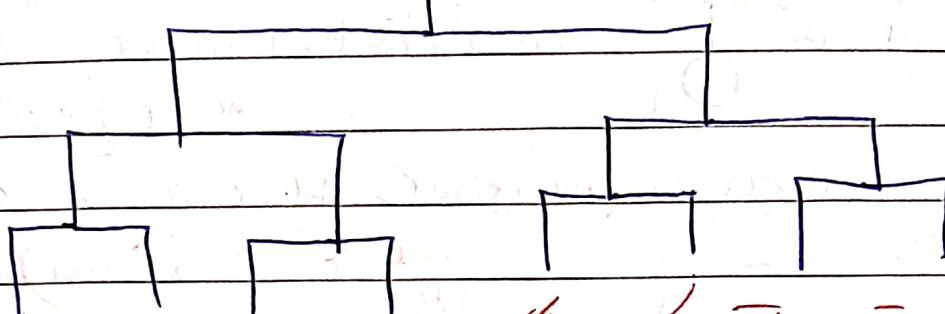


(data structure)

| 6-2 |

* ساختار (struct) یا صنایع (products) سازمانی (organized) از عناصر

ساختار باهم ربط دارند.



خط ۳ ۸۱۴۶۲۰ (صنایع) و میراث از نسبت واحد کردن اسن

خط ۴ ۸۱۴۶۲۰ (دانات) که هدایت خاصی به الیم رسید اسن

نمر (نخان)

خط ۵ ۸۱۴۶۲۰ (نهاد) و تابعی مجموعکاری از مجموعکاری که هدایت

خط ۶ ۸۱۴۶۲۰ (نهاد) مجموعکاری از مجموعکاری

خط ۷ ۸۱۴۶۲۰ (نهاد) که هدایت مجموعکاری از مجموعکاری

خط ۸ ۸۱۴۶۲۰ (نهاد) که هدایت مجموعکاری از مجموعکاری

خط ۹ ۸۱۴۶۲۰ (نهاد) که هدایت مجموعکاری از مجموعکاری

خط ۱۰ ۸۱۴۶۲۰ (نهاد) که هدایت مجموعکاری از مجموعکاری

خط ۱۱ ۸۱۴۶۲۰ (نهاد) که هدایت مجموعکاری از مجموعکاری

خط ۱۲ ۸۱۴۶۲۰ (نهاد) که هدایت مجموعکاری از مجموعکاری

خط ۱۳ ۸۱۴۶۲۰ (نهاد) که هدایت مجموعکاری از مجموعکاری

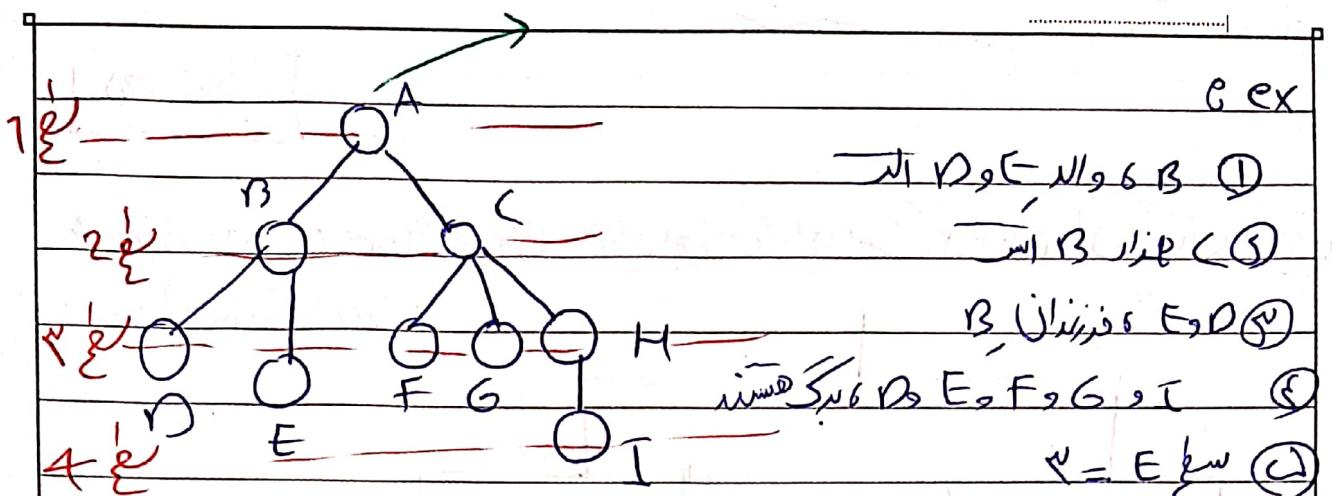
خط ۱۴ ۸۱۴۶۲۰ (نهاد) که هدایت مجموعکاری از مجموعکاری

خط ۱۵ ۸۱۴۶۲۰ (نهاد) که هدایت مجموعکاری از مجموعکاری

خط ۱۶ ۸۱۴۶۲۰ (نهاد) که هدایت مجموعکاری از مجموعکاری

خط ۱۷ ۸۱۴۶۲۰ (نهاد) که هدایت مجموعکاری از مجموعکاری

خط ۱۸ ۸۱۴۶۲۰ (نهاد) که هدایت مجموعکاری از مجموعکاری

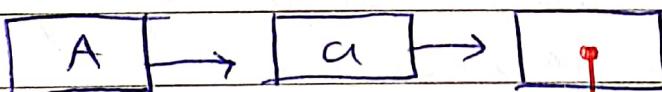


$A = (a, (b, c))$ أمثلة

زمرة

$A (a, z(b, c))$

العنصر



Level

1

نقطة

2

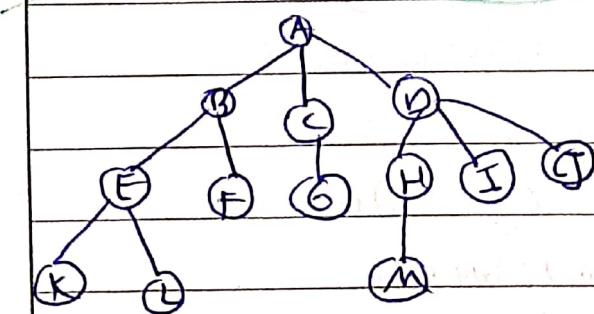
نقطة

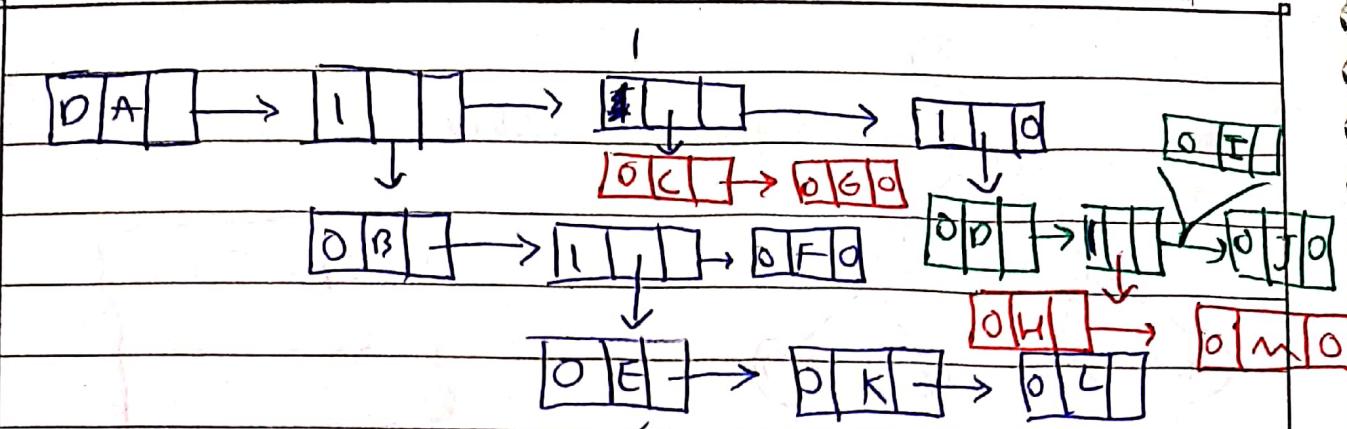
3

$(A(B(E(K, L), F), G))$

4

$(D(H(M), I, J)))$



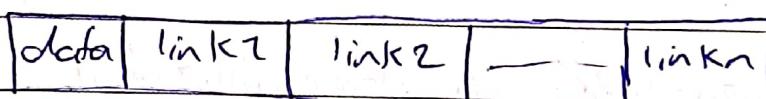


B جوی اول =

B جوی اول =

نایس خاص) ملک نه ملک کرده همای بیفول با
نایس خاص) ملک نه ملک کرده رفعه لفافه لفافه که سابل

نایس کرده همای بیفول



$\wedge = 7$ $\wedge = 7$ $\wedge = 7$ حافظا

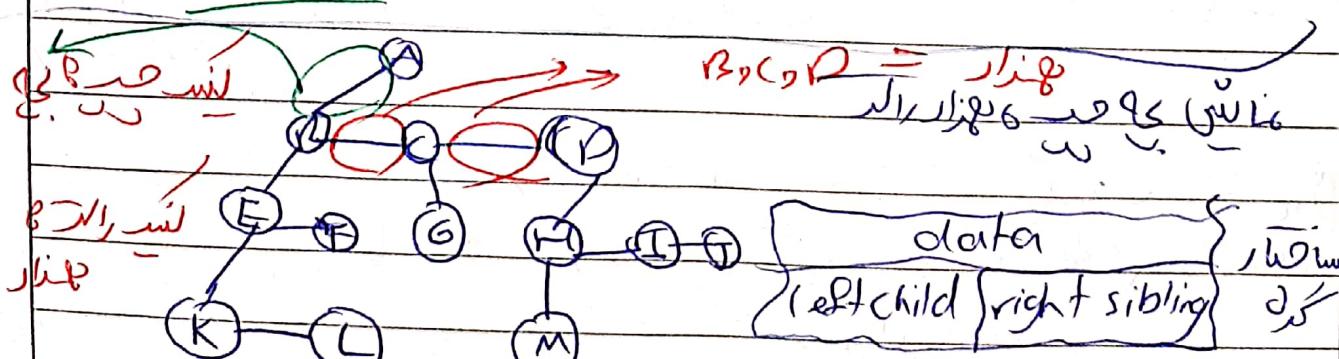
$n(k-1) + 1$ $n(k-1)$ $n(k-1)$ $n(k-1)$ $n(k-1)$ $n(k-1)$ $n(k-1)$

