GIẢI ĐỀ :

Đề 1:

**Câu 1** : *(25 điểm)* Hãy cài đặt và đánh giá độ phức tạp của thuật toán sắp xếp nổi bọt trên mảng gồm n số nguyên. Minh họa các bước thực hiện thuật toán sắp xếp nổi bọt trên mảng a[ ] = {2, 17, 3, 1, 9, 3, 2}.

void bubble\_sort(int a[],int n)

{

int i,j,tmp;

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

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

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

{

tmp=a[j+1];

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

a[j]=tmp;

}

}

* Độ phức tạp : 0(n bình)
* Minh họa:

1, Ý tưởng: Xuất phát từ đầu hoặc cuối dãy so sánh 2 phần tử liên tiếp vs nhau nếu phần tử đứng đằng trước mà lớn hơn thì đổi chỗ.. sau mỗi lượt phần tử lớn nhất sẽ đc chuyển về cuối dãy(or phần tử nhỏ nhất sẽ đc chuyển về đầu dãy) phần tử này sẽ ko đc xét đến ở bước tiếp theo( quá trình trên lặp n-1 lần).

3, Mô tả:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Key bước | a[0] | a[1] | a[2] | a[3] | a[4] | a[5] | a[6] |
| Ban đầu | 2 | 17 | 3 | 1 | 9 | 3 | 2 |
| B1 | 2 | 17 |  |  |  |  |  |
| B2 | 2 | 3 | 17 |  |  |  |  |
| B3 | 1 | 2 | 3 | 17 |  |  |  |
| B4 | 1 | 2 | 3 | 9 | 17 |  |  |
| B5 | 1 | 2 | 3 | 3 | 9 | 17 |  |
| B6 | 1 | 2 | 2 | 2 | 3 | 9 | 17 |

Câu 2: a) Khai báo danh sách liên kết đơn chứa các trường thông tin về nhân sự như sau: Tên nhân viên, mã nhân viên, hệ số lương, năm sinh.

b) Cài đặt hàm sắp xếp danh sách giảm dần theo năm sinh, nếu cùng năm sinh thì theo mã nhân viên.

A . Khai báo

typedef struct{

char TenNV[30];

int mnv;

float hsl;

int ns;

}NV;

typedef struct tagNode{

NV infor;

struct tagNode \*link;

}Node;

typedef struct{

Node \*pHead;

Node \*pTail;

int spt;

}Llist;

B Sắp xếp giảm dần theo năm sinh , cùng năm theo mnv

void sx(list \*L)

{

Node \*i,\*j;

sv temp;

for(i=L->pHead; i->link!=NULL; i=i->link)

for(j=i->link;j->link!=NULL;j=j->link)

{

If((i->infor.ns,j->infor.ns)>0 || (i->infor.ns,j->infor.ns)==0 && i-> infor.msv < j->infor.mnv ))

{

temp=i->infor;

i->infor=j->infor;

j->infor=temp;

}

}

}

Câu 3 :

Int S=0;n,I;

Printf (“ Nhap n : “);

Scanf (“%d”,&n);

While (n !=0)

{

S+= n%10;

n/=10;

}

For ( i=2;i<=(S/2);i++)

{

If (S%i == 0)

printf (“\n %d Không là số nguyên tố “,S);

else printf (“\n %d là số nguyên tố”,S);

}

Câu 4 a) Khai báo cấu trúc một nút của cây tìm kiếm nhị phân. Thông tin chứa trong nút gồm: tên, số chứng minh thư, địa chỉ.

b) Trình bày cách thức chèn một nút vào cây tìm kiêm nhị phân. Hãy chèn lần lượt các khóa sau vào một cây tìm kiếm nhị phân rỗng mà các nút chứa các số nguyên: 19, 6, 2, 8, 26, 24, 72, 35, 26, 12. Viết hàm duyệt cây theo thứ tự trước và đưa ra kết quả thực hiện hàm đó với cây vừa dựng.

