Họ và tên: Phan Công Trung

Lớp: k43B

Mã sinh viên: 19T1021292

**Câu 1:**

**Thuật toán 1:**

Ta thấy phép so sánh n==0 là PTTC của tt.

Gọi g(n) là số lần thực hiện PTTC của CTC kích thước n. g(n/2) là … n/2.

Khi n=0, g(n)=g(0)=1. Khi n>0 thì pt gọi lại CTC HamMu(x,n/2) 2 lần

=> g(n)=

Ta có g(n)=2g(n)2+1=…..=2kgn2^k+2^k-1 (1)

Kích thước chương trình giảm xuống thì k sẽ tang lên nên chương trình sẽ dừng khi

n/2 k =1 <=>n=2^k k=log2 n <=>(1); g(n)=n.g(1)+n-1=n(2g(0)+1)+n-1=4n-1

Vậy độ phức tạp của thuật toán này làT(n)=O(4n-1)= O(n)

**Thuật Toán 2:**

Ta thấy phép so sánh n==0 là PTTC của tt.

Gọi g(n) là số lần thực hiện PTTC của CTC kích thước n. g(n/2) là … n/2.

Khi n=0, g(n)=g(0)=1. Khi n>0 thì pt gọi lại CTC HamMu(x,n/2) 1 lần

=> g(n)=

Ta có g(n)=gn2+1=…=gn2k+k (1)

Kích thước chương trình giảm xuống thì i sẽ tang lên nên chương trình sẽ dừng khi

n/2 k =1 => n=2 k => k=log 2 n =&gt;(1) => g(n)=g(1)+ log 2 n =g(0)+1+ log 2 n = log 2 n+2

Vậy độ phức tạp của thuật toán này là T(n)=O(log2n+2)= O(log2n)

**Câu 3:**

struct Nut{

int info;

Nut \*right;

Nut \*left;

};

a)

int duyet(Nut \*t, int &min){

if(p->info < min)min=p->info

duyet(p->left);

duyet(p->right);

return min;

}

void Min(Nut \*T){

Nut \*p = T;

min = p->info;

min = duyet(p, min);

cout<<min;

}

b)

int Cap(Nut \*T){

if (T == NULL || (T->left == NULL && T->right == NULL))

return 0;

if(1 + max(Cap(T->left)), Cap(T->right) > 2)return 2

return 1;

}

**Câu 4:**

1. void sapxep(int a[], int n){

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

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

if (a[j] > a[j+1]){

int temp = a[j];

a[j] = a[j+1];

a[j+1] = temp;

}

}

}

}

b)

int main(){

int a[] = {9,15,2,37,6,25,7,20,8,30};

bubbleSort(a, 10);

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

cout<<a[i]<<" ";

}

return 0;

}

**Câu 2:**

struct Nut{

int so;

Nut \*next;

};

a)

int kt(Nut \*L){

Nut \*p = L;

Nut \*q = p->next;

while(q!=NULL){

if(q->so < p->so)return 0;

p = p->next;

q = q->next;

}

Return 1;

}

b)

void\* them\_Nut(Nut \*L3, int x){

Nut\*L3

Nut \*p = new Nut;

p->so = x;

p->next = NULL;

if(L3 == NULL){

L = p;

}

else{

Nut \*q = L;

while(q!=NULL){

q = q->next;

}

q->next = p;

}

}

Nut\* kt(Nut \*L1, Nut \*L2){

for(Nut \*p = L1; p!=NULL; p=p->next){

for(Nut \*q = L2; q!=NULL; q=q->next){

if(q->so == p->so){

them\_Nut(L3, p->so);

}

}

}

Return L3;

}