صفر صفر صفر صفر صفر صفر صفر

$= 2$ صفرها

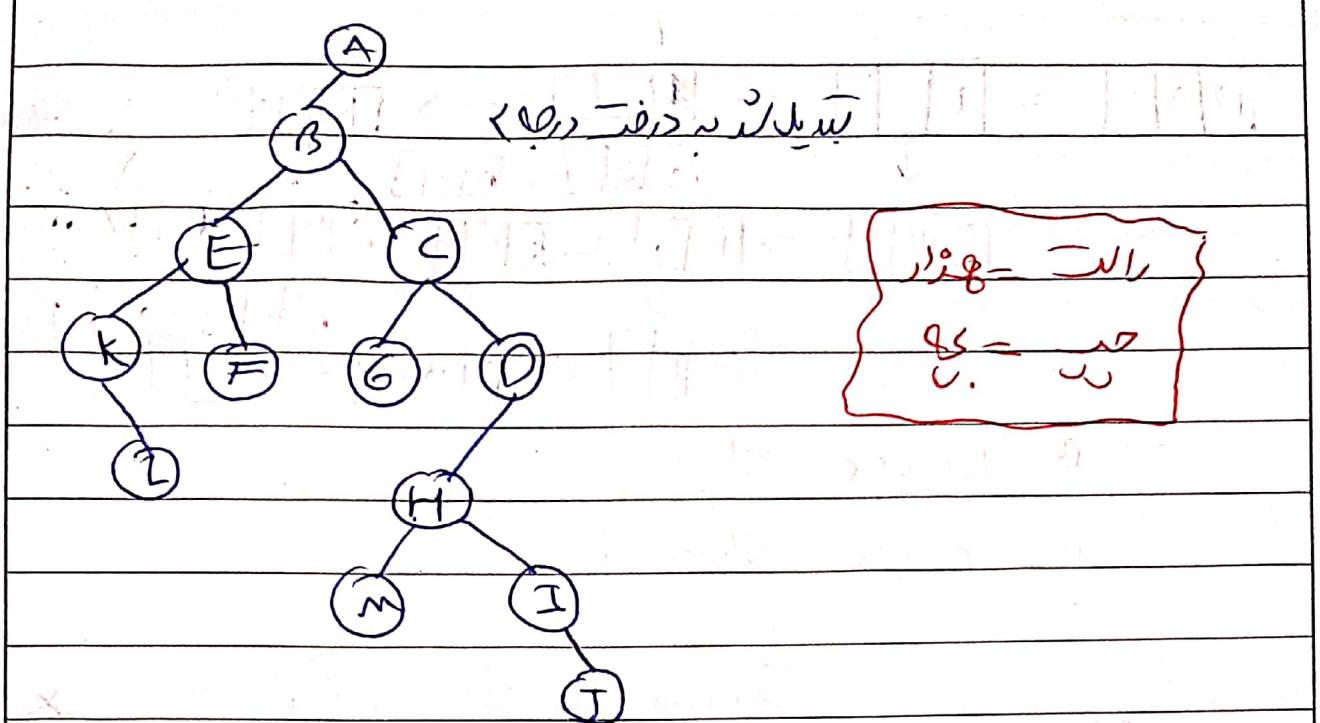
حافظا درخت درخت درخت درخت درخت درخت

درخت درخت درخت درخت درخت درخت



دستور

hirmandpaper



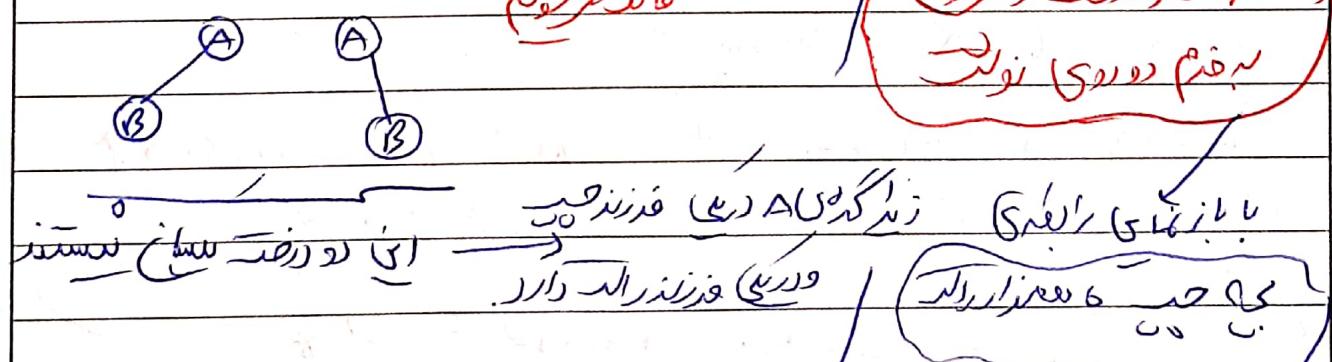
لطفاً کس خواهد داشت که درجه دار (از زیرنامه) مجموعه هایی از ترکیب که با خانی است باشد
رسانی و زیرنفث جی و زیر درخت را در همین

درخت درجی و سی زیرنفث جی و زیر درخت

قابل انتخاب

(درخت های درج)

فرزندت را صرف
نمی دادی نویل

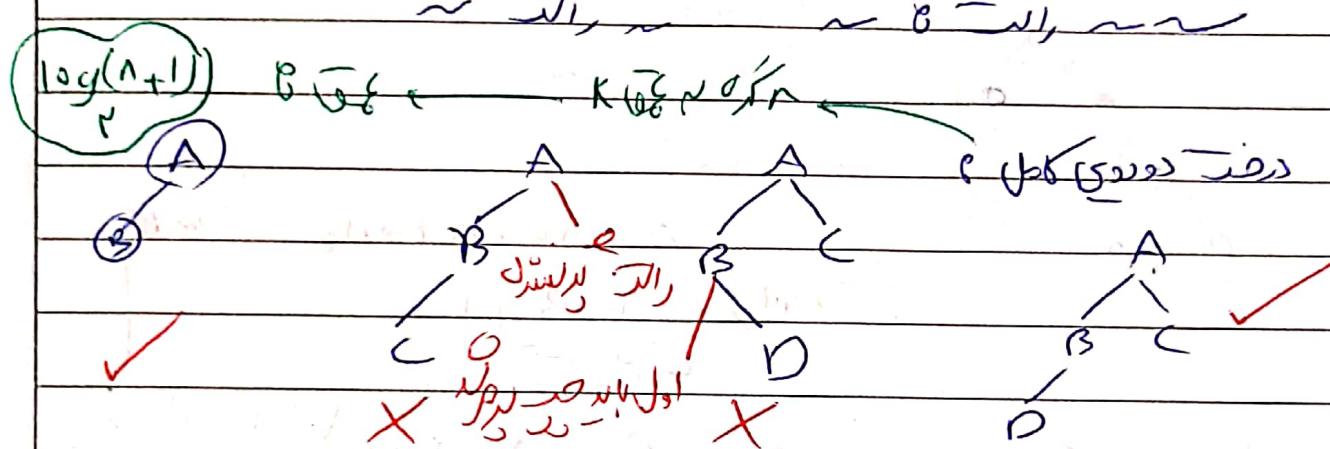
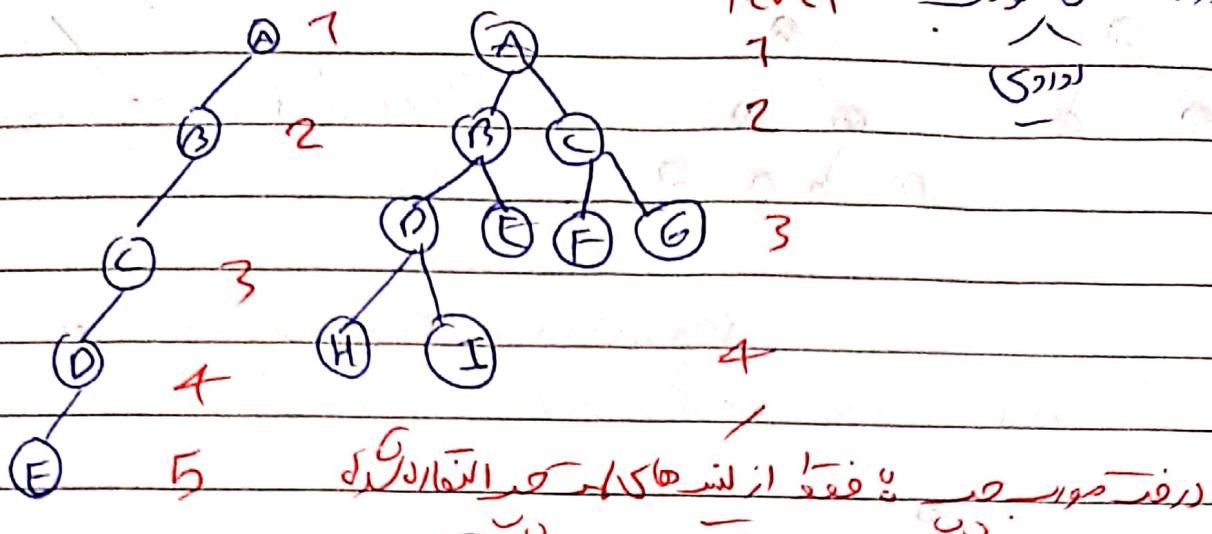