A khai báo:

1. typedef struct{

char ten[30];

int SoCMT;

char diachi[30];

}Data;

typedef struct tagNode{

Data infor;

struct tagNode \*left, \*right;

}Node;

typedef Node \*BST;

)

**cách chèn**: Thêm 1 nút X vào cây có nút gốc T.

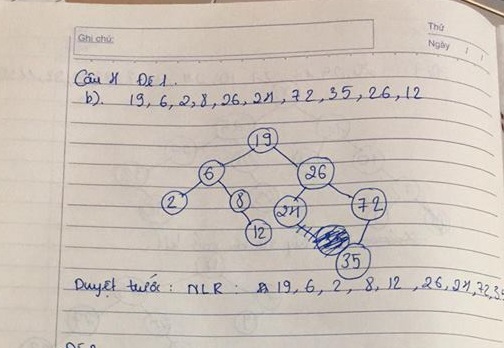
b1) nếu T=NULL thì thêm X vào T.

b2) nếu T khác NULL thì:

th1:nếu X=khóa tại T( có thể bỏ qua or thêm X vào bên phải T\_tùy từng bài toán)

th2: nếu X >khóa tại T thì thêm X vào bên phải T.

1. nếu X <khóa tại T thì thêm X vào bên phải T



**Đề 2**

Câu 1 Đổi chỗ trực tiếp

a) void exchange\_sort(int a[],int n)

{

int i,j,temp;

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

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

{

if(a[i]>a[j])

{

temp=a[i];

a[i]=a[j];

a[j]=temp;

}

}

}

Độ phức tạp : O(n bình)

1. 16, 4, 8, 19, 30, 26, 7, 1

Minh họa :

Ban đầu : 16,4,8,19,30,26,7,1

B1 : 1,16,8,19,30,26,7,4

B2 : 1,4,8,19,30,26,7,16

B3 : 1,4,7,19,30,26,8,16

B4 : 1,4,7,8,30,26,19,16

B5 : 1,4,7,8,16,26,19,30

B6: 1,4,7,8,16,19,26,30

Câu 2 *a)* Khai báo danh sách liên kết đơn chứa các trường thông tin về sinh viên như sau: Mã sinh viên, Họ và tên, Điểm trung bình

1. Hãy cài đặt hàm loại bỏ phần tử có mã sinh viên x trong danh sách (x là tham số được truyền vào).

b.

void delete(DSLK \*L)

{ int msv;

Node \*p, \*q;

p=L->Head;

if(L->Head==NULL)return ;

else{

printf(“Nhap msv cua sv can delete”);

scanf(“%d”,&msv)

if(L->Head==L->Tail&&msv==p->infor.msv)

{

L->Head=NULL;

L->Tail=NULL;

}

else{

if(L->Head==L->Tail&&msv!=p->infor.msv) return;

else{

q=p;

p=p->link;

if(msv==p->infor.msv)

{q->link=p->link;

free(p);

L->spt--;

}

}

}

}

A Khai báo:

typedef struct sinhvien{

char ten[30];

int masv;

float dtb;

}data;

typedef struct tagnode{

data infor;

struct tagnode \*link;

}node;

typedef struct danhsach{

node \*phead;

node \*ptail;

int spt;

}llist;

B

Câu 3 :

For (i=0; i<n-1;i++)

{

If (a[i] % 2 !=0)

{

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

{

if ( a[i]>a[j] && a[j] %2 !=0)

{

Temp = a[i];

A[i] = a[j];

A[j] = temp;

}

}

}  
 }

Câu 4 *) a)* Trình bày cách thức chèn một nút vào cây tìm kiêm nhị phân. Hãy chèn lần lượt các khóa sau vào một cây tìm kiếm nhị phân rỗng mà các nút chứa các số nguyên: 19, 7, 1, 9, 25, 21, 70, 7, 30, 8. Xóa khỏi cây nút gốc. Đưa ra kết quả duyệt cây theo thứ tự trước và thứ tự sau với cây sau khi xóa nút gốc.

1. Viết hàm thêm một nút vào cây tìm kiếm nhị phân.

B Viết hàm thêm 1 nút vào cây:

void InsertNode(BSTRee \*T , data x)

{

if((\*T) == NULL)

{

(\*T) = (Node2\*)malloc(sizeof(Node2));

(\*T)->infor = x;

(\*T)->left = NULL;

(\*T)->right = NULL;

}

else

{

if(x.ma > (\*T)->infor.ma )

InsertNode(&(\*T)->right,x);

else if(x.ma < (\*T)->infor.ma)

InsertNode(&(\*T)->left,x);

else return;

}

}

A )

**cách chèn**: Thêm 1 nút X vào cây có nút gốc T.

b1) nếu T=NULL thì thêm X vào T.

