if (p<m)

for (i=p+1; i <= m; i++) c[i] = a[il;

else

for (i=p+1; i <= n; i++) c[i] = b[i];

while (p>0 && c[p] = 0) p = p-1;

A

0(m\*n)

B

O(m+n)

C

O(max(m,n))

D

O(logmn)

**Câu 12:** Xác định độ phức tạp cho đoạn chương trình sau:

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

for (int j = 0; j < n / 2; j++) {

for (int k = 0; k < 10; k++) {

printf("%d %d %d\n", i, j, k);

}

}

}

A

O(n2\*10)

B

0(n2)

C

O(n logn)

D

O(n)

**Câu 13** Xác định độ phức tạp cho đoạn chương trình sau:

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

for (int j = 0;j < n; j++) {

for (int k = 0; k < logn; k++) {

printf("%d %d %d\n", i, j, k);}} }

A

O(n log n)

B

O(n2 log n)

C

O(logn)

D

0(n3)

**Câu 14:** Chi phí thời gian của một quá trình tính toán là

A

thoi gian cần thiết đe kiểm tra qua trình tính toán

B

thời gian cần thiết để thực hiện một qua trình tính toán

C

thời gian cần thiết để thiết kế thuật toán

D

thời gian cần thiết để xây dựng thuật toán

**câu 15** : Với máy xử lý thuật toán bằng ngôn ngữ tựa ALGOL, giá về thời gian là:

A

Số phép tính số học

B

Số phép tính quan hệ

C

Số phép tính logic

D

Số phép tính căn bản

**Câu 16:** Nếu độ phức tạp của lệnh là O(1) thì độ phức tạp của đoạn chương trình sau :

for (i=1 ; i <= n ; i++) {lệnh}

được xác định bằng:

A

O(1)

B

O(logn)

C

0(n)

D

O(nlogn)

**Câu 17**: Xác định độ phức tạp cho đoạn chương trình sau:

int s=0;

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

for (int j=1; j≤i; ++j)

S=S+1;

printf("%d \n", s);

A

0(n2)

B

0(logn)

C

O(n)

D

O(nlogn)

**Câu 18:**

Xác định độ phức tạp cho đoạn chương trình sau:

int i = 0;

while (i < n) {

int j = i;

while (j < n) {

printf("%d %d\n", i, j);

j+=2;}

i++;}

A

0(n)

B

O(n log n)

C

O(n3)

D

O(n2)

**Câu 19:** Xác định độ phức tạp cho đoạn chương trình sau:

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

for (int j = 0; j < i; j++) {

for (int k = j; k < n; k++) {

printf("%d %d %d\n", i, j, k);} } }

A

0(n)

B

0(n2)

C

O(n3)

D

O(n logn)

**Câu 20 :** Xác định độ phức tạp cho đoạn chương trình sau:

void BS(int all, int n) {

for(int i=0; i<n-1; i++) {

for(int j=n-1; j>i; j-) {

if (aljil<a[j-1]) {

int tg = aljl;a[i] = a[j-1];a[j-1] = tg;

for(int i=0; i<n; i++){ printf("%d \t",a[i]);}

A

O(n logn)

B

0(n2 logn)

C

0(n)

D

O(n2)