درخت دفعی و مرتکنی باشند و کی درخت چون کس کس

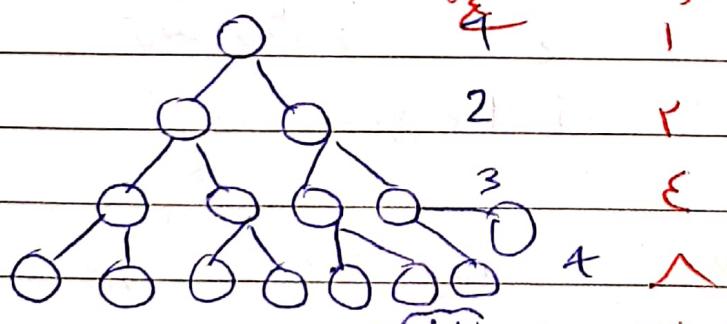
فرزندی هست فرزندی هست که می خواهد دارد

که هم این می خواهد

درختی = درخت دفعی



دقیق کردن بسیع بردار و بنیان یابی جدید خالی نمیتواند دارای چندین ریشه باشد

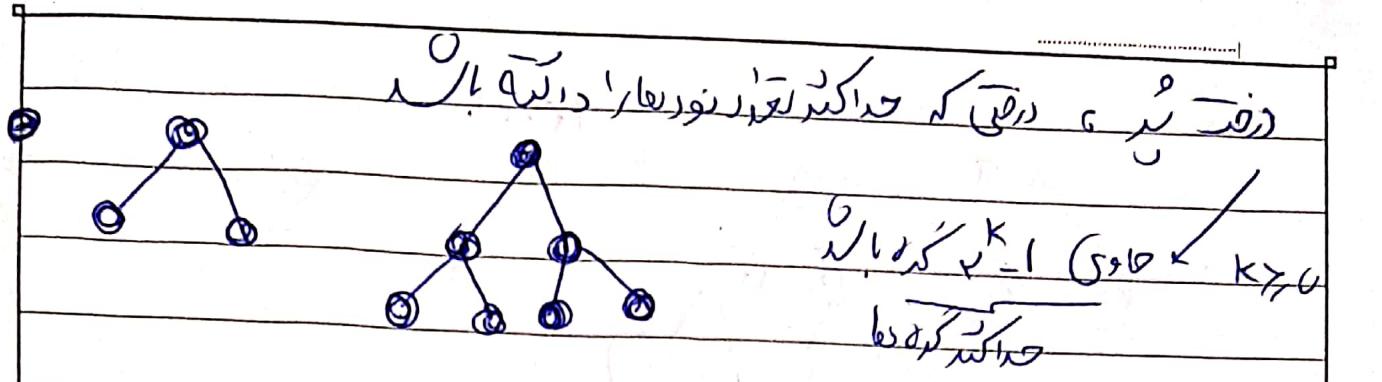


حاکم تعداد کل طبقه در لغات درخت مکانیکی صورتی

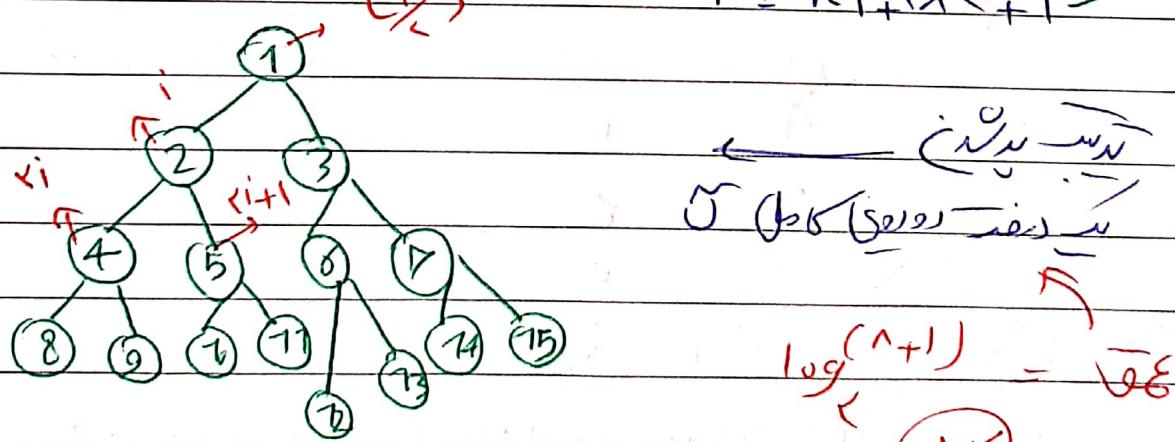
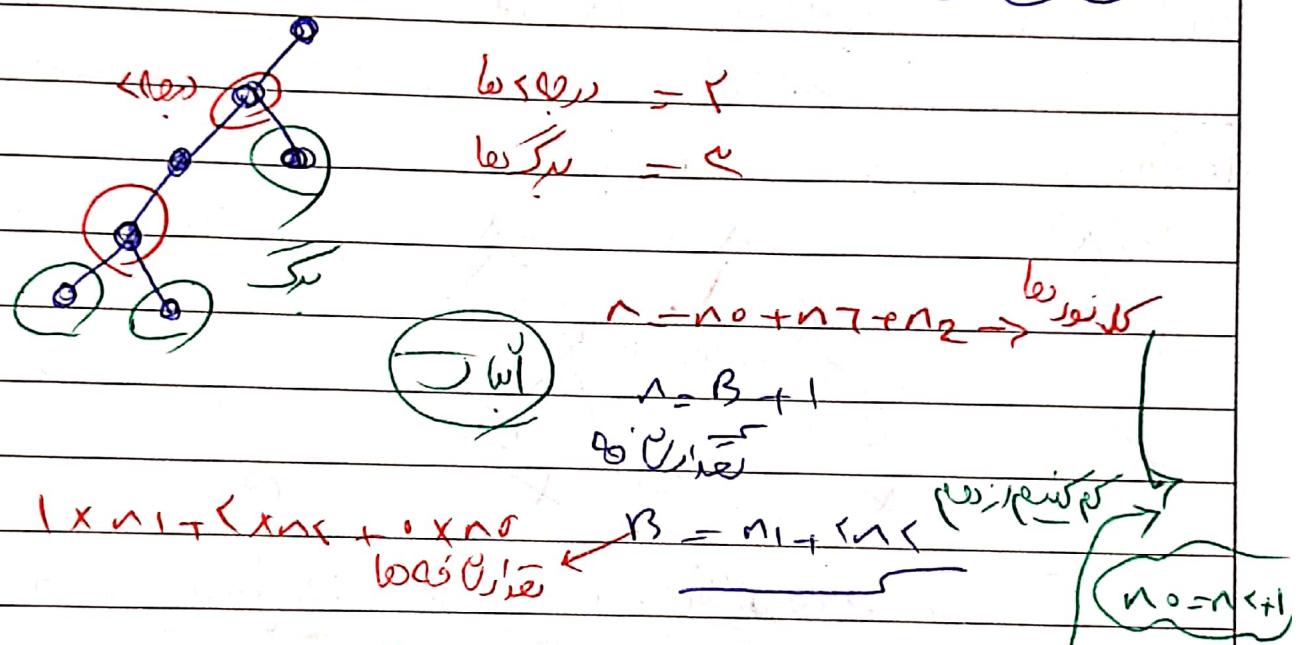
$K-1 \leq i \leq K$

$$T_n = \sum_{j=0}^{K-1} \binom{n}{j}$$

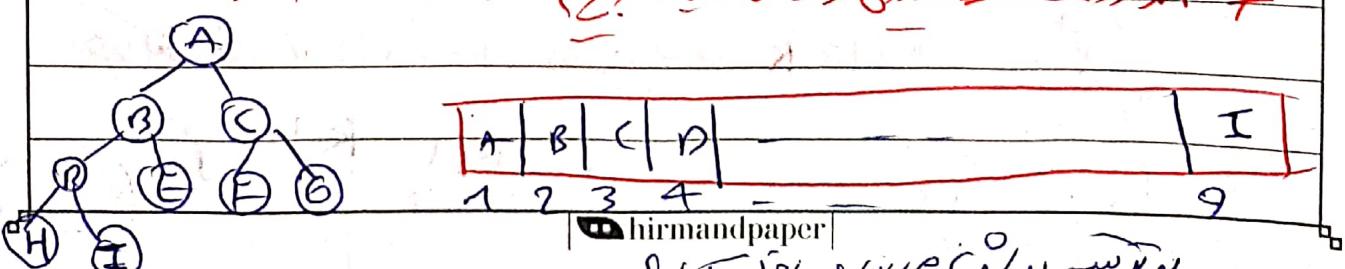
hirmandpaper

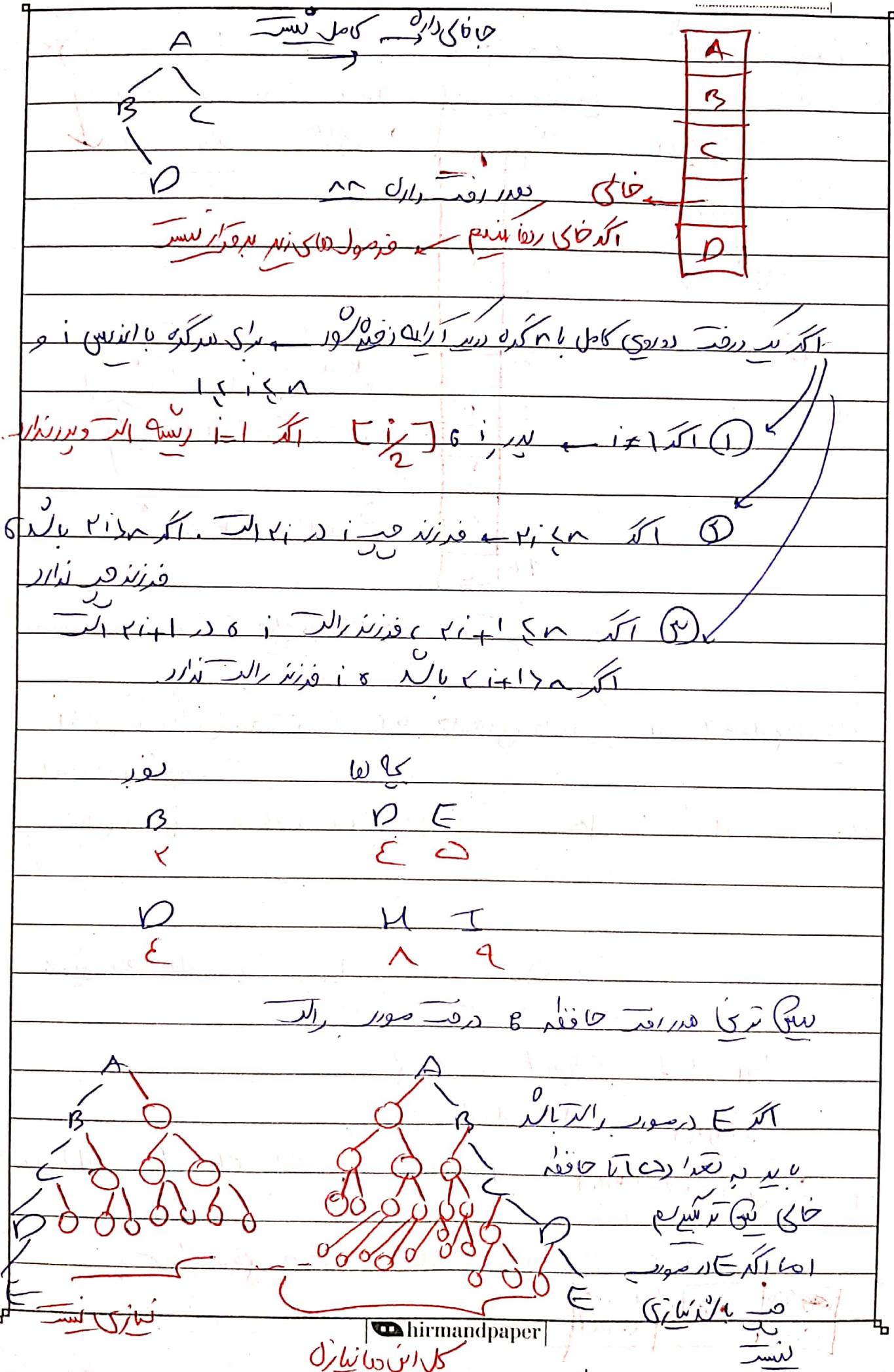


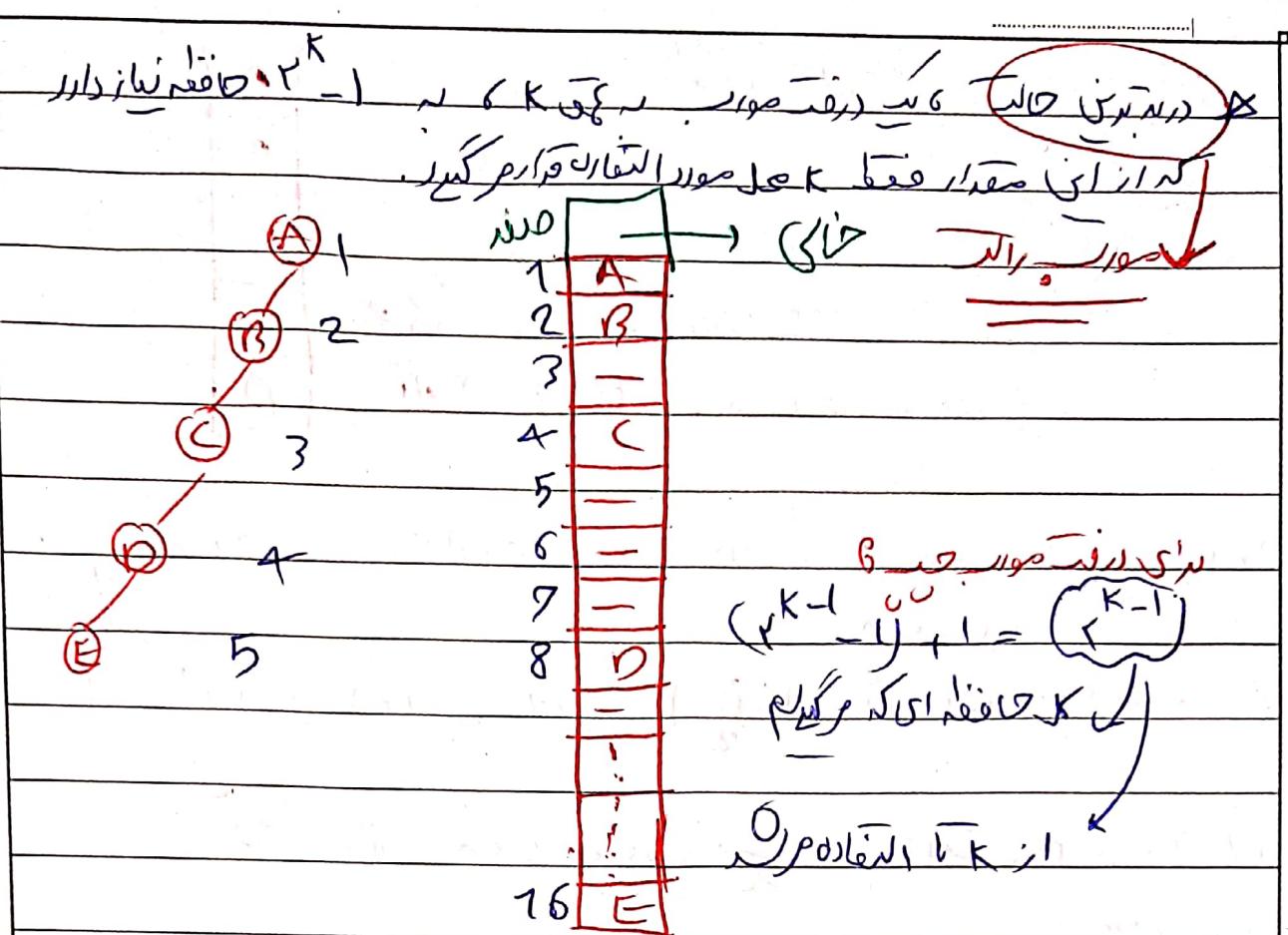
الف) n_0 چهار که کمترین نورده است و $n_0 = n - 1$