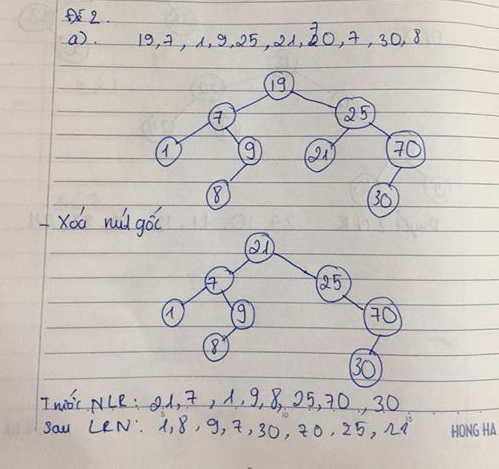
b2) nếu T khác NULL thì:

th1:nếu X=khóa tại T( có thể bỏ qua or thêm X vào bên phải T\_tùy từng bài toán)

th2:nếu X >khóa tại T thì thêm X vào bên phải T.

nếu X <khóa tại T thì thêm X vào bên phải T.

**Vẽ :**

****

**Đề 3**

Câu 1

void QuickSort\_mid(Sinhvien a[],int left, int right) // **SX Nhanh**

{

int i=left;

int j=right;

int tam;

int x= (left + right)/2;

while (i<=j)

{

while (a[i]<x)i++;

while (a[j]>x)j--;

if (i<=j)

{

int tam= a[i];

a[i] = a[j];

a[j] = tam;

}

}

}

* Độ phức tạp 0(N^2)

Câu 2 a) Khai báo danh sách liên kết đơn chứa các trường thông tin về sinh viên như sau: Tên sinh viên, mã sinh viên, giới tính.

b) Giả sử có một danh sách liên kết đơn L chứa các sinh viên. Hãy viết hàm tách danh sách L thành 2 danh sách L1 (chứa các sinh viên nam) và L2 (chứa các sinh viên nữ).

1. Như trên
2. void tackds(llist l ,llist \*L1 , llist \*L2)

{

Node \*p;

P=L.pHead;

initList(L1);

initList(L2);

while(p!=NULL) {

if(p->infor.gioitinh == 0)

insert\_Head(L1 , p->infor);

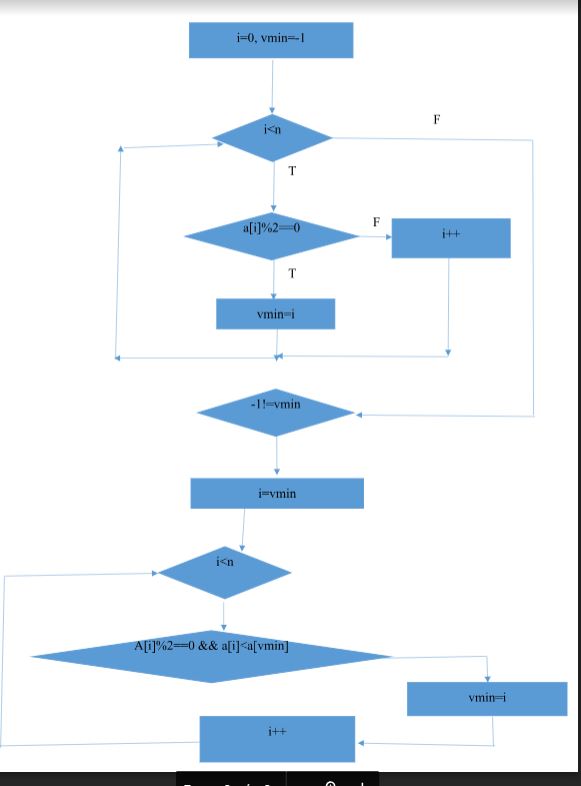
Else insert\_head(L2 , p->infor);

p=p->link;

}

}

Câu 3



Câu 4 a)Thông tin của nhân viên trong một công ty gồm: mã nhân viên, họ tên, lương. Hãy khai báo cấu trúc dữ liệu cây tìm kiếm nhị phân để lưu thông tin trên. Khóa là mã nhân viên.

b) Trình bày cách thức chèn một nút vào cây tìm kiêm nhị phân. Vẽ cây tìm kiếm nhị phân tương ứng khi thêm lần lượt các nhân viên sau: {(12, Hùng, 100), (22, Huy, 200), (11, Hậu, 150), (7, Vân, 100), (14, Trang, 400), (24, Sỹ, 300), (2, Linh, 100), (10, Lâm, 700), (22, Huy, 200)}.

A

typedef struct{

char ten[30];

int maNV;

float luong;

}Data;

typedef struct tagNode{

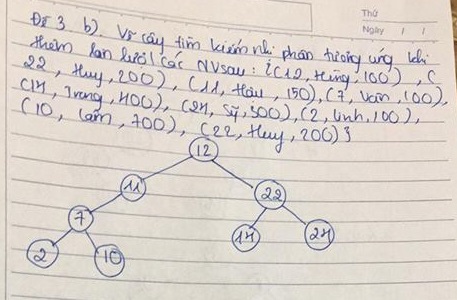
Data infor;

struct tagNode \*left, \*right;

}Node;

typedef Node \*BST;

b. – Cách chèn : như trên



**ĐỀ 4:**

Câu 1

**, Sắp xếp chèn:**

void insertion\_sort(int a[] , int n)

{

int i,j,vt;

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

{

X= a[i];

Vt = I;

while( vt > 0 && a[vt-1] > x)

{

A[vt] = a[vt-1];

Vt--;

}

a [vt] = x;

}

}

* Độ phức tạp:

+ TH xấu nhất ( so sánh a[i] với nhiều nhất là I phần tử -> có I phép so sánh và dịch chuyển )

+ Tg thực hiện thuật toán :

1+2+3+…+(n-1) = n(n-1)/2 = 0(n bình )

* Minh họa :

Ban đầu : 19, 7, 16, 2, 6, 4, 20, 18, 12.

B1: 2,7,16,19,6,4,20,18,12

B2: 2,4,16,19,6,7,20,18,12

B3 : 2,4,6,19,16,7,20,18,12

B4 ; 2,4,6,7,16,19,20,18,12

B5 : 2,4,6,7,12,19,20,18,16

B6 : 2,4,6,7,12,16,20,18,19

B7 : 2,4,6,7,12,16,18,20,19

B8: 2,4,6,7,12,16,18,19,20

Câu 2 a) Khai báo danh sách liên kết đơn chứa các trường thông tin về hàng hóa như sau: Tên mặt hàng; Giá mặt hàng; Số lượng còn trong kho

b) Hãy cài đặt hàm sắp xếp danh sách ở câu a theo chiều tăng dần của giá mặt hàng, nếu cùng giá thì theo tên.

1. Khai báo

typedef struct HH{

char tenMH[30];

float giaMH;

int sl;

}HH;

typedef struct tagnode{

HH infor;

struct tagnode \*link;

}node;

typedef struct danhsach{

node \*phead;

node \*ptail;

int spt;

}llist;

B . Sắp xếp:

void sx(list \*L)

{

Node \*i,\*j;

sv temp;

for(i=L->pHead; i->link!=NULL; i=i->link)

for(j=i->link;j->link!=NULL;j=j->link)

{

If((i->infor.giaMH,j->infor.giaMH)>0 || (i->infor.giaMH,j->infor.giaMH)==0 && (strcmp( i-> infor.tenMH , j->infor.tenMH )>0))

{

temp=i->infor;

i->infor=j->infor;

j->infor=temp;