ب) در این روش حداقل تعداد نوردها را حافظه نمایی می‌کند







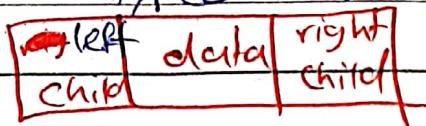
الليل نهار اللئن) دفتر در آرای ۸ طایبی و حذف شد (وزیر مسالم ۶ نهم ۱۴۵)

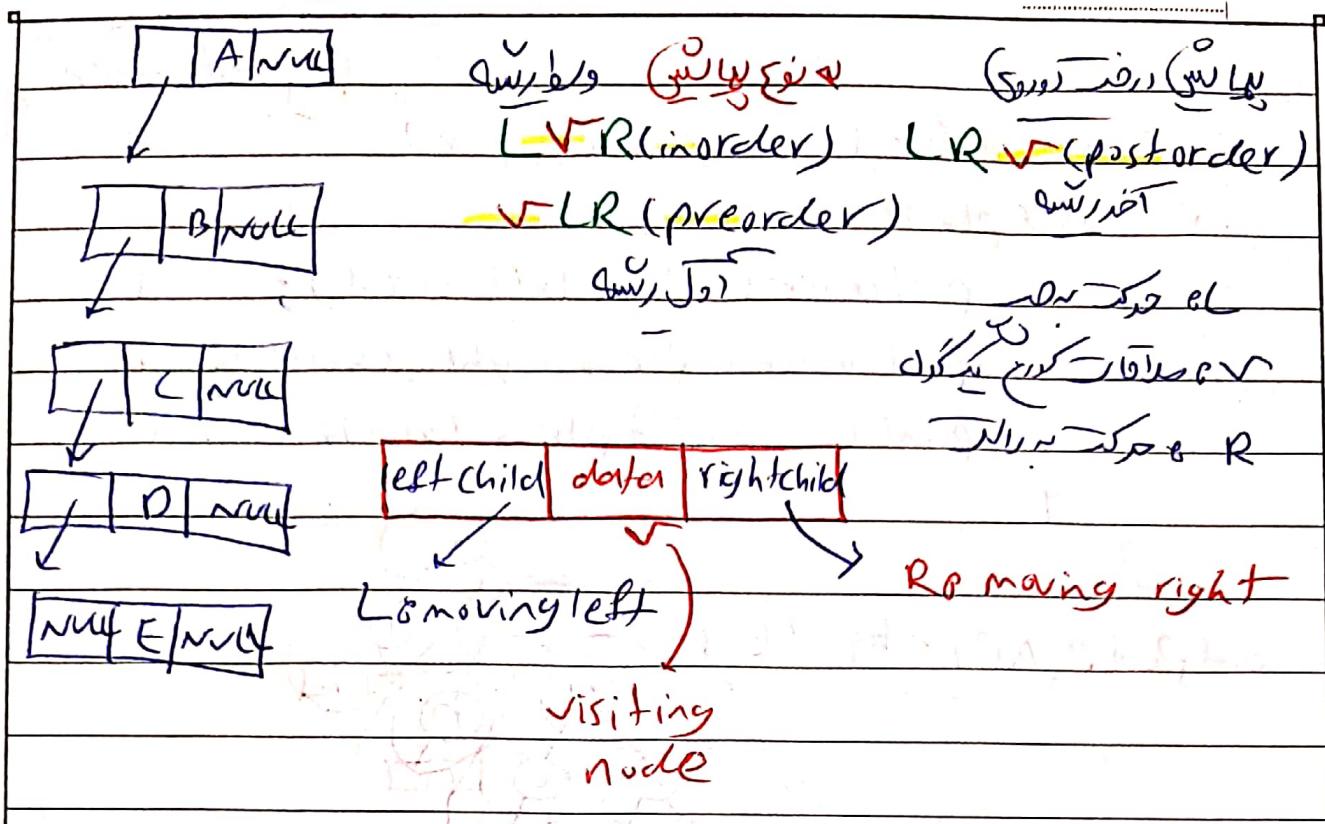
Signed Initials: Serge Tigani (Signature) (initials)

~~typedef struct node {~~

inf data; **data** left child right child
left-child right-child

~~typedef struct node * tree-pointer;~~





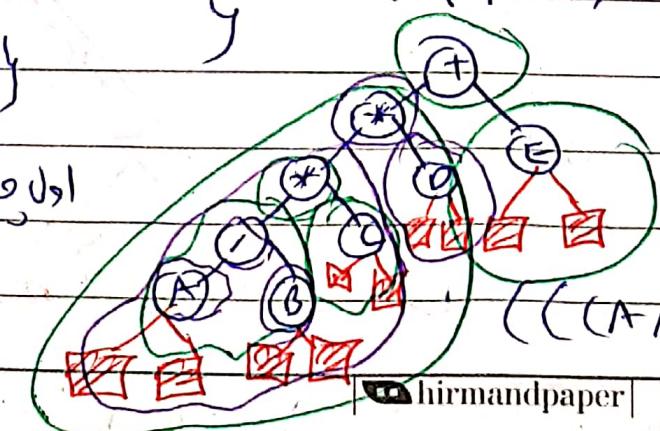
العنصر (جذع) في inorder
العنصر (جذع) في postorder
العنصر (جذع) في preorder

inorder (LRP) (visiting)

void inorder(tree pointer ptr) inorder (جذع)
 ↘ if (ptr) ↘
 inorder (ptr → left child); L

printf ("%d", ptr → data); ✓

inorder (ptr → right child); R



Postorder

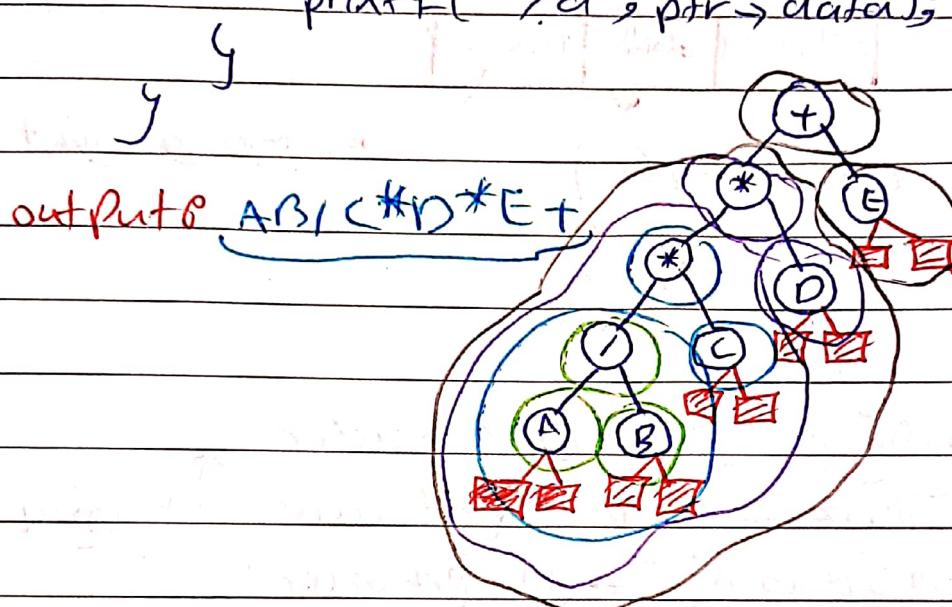
void postorder(tree_pointer ptr)

if (ptr) {

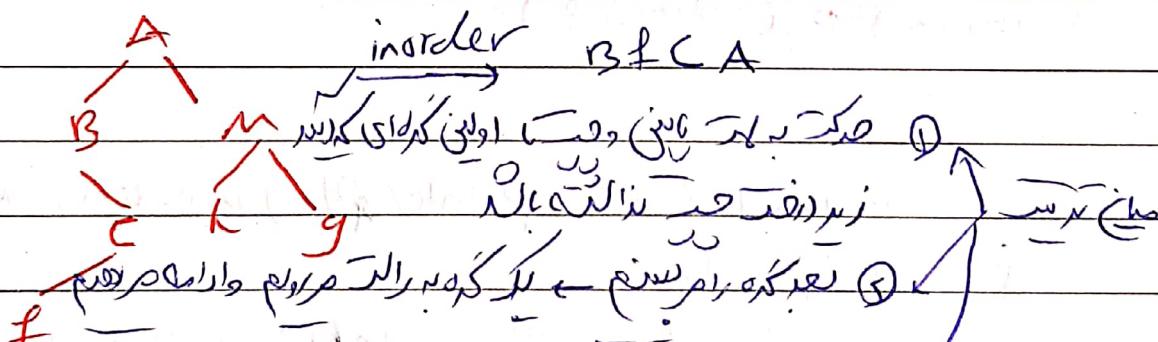
postorder(ptr->left_child); L

postorder(ptr->right_child); R

printf("%c", ptr->data); S, V



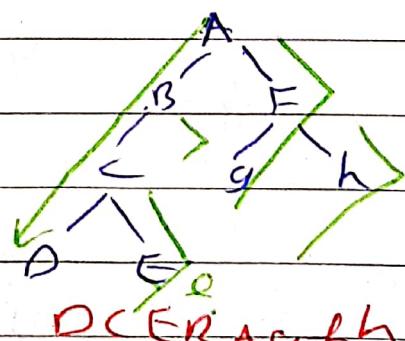
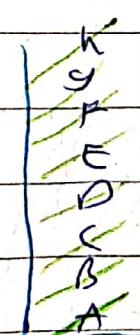
inorder B f C A



أولاً نزور العقد 左子树

(Root node)

ثُمَّ نزور العقد ذات اليمين 左右子树



Stack Inorder

مقدمة إلى الـ Stack

void iter_inorder(tree_pointer node)

 int top = -1; // initialize stack

 tree_pointer stack [max_stack_size];

 for (فون) {

Left

Inits

$O(n)$

زمان و مساحت

 for (; node != node->left_child);
 add (&top, node); // add to stack

 node = delete (&top); // delete from stack

 if (!node) break; // empty stack

 printf ("%d", node->data);

 node = node->right_child; // Right

Y

Y

① تأكيد صحة الـ stack (جهاز دخول و دخول مركب)

② إزالة آخر خارج الـ stack (جهاز دخول مركب)

③ إدخال كل خارج الـ stack (جهاز دخول مركب)

أولاً يدخل كل خارج الـ stack (جهاز دخول مركب)

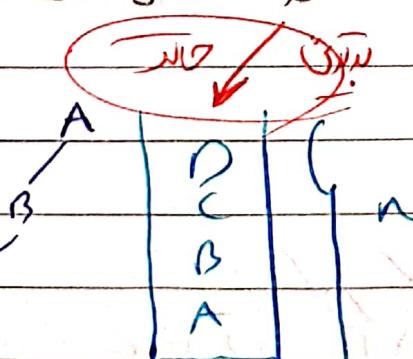
ثانياً يدخل كل خارج الـ stack (جهاز دخول مركب)