}

}

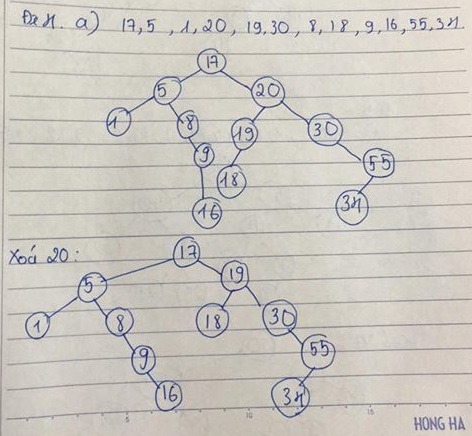
}

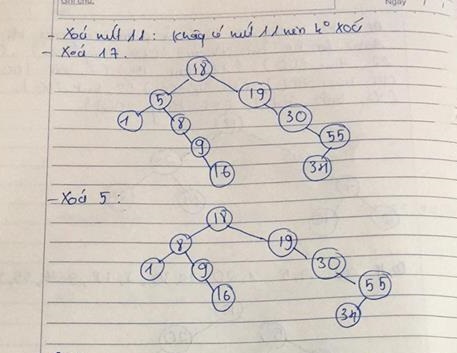
**Câu 4** : *(25 điểm)*

a) Hãy chèn lần lượt các khóa sau vào một cây tìm kiếm nhị phân rỗng mà các nút chứa các số nguyên: 17, 5, 1, 20, 19, 30, 8, 18, 9, 16, 55, 34. Trình bày cách thức xóa một nút khỏi cây tìm kiếm nhị phân. Xóa khỏi cây lần lượt các nút chứa các khóa: 20, 11, 17, 5. Vẽ lại các cây sau khi xóa mỗi nút.

b) Chuyển biểu thức trung tố sau thành biểu thức tiền tố, hậu tố: 12\*5 + 2\*(17-3\*4) + 8

BL a. Vẽ cây:





* **Cách xóa nút :**

th1) nút cần xóa ko có con(xóa là xong)

th2) nút cần xóa có 1 con.. xóa nút cần xóa và thế con của nó vào vị trí tương ứng.

th3) nút cần xóa có 2 con..

* cách 1) đổi chỗ nút cần xóa cho nút lớn nhất bên trái
* cách 2) đổi chỗ nút cần xóa cho nút nhỏ nhất bên phải.

Sau khi đổi chỗ thì xóa nút cần xóa ở vị trí mới

**Đề 5**

Câu 1:

**B,Tìm kiếm nhị phân: (Chú ý: mảng a[ ] đã cho đã đc sắp xếp)**

1, Phương pháp:

* Đặt **L**=0;**R**=n-1.. tính **m**(vị trí giữa dãy) **m**=(**L+R**)/2
* So sánh khóa **x** với **a**[**m**].
* Trường hợp 1: x == a[m] return m;
* Trường hợp 2: x > a[m] thì tìm bên phải m(từ m+1 đến cuối)
* Trường hợp 3: x < a[m] thì tìm bên trái(từ đầu đến m-1)

Lặp lại các bước trên khi tìm bên phải hoặc bên trái và return -1 khi dãy rỗng.

2, Độ phức tạp: Vì sau mỗi bước tìm kiếm miền dữ liệu lại giãm đi phân nữa, tuần tự tiến dần tới 0 nên số lượng các bước tìm kiếm cũng tăng dần tới tối đa là log2(N). Vì vậy độ phức tạp của thuật toán này được xem là O(log2(N)).

Câu 2 Khai báo cấu trúc dữ liệu danh sách liên kết kép chứa các thông tin về nhân sự gồm có các trường thông tin: Họ và tên; Thâm niên công tác (tính theo năm); Hệ số lương; Lương cơ bản.

1. Viết hàm thêm một bản ghi chứa thông tin nhân sự của một người vào cuối danh sách đã khai báo ở phần a.
2. Khai báo:

typedef struct {

char hovaten[30];

int thamnien;

float hesoluong;

float luongcoban;

}Data;

typedef struct DNode{

struct DNode \*Pre; // trỏ đến phần tử đứng trước

struct DNode \*Next; // trỏ đến phần tử đứng sau

}DNode;

typedef struct {

Node \*head;

Node \*tail;

int spt;

}DList;

B. thêm 1 bản ghi

void nhap(congnhan \*x)

{ printf(“nhap ho ten:”);

fflush(stdin);

gets(x->ten);

printf(“nhap tham nien:”);

scanf(“%d”,&(x->nam);

printf(“nhap he so luong :”);

scanf(“%f”,&(x->hsl);

printf(“nhap luong co ban :”);

scanf(“%ld”,&(x->lcb);

}

void Insert\_Head(DSLK \*L,congnhan x)

{Node \*p;

p=(Node \*)malloc\*(sizeof(Node));

p->infor=x;

p->link=NULL;

if(L->Head==NULL){

L->Head=p;

L->Tail=p;

}

else{

p->link=L->Head;

L->Head=p;

}

L->spt++;

}

*voi Insert\_Tail(DSLK \*L,congnhan x)*

*{ Node \*p;*

*p=(Node \*)malloc(sizeof(Node));*

*p->infor=x;*

*p->link=NULL;  
 if(L->Head==NULL){*

*L->Head=p;*

*L->Tail=p;*

*}*

*else{*

*L->Tail->link=p;*

*L->Tail=p;*

*}*

*L->spt++;*

*Câu 3*

**int d,i ,n;**

**d=0;**

**i=n-1;**

**if(i≥=0) {**

**for(i=n-1;i>=0;i - -)**

**{**

**if(a[i]%2==0)**

**{**

**printf(" a[%d]=%d",i,a[i]);**

**d=1;**

**}**

**}**

**else**

**if(d==0){**

**printf("khong co so thoa man");**

**}**

**Câu 4** : *(25 điểm)*

1. Chèn lần lượt các khóa sau vào một cây tìm kiếm nhị phân rỗng: 20, 5, 1, 17, 30, 24, 7, 18, 23, 9, 32, 25. Trình bày cách thức xóa một nút khỏi cây tìm kiếm nhị phân. Xóa khỏi cây lần lượt các nút chứa các khóa sau: 24, 10, 7. Vẽ lại các cây sau khi xóa mỗi nút.

Chuyển biểu thức trung tố sau thành dạng hậu tố và tiền tố: x\*y + z\*(t-u). Trong đó z, y, z, t, u là các biến (toán hạng).

* 1. Cach thức xóa : như trên

Vẽ Cây:

**ĐỀ 6 :**

**Câu 1** Cài đặt thuật toán sắp xếp nổi bọt tăng dần trên mảng cấu trúc sinh viên gồm các trường thông tin sau: Mã sinh viên; Tên sinh viên; Tuổi sinh viên. Trường khóa để sắp xếp là trường tuổi, nếu cùng tuổi thì theo tên.

void sapxep(sinhvien a[],int n)

{

int i,j;

sinhvien tam;

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

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

{

if(a[j-1].tuoi>a[j].tuoi)

{

tam=a[j];

a[j]=a[j-1];

a[j-1]=tam;

}

if(a[j-1].tuoi==a[j].tuoi)

{

if(strcmp(a[j-1].hoten,a[j].hoten)>0)

{

tam=a[j];

a[j]=a[j-1];

a[j-1]=tam;

}

}

}

}

Câu 2 a Khai báo cấu trúc dữ liệu cây tìm kiếm nhị phân chứa các thông tin về nhân sự gồm có các trường thông tin: Họ và tên; Mã nhân sự; Hệ số lương; Lương cơ bản.

B .Viết hàm thêm một bản ghi chứa thông tin nhân sự của một người vào cây đã khai báo ở phần a.

1. typedef struct{

char ten[30];

int mns;

float hsl;

double lcb;

}NV;

typedef struct tagNode{

NV infor;

struct tagNode \*left, \*right;

}Node;

typedef Node \*BST;

void Nhap(Nhan su \*x)

{ printf(“nhap ho ten:”);

fflush(stdin);

gets(x->ten);

printf(“nhap ma nhan su :”);

scanf(“%d”,&(x->mns);

printf(“nhap he so luong :”);

scanf(“%f”,&(x->hsl);

printf(“nhap luong co ban :”);

scanf(“%ld”,&(x->lcb);

}

1. void insert\_Node(NV x,BST \*T){

if(( \*T)==NULL){

(\*T) = (Node\*) malloc (sizeof(Node));

(\*T)->infor=x; ;

(\*T)->left=NULL;