第三次 زيارتك

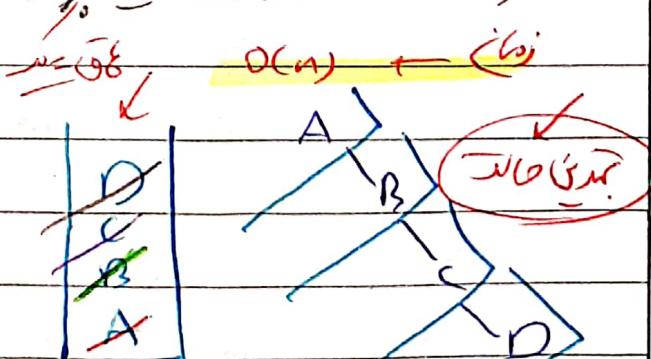
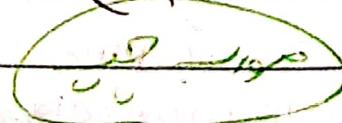
第三次 زيارتك

第三次 زيارتك

$O(n)$ time

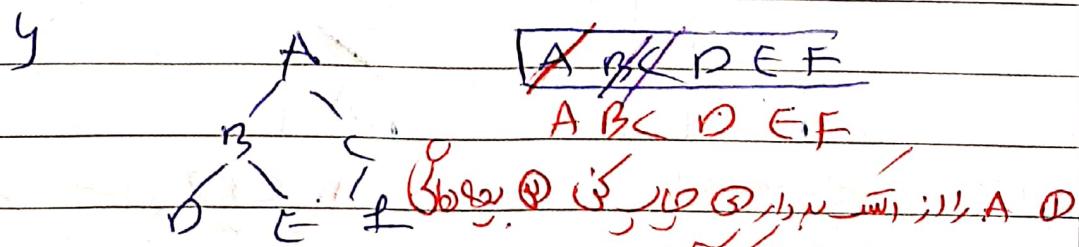


DFS بحسب التسلسل



ABCD

والآن دورة دخول صرفة، (بعد ذلك صرفة)



بادی مکانیزم

کوئی نہ رہے تاکی (دروجی) Postorder (پوسٹ ارڈر) (اول فیر بعد والے جمع)

tree pointer) copy (tree pointer original)

(2nd) tree pointer temp

if (original)

~~tmp = (tree pointer)malloc(sizeof(node));~~

IF (IS_FULL(temp))

```
fprintf(stderr, "The memory is full");
```

exit(-1);

temp, left child - copy (original, left child)

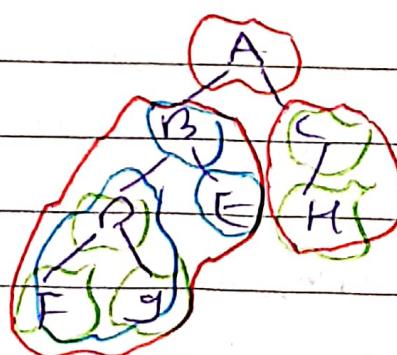
~~temp~~ → right child = ~ ~ right

temp, data = original → data

return temps

return NULL;

4



(مُلْك) (أَخْلَق) (صَانِفَة)

bill, bill, ① = i, i re-i, e, o)

int equal(tree pointer first, tree pointer second)

↓

return (!first && !second) || (first &&

second && (first->data == second->data)) ||

L ← equal(first->left_child, second->left_child) &&

R ← equal(~->right, ~, ~->right &&)

) true

if (first && second) first is

1. both ① is null, return second, first is

2. if ① is not null, ② is

w w ~ ~

preorder

Gravitational

$m_1 \vee (\neg m_1 \wedge m_2)$

for $m_1 = T, L$

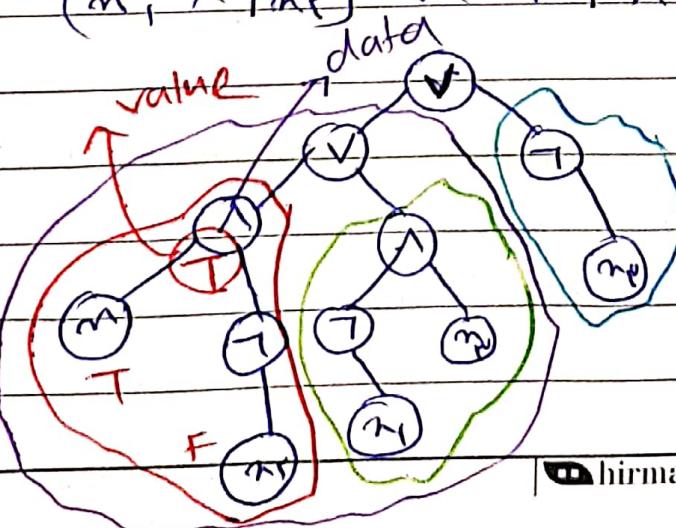
for $m_2 = T, R \rightarrow r \times g(n)$

for $m_2 = L, R$

($\neg m_1 \vee m_2$) $\neg m_1 \vee m_2$ $\neg m_1 \vee m_2$ $\neg m_1 \vee m_2$ $\neg m_1 \vee m_2$ $\neg m_1 \vee m_2$

$(m_1 \wedge \neg m_2) \vee (\neg m_1 \wedge m_2) \vee \neg m_1 \vee m_2$

postorder



Gravitational Gravity

دستگاه مکانیکی (G)

values is True , values

مکانیکی

سکون

info, data - {not, and, or, true, false}

این دوییزی = value

۱, ۰, ۱, ۰, ۱, ۰

left	data	value	right
child			child

typedef enum {not, and, or, true, false} logical;

typedef struct node *tree_pointer;

typedef struct node {

tree_pointer left child;

logical data; نوع داده

short int value; نیازی نیست

y; یک عدد

- میدیم این دوییزی ها را (sub data) را (یعنی گذشت)

void post_order_eval(tree_pointer node)

y

if (node) {

L → post_order_eval(node → left_child);

R → ~ ~ right ~

✓ → switch (node → data) {

پیغامی
case not & node → value = !node, right child
breaks; → value

case and & node → value = node, right child,

 && node → left_child → value;

 breaks;

case or & node → value = node → right - child,

 || node → left_child → value ;

 breaks;

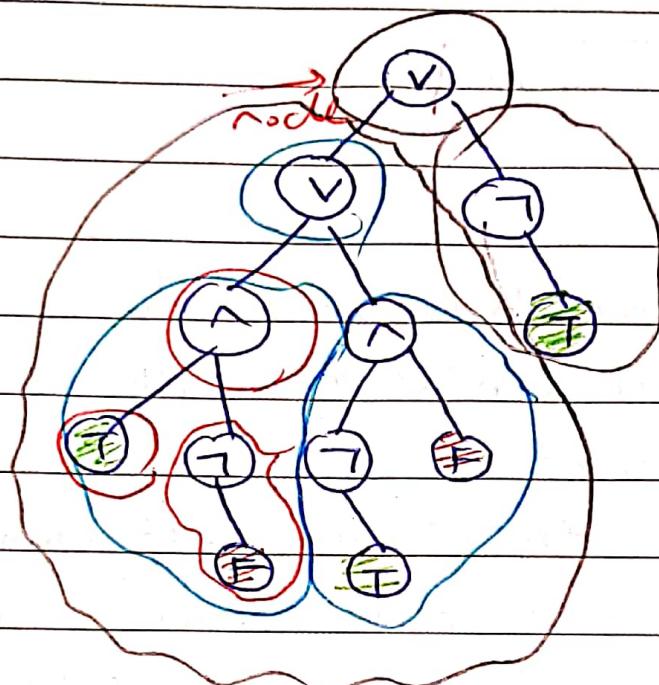
value

$\text{if } \text{true} \rightarrow \text{value} = \text{TRUE}$

case true : node->value = TRUE; & node->right
break; \rightarrow live TRUE go value is \neg TRUE

case false : node->value = FALSE;

y
y
y



int count(ptr) \rightarrow node value
{ if (ptr)

m = count(ptr->left);

n = n (\rightarrow right);

return (1+m+n);

y

return 0;

y

نحوی کسی نه

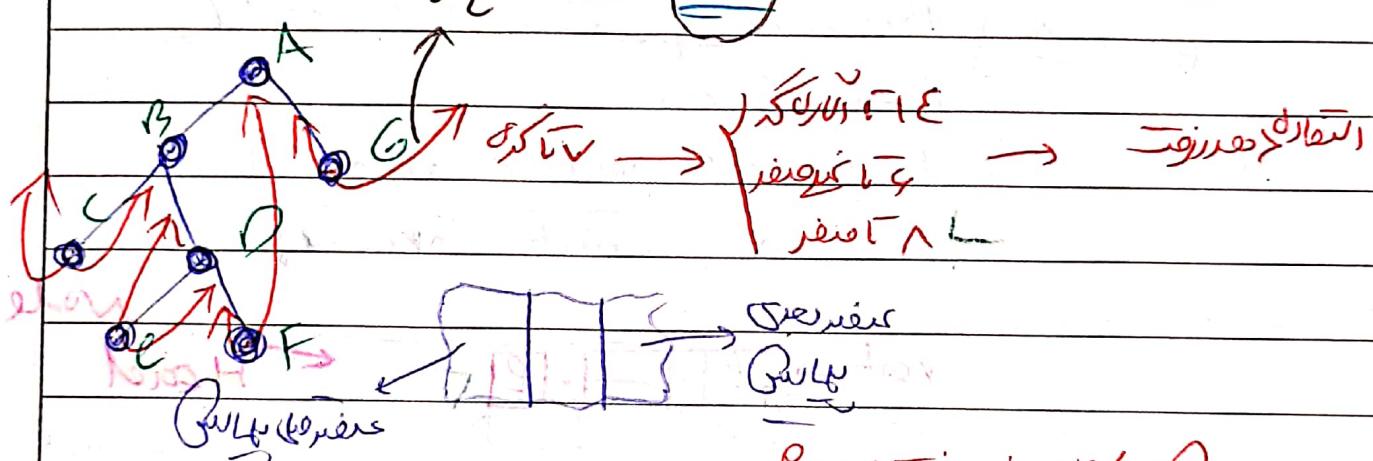
node

نحوی کسی نه

نحوی کسی نه

نحوی کسی نه

نحوی کسی نه



Inorder is in order Qul

$\leftarrow B \text{ (left)} \leftarrow C \text{ (left) } E \text{ (left) } F \text{ (left) } A \text{ (left) } G \text{ (left) }$

$E = \text{right} \rightarrow \text{child}$

نحوی کسی نه

نحوی کسی نه

نحوی کسی نه

نحوی کسی نه

Inorder

H D I B E A F C G

left-thread \Rightarrow boolean miss (initial \star)
right = \sim

N_l - True $\leftarrow \text{ptr}, \text{left-thread}$

لهم انشئني بغير ذنب

```
typedef struct threaded tree *threadded  
typedef struct threaded_tree ^
```

100% Natural

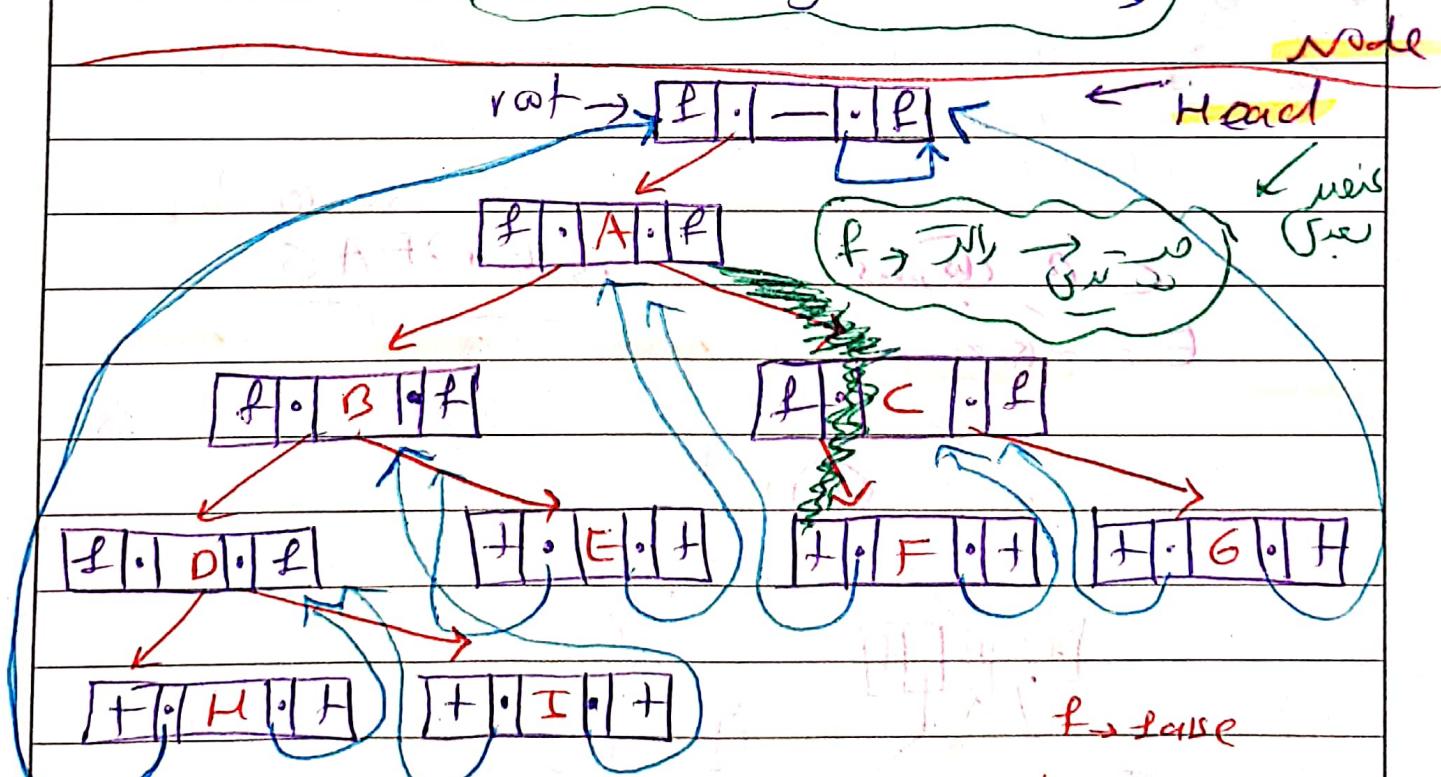
short int left_thread;

threaded pointer left child:

char dates:

~~threechild_pointer~~ right_child;

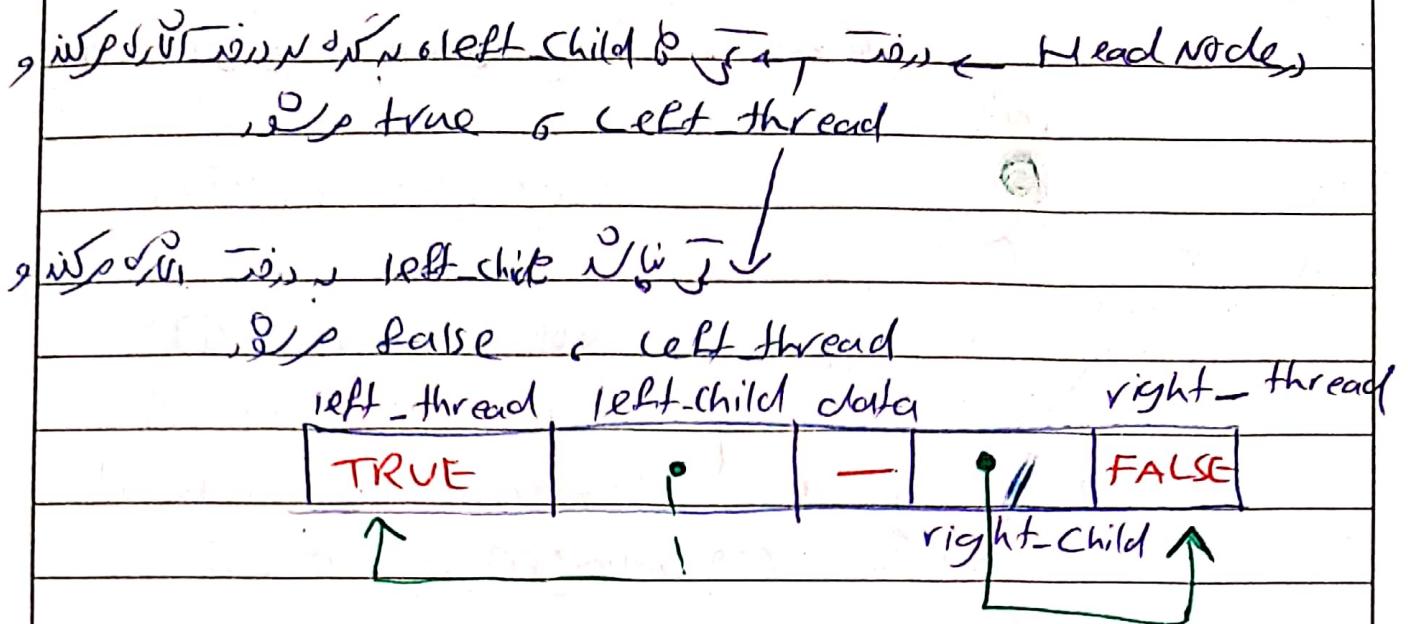
short int right threads



root UDIBEAFCG root

اڑکی سفید رہیں گے اگرچہ رام دنیا کی باتیں سنائیں گے

میراند پردازی + دینی اسلامی + علی الہ بالآخر و علیہ السلام و صریح فرموده (ع)



$i \leftarrow$ (succ) $i \leftarrow$ (successor)

threaded = pointer insucc (threaded - pointer tree)

threaded = pointer temp;

temp = tree \rightarrow right child;

if (!tree \rightarrow right-thread) \rightarrow $i \leftarrow$ \bar{i}

while (!temp \rightarrow left-thread) \rightarrow $i \leftarrow$ (succ) thread

temp = temp \rightarrow left child;

return temp;

y

void tinorder (threaded - pointer tree) {

 threaded - pointer temp = tree;

 for (; ;) {

 temp = insucc (temp);

Time Complexity

 if (temp = = tree)

$O(n)$

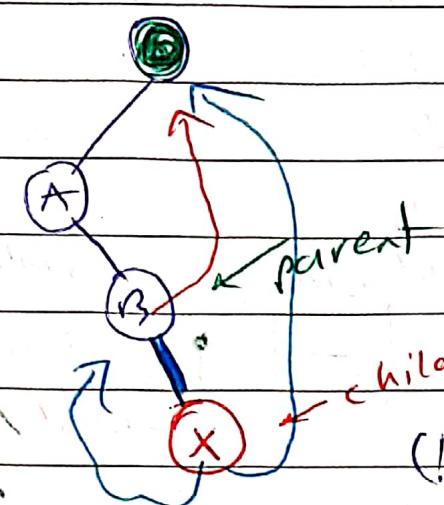
 break;

 printf ("%c", temp->data);

y

فہرست کرنے سے کہا جاتا ہے (جیسا کہ اس کا نام)

root

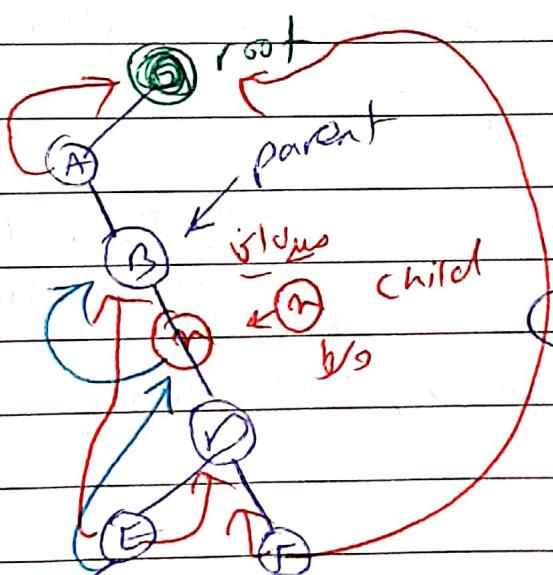


B (جیسا کہ اس کا نام)

(جیسا کہ اس کا نام)

child (جیسا کہ اس کا نام)

لگانے والے بھائی



(جیسا کہ اس کا نام)

اچھا ہے

B (جیسا کہ اس کا نام)

کوئی بھائی نہیں

(جیسا کہ اس کا نام)

کہا جاتا ہے (جیسا کہ اس کا نام)

A

B

M

نفر جی ①

C ②

نفر جی ③

N

D

E

H

F

G

hirmandpaper

35

36

37

(ج) داده هایی را در نظر بگیر که در آنها از پرده های مرتبط است.

void insert_right(thread_pointer parent, thread_pointer child){

threaded_pointer temp;

child → right_child = parent → right_child;

child → right_thread = parent → right_thread;

child → left_child = parent;

child → left_thread = TRUE;

parent → right_child = child;

parent → right_thread = FALSE;

if (!child → right_thread){

temp = insucc(child);

temp → left_child = child;

g

دلتا

o(n)
o(1)

o(n)
o(1)

پیوسته

max = ۰



o(n)
o(1)

o(n)

حذف
o(1)

حذف

حذف

o(1)

o(n)

o(n)

حذف حذف
o(1) o(1)

insert

delete

Max heap

$O(\log n)$

$O(\log n)$

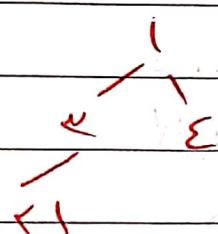
(lo <= 4m)

گزینه های مختلفی داریم که ایجاد شوند و max tree
نامیدیم

پیشنهاد شده است (min tree) $\left(\frac{m}{n}\right)$ تا نظر داشته باشیم که ایجاد شوند و min tree
 $lo \leq \frac{n}{m} < 4m$ $\left(\frac{m}{n}\right)$

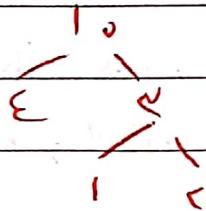
max tree (بلطفاً) تا نظر داشته باشیم که ایجاد شوند و max heap

min tree \rightarrow 8 min



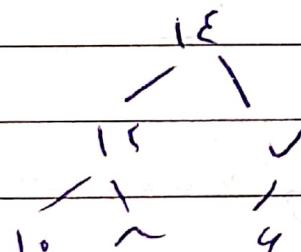
min tree ✓

min heap ✓

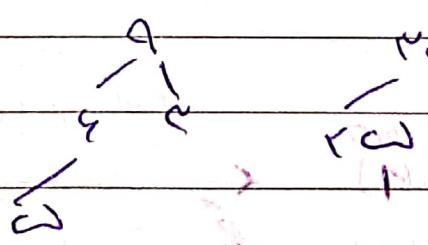


maxtree ✓

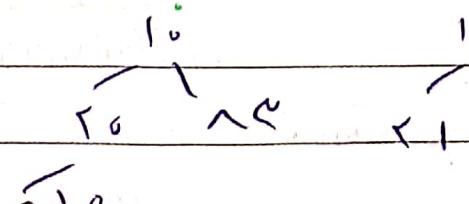
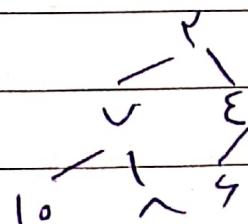
max heap X



و max heap



و min heap



(111)

max heap since crucial

void insert_max_heap(element item, int *n)

{

int i;

if (HEAP_FULL(*n))

fprintf(stderr, "the heap is full ");

exit(7);

}

*n++ = i = ++(*n);

big O (j)

O_Y

while ((i != 1) && (item.key > heap[i/2].key))

Y

heap[i] = heap[i/2];

parent link

i /= 2; → item up heap

Y

heap[i] = item;

O(log n)

Y

50

o + < & o & v
X | I E K V | o & S T A

I E E A.