(\*T)->right=NULL;}

else{

if((\*T)->infor.mns<x.mns)) insert\_Node(x, &(\*T)->right);

else if(\*T)->infor.mns>x.mns) insert\_Node(x, &(\*T)->left):

else printf(“khoa bi trung”);

}

}

Câu 3

int k=3, b=100;

//print(“Nhap n : ”);

//scanf(“%d”, &amp;n);

for(int i=n-1; i&gt;=k+1 ; i--){

a[i] = a[i-1];

}

a[k] = b;

for( I =0; I &lt;n ; i++){

printf(“%d”, a[i]) ;

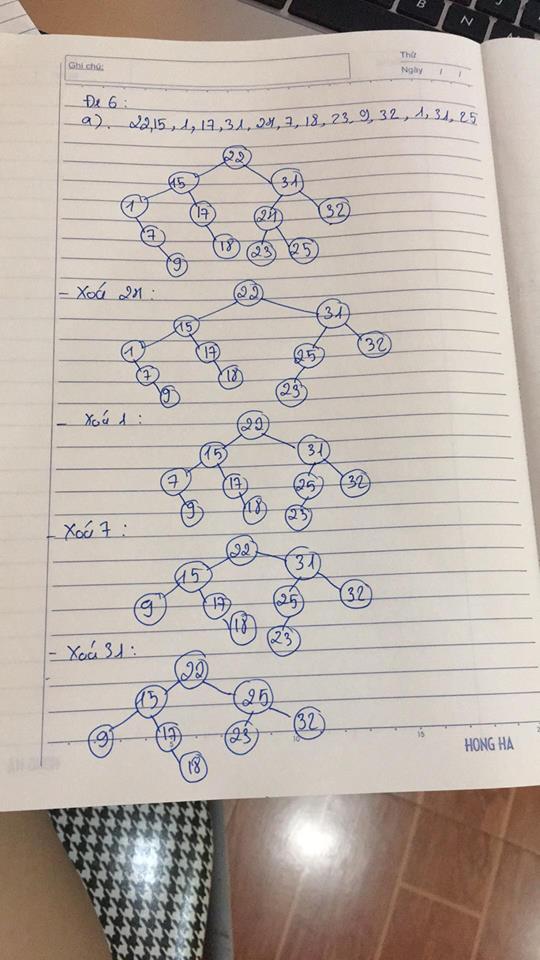
}

5,6,-1,100,3,-9

**Câu 4** : *(25 điểm)*

1. Trình bày cách thức chèn một nút vào cây tìm kiếm nhị phân. Chèn lần lượt các khóa sau vào một cây tìm kiếm nhị phân rỗng: 22, 15, 1, 17, 31, 24, 7, 18, 23, 9, 32, 1, 31, 25. Xóa khỏi cây các lần lượt nút chứa các khóa sau: 24, 1, 7, 31. Vẽ lại các cây sau khi xóa mỗi nút.

Chuyển biểu thức trung tố sau thành dạng hậu tố và tiền tố: x\*y + z\*(t-u). Trong đó z, y, z, t, u là các biến (toán hạng).



ĐỀ 7:

Câu 1

**B,Tìm kiếm nhị phân: (Chú ý: mảng a[ ] đã cho đã đc sắp xếp)**

**Giống c1 đề 5**

Câu 2

1. Khai báo như đề 6
2. Tổng lương:

Float tongluong( BST T)

{

If (T!=NULL)

{

Tongluong (T->left);

Tongluong(T-> right );

Return T->infor.lcb \*T->infor.hsl;

}

Return 0;

}

// VD Duyet luong >10000000

Void Dtruoc(BST T)

{

If ( T != NULL && T-> infor.luong > 10000000 )

{

Xuat (T ->infor);

Dtruoc(T->left);

Dtruoc(T->right);