E K I E

I E

E K V
افلام مترجم

رایج فرم

maxHeap is here

element delete_max_heap (int *n)

no logic

int parents, child;

element item, temp;

if (HEAP_EMPTY (*n)) if ($n = 0$)
item heap [1];
printf (stderr, "the heap is empty");
exit (1);

loop until $n = \text{max}$

item = heap [1];

temp = heap [$*n - 1$];

parent = 1; \checkmark مکانیکی

child = 2;

$O(\log n)$

- دو (بین)

- دو تا

while (child <= *n) {

if (child <= *n) && (heap [child].key
< heap [child + 1].key)

child++;

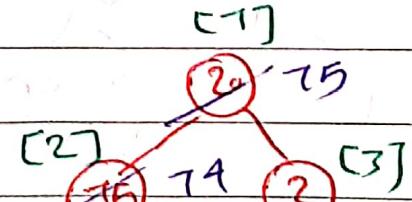
if (temp.key >= heap [child].key) break;

heap [parent] = heap [child];

parent = child;

child += 2;

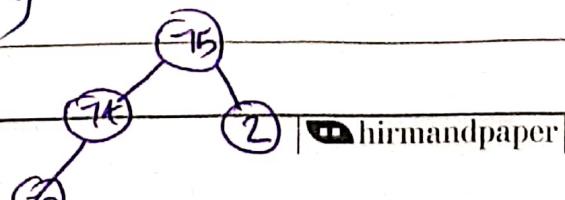
y



heap [parent] = temp;

return item;

y

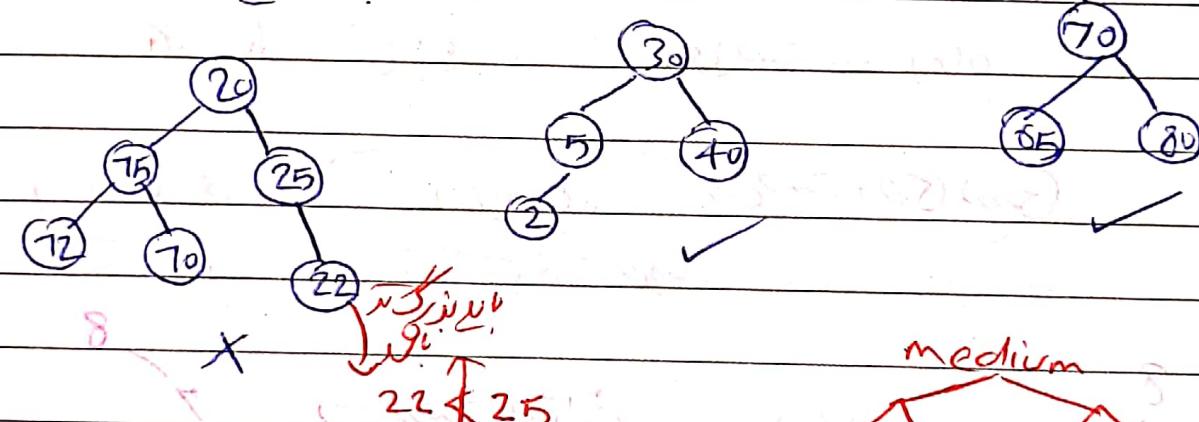


$O(10g, n)$ \leftarrow heap gives max/min \rightarrow $O(n)$

جستجو در درخت جستجوی مترقبه \leftarrow $O(n)$ \leftarrow heap is always sorted \rightarrow $O(n)$

(ورودی) مرتب

جستجوی ایجاد شده \leftarrow (جستجوی مترقبه) \leftarrow $O(n)$



tree-pointer search (tree pointer root, int key)

(جستجوی بازگشتی)

if (!root) return null;

if (key == root->data) return root;

if (key < root->data)

return search(root->left_child, key);

return search(root->right_child, key);

tree pointer search 2 (tree pointer tree, int key)

O(h)

while (tree) {

if (key == tree->data) return tree;

if (key < tree->data)

tree = tree -> left_child;

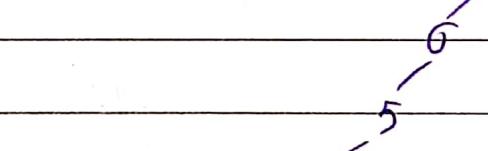
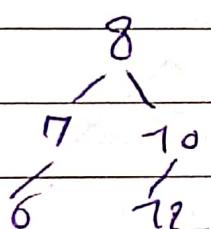
else

tree = tree -> right_child;

return null; \rightarrow null

O(h) \rightarrow $n \log n$ \rightarrow h \rightarrow

(جاء) \rightarrow $n \log n$ \leftarrow $n \log n < O(n)$
 \rightarrow خواص \rightarrow $n \log n$



أكبر قيمة \rightarrow \max \rightarrow \max \rightarrow \min

أكبر قيمة \rightarrow \max \rightarrow \max \rightarrow \min \rightarrow $O(h)$

جست و جوی راه راه (انواع اندیشه)

tree_Pointer Index search (tree_Pointer tree, int K)

جسٹیس ایج

tree = pointer + tree;

while (f)

if ($k = -t \rightarrow \text{leftSize}$) return t ;

~~if (k < t -> leftsize) t = -> leftchild;~~

else {

~~K - = + → left size;~~

$t = t \rightarrow \text{rightchild}$

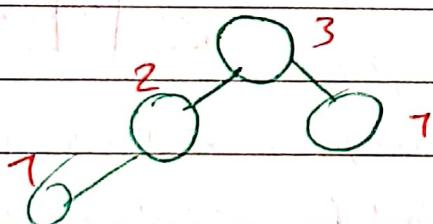
o(h) |

4

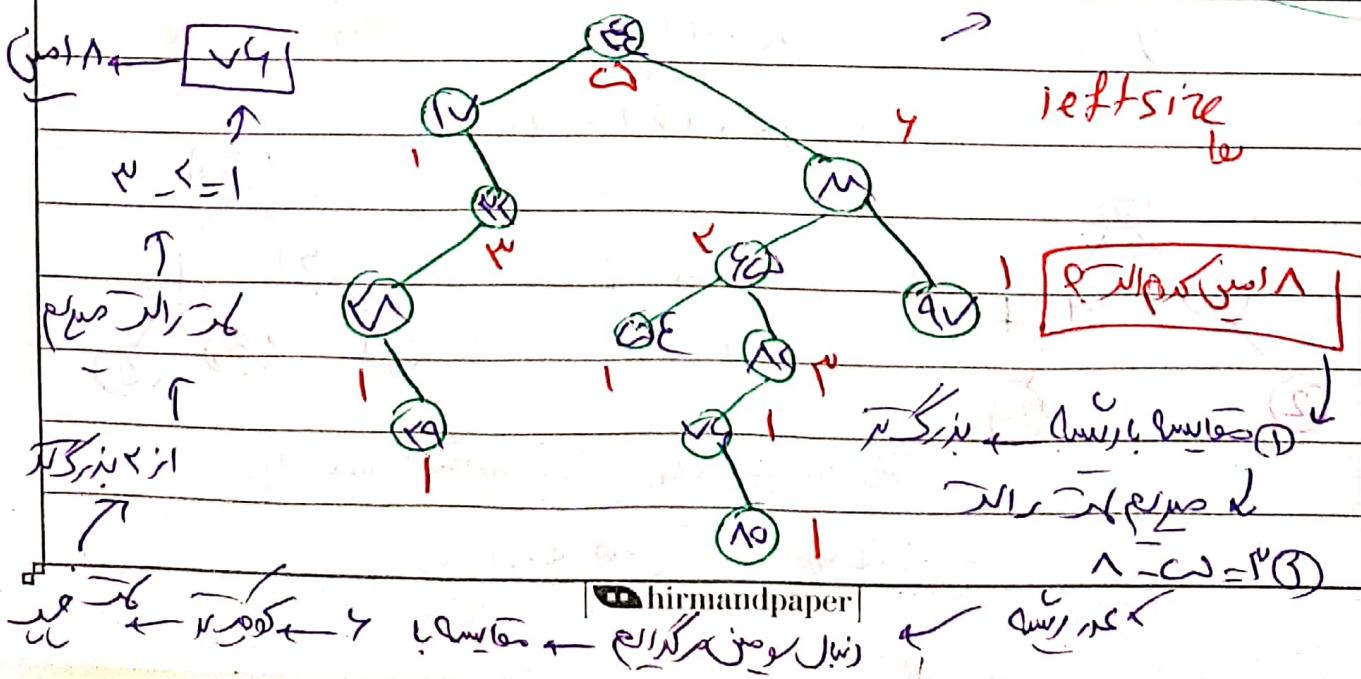
return 0;

4

* دنگون و راک سے مدد افغان (عطا، گنجویانی نسیفر-صریف)



$$49 = \sin(2\pi)$$



(جواب) (جواب تابعی است، پس از اینجا که می خواهیم)

int insert_node(tree pointer tree, int n)

tree pointer p = tree, q = 0;

while(p)

q = p;

if (n == p->data) return false; // موجود است

if (n < p->data) p = p->leftchild;

else p = p->rightchild;

y \rightarrow previous node

// perform insertion

q = 0; parent
of new

p = (tree pointer) malloc (sizeof(node));

p->leftchild = p->rightchild = 0; start

0/NULL

p->data = n;

NULL



if (!tree) tree = p;

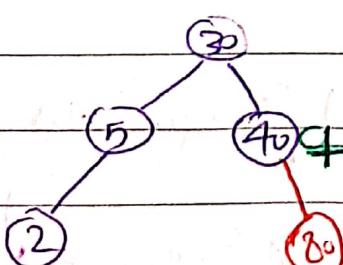
else if (n < q->data) q->leftchild = p;

else q->rightchild = p; \rightarrow (Circular)

return true;

O(h)

$O(\log n) < O(h) < O(n)$



افزونه ایجاد

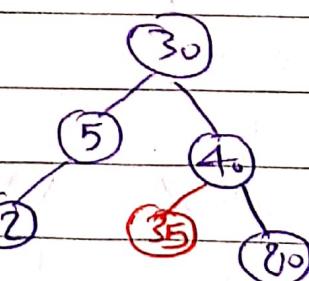
ایجاد نو

ایجاد نو

افزونه ایجاد

ایجاد نو

ایجاد نو



پس از اینجا می خواهیم داد

از زیر نو

حذف بی عضو از درخت جستجوی (عویض)

۱۶ حذف ععنو بیگ

۱۷ حذف ععنو غیر بیگ که فتحه بی دارد

۱۸ حذف ععنو بیگ که دو بی دارد. بیگ تری ععنو را زیدف خواهد داشت
که بی تری ععنو را زیدف خواهد داشت حال خوب نیست خوب نیست
لئنکه با این عمل قابل خود حذف نیست