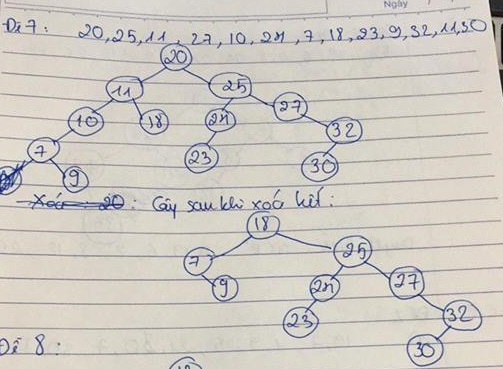
}

}

1. Câu 4 :

- Chèn lần lượt các khóa sau vào một cây tìm kiếm nhị phân rỗng: 20, 25, 11, 27, 10, 24, 7, 18, 23, 9, 32, 11, 30. Trình bày cách thức chèn một nút vào cây tìm kiếm nhị phân. Xóa khỏi cây lần lượt các nút chứa các khóa sau: 20, 10, 45, 11. Vẽ lại các cây sau khi xóa mỗi nút.

* Chuyển biểu thức hậu tố sau thành dạng trung tố, tiền tố và cho biết giá trị của biểu thức: 12, 30, 48, 2, /, 29, -, \*, +, 37, 17, -, 2, /, +

****

**ĐỀ 8**

Câu 1

**B, Sắp xếp chọn trực tiếp:**

void selection\_sort(sv a[],int n)

{

int i,j,tmp;

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

{

Int min\_index = i;

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

if( a[j] < a [min\_index])

min\_index = j;

tmp = a[min\_index];

a[min\_index] = a[i];

a[i]= tmp;

}

}

* Độ phức tạp:
* 1, Ý tưởng:
* Chọn phần tử nhỏ nhất & đổi chỗ cho phần tử đầu tiên.
* Sau mỗi bước, phần tử đầu tiên sẽ ko đc xét ở bước tiếp theo, quá trình lặp lại n-1 bước.
* 3, Mô tả.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Key bước | a[0] | a[1] | a[2] | a[3] | a[4] | a[5] | a[6] |
| Ban đầu | 2 | 17 | 3 | 1 | 9 | 3 | 2 |
| B1 | 1 | 17 | 3 | 2 | 9 | 3 | 2 |
| B2 |  | 2 | 3 | 17 | 9 | 3 | 2 |
| B3 |  |  | 2 | 17 | 9 | 3 | 3 |
| B4 |  |  |  | 3 | 9 | 3 | 17 |
| B5 |  |  |  |  | 3 | 9 | 17 |

Câu 2 a) Khai báo danh sách liên kết đơn chứa các trường thông tin về sinh viên như sau: Tên sinh viên, mã sinh viên, điểm trung bình.

b) Giả sử có hai danh sách liên kết đơn L1, L2 chứa các sinh viên. Hãy viết hàm gộp hai danh sách trên thành một danh sách sao cho danh sách kết quả được sắp xếp tăng dần theo điểm

Typedef struct {

Char ten[30];

Int msv;

Float dtb;

}SV;

Typedef struct tagNode {

SV -> infor ;

Struct tagNode \*link;

}Node;

Typedef struct {

Node \*phead;

Node \*pTail;

Int spt;

}Llist;

1. Gộp danh sách

void gop(dslk \*L1, dslk \*L2,sv x)

{

L1->tail->link=L2->head;

L1->tail=L2->tail;

L1->spt=L1->spt+L2->spt;

Node \*i, \*j;

for(i=L1->head;i!=L1->tail;i=i->link) // sap xep

for(j=i->link ; j!=NULL;j=j->link)

if(j->infor.dtb<=i->infor.dtb)

{

sv tg=i->infor;

i->infor=j->infor;

j->infor=tg;

}

}

Câu 3

I++

F

Sum = this sum

This sum > sum

This sum += a[i]

I < n

I=0

This Sum = 0

Sum = 0

This sum < sum

F

T

F

Câu 4

VD: Hãy viết hàm liệt kê các học sinh đỗ đại học (điều kiện: tổng điểm ba môn >=15 và không môn nào điểm =0).

void dodaihoc(Sinhvien a[],int n)

{

int i;

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

if((a[i].dt+a[i].dl+a[i].dh) >=15.0 && a[i].dt>0 && a[i].dl >0 && a[i].dh>0)

printf("%s ",a[i].ten);

}

Tiên tố : NLR

Hậu Tố : LRN

Trung Tố : LNR