راز \rightarrow بیگ تری

حذف را زد را زد

(حالات ۱ و ۲) دارد

* ارتفاع درخت جستجوی (عویض) میان ععنو و بیگ

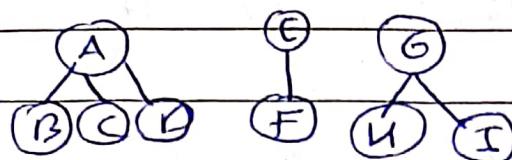
* وقت که بیگ را حذف و اضافه نماییم صورت تغییری ایجاد نماید ارتفاع را بدستور

$$O(\log n) = \text{ستون}$$

* در عویضی جستجوی مترادف در بیگی حالات عقیقی

AVL, 2-3 / red-black میان

صویز ۵۸۶۷۰ دفتر صدا
از هر دفتر یک جنل به صدا
با این روش می‌توان از هر دفتر (دوره) یک جنل با (دوره) جنل



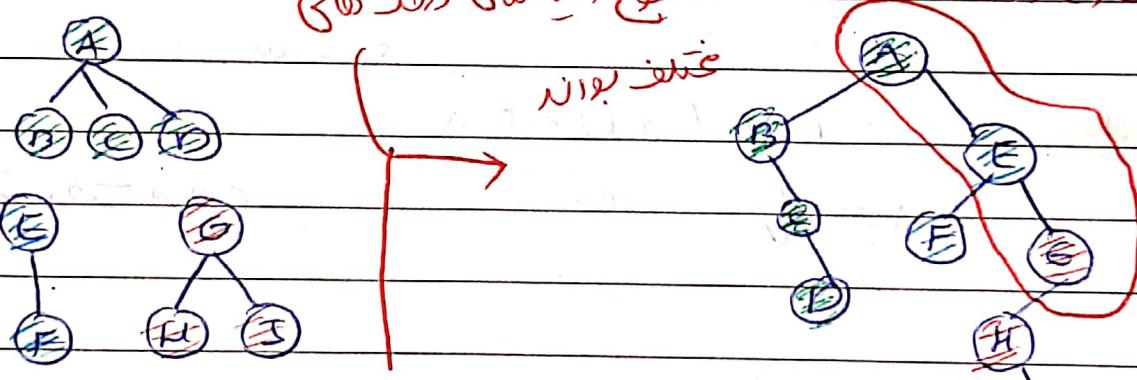
جنل با ۳ دفتر

بسیل جنل ر رفت (دوره) بسیل کنم

۵) الفال دفتر (ساکن) از خانواده (گروه)

پر کردن

خواهش (دفتر های) از خانواده



فرزند = جنل

خواهش = نیز

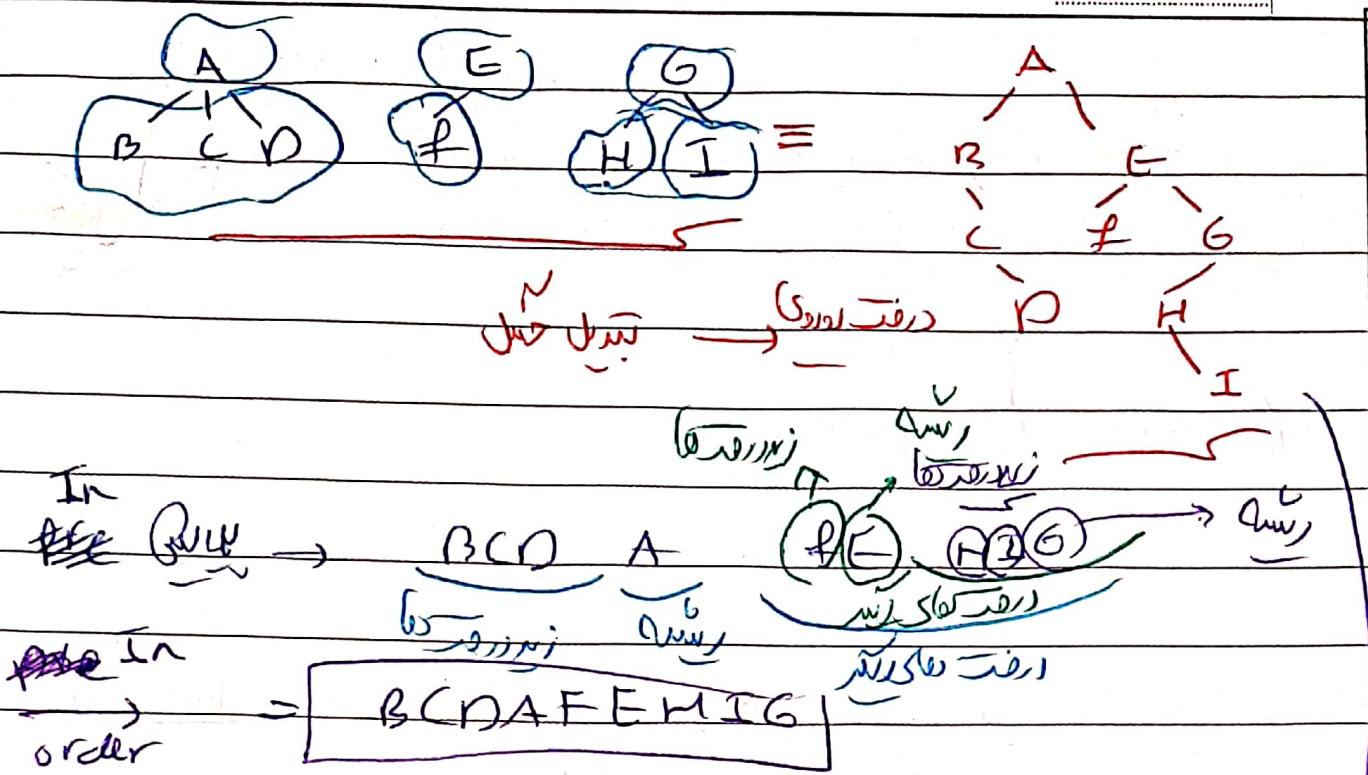
۱) زیر دفتر اول (۲) زیر دفتر اول (۳) زیر دفتر اول

۴) پرف دفتر اول (۵) پرف دفتر اول (۶) پرف دفتر اول

۷) زیر دفتر اول (۸) زیر دفتر اول (۹) زیر دفتر اول

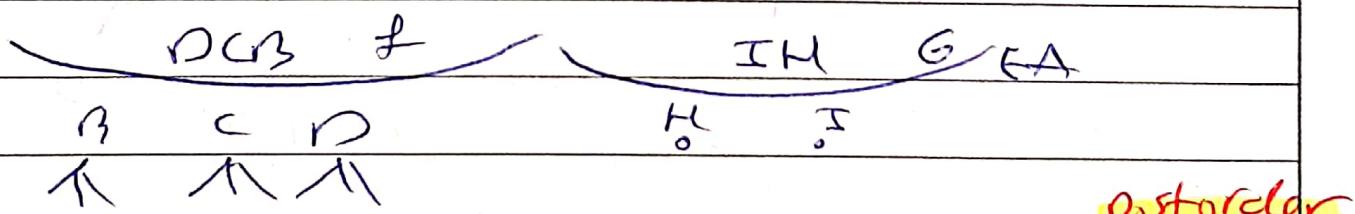
۱۰) درخت (دفتر اول)

۱۱) زیر دفتر اول (۱۲) زیر دفتر اول (۱۳) زیر دفتر اول



pre order ~~in~~ ~~out~~ ~~in~~ ~~order~~ = **A B C D E F G H I** اول ترتیب خروجی (مقداری که خروجی)

post ~~in~~ ~~out~~ ~~in~~ ~~order~~ = **D C B F I H G E A**

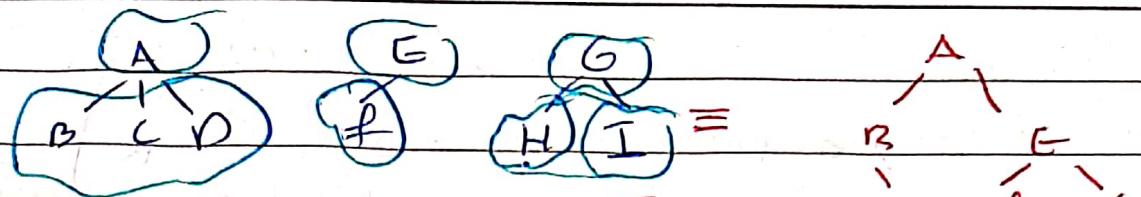


D C B F I H G E A

postorder
warning

(جمله) (جمله) (جمله) (جمله)

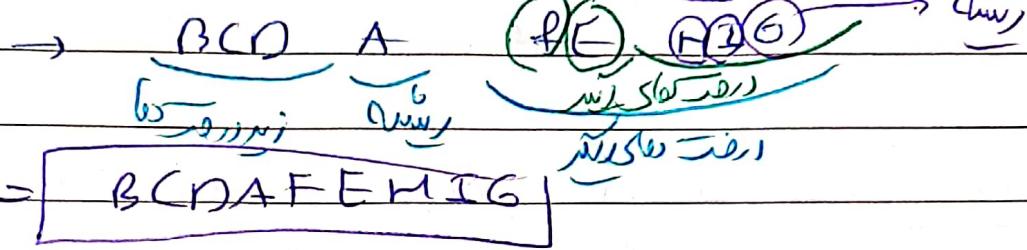
جمله = جمله pre_in (جمله) pre_in (جمله)



جیل جیل → پیش از این

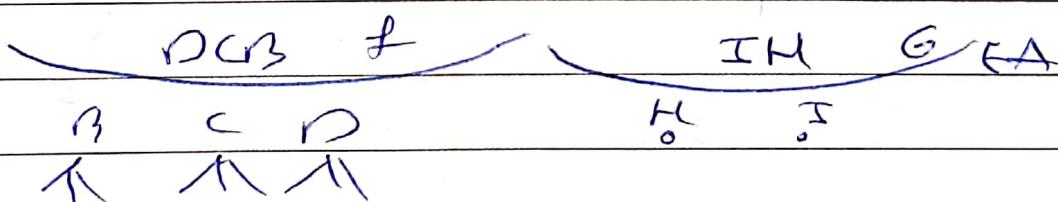
In
pre
order

~~post~~ In
order



pre order جیل GABCD EFGHI اول تا آخر

post جیل DCBFIHGEA



post order

DCBFIHGEA

(جیل خود را درست کنید) (پیش از این خود را درست کنید) ↗ ↘

جیل = خود را درست کنید پیش از این خود را درست کنید

نحوه ای ار داده
این تعداد را چند می کنیم

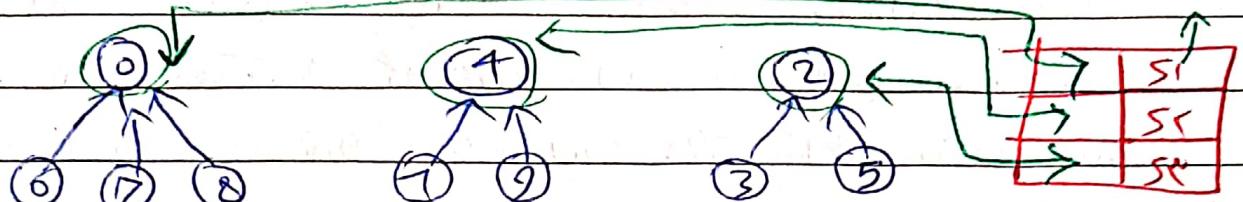
کوچکترین مجموعه

اجماع صورتی

$s_i \cup s_j = \{ \text{all elements } m \text{ such that } m \text{ is in } s_i \text{ or } s_j \}$

نحوه ای ار داده و $\text{Find}(i) \oplus \text{Find}(j)$

باید چه کنم



$set_1 = \{0, 6, 7, 8\}$

$set_3 = \{2, 3, 5\}$

از فرزند های اول رجوعات تا پایه بولی

بررسی روش کاری برای اینجا

نحوه ای ار

فرزند را

✓

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

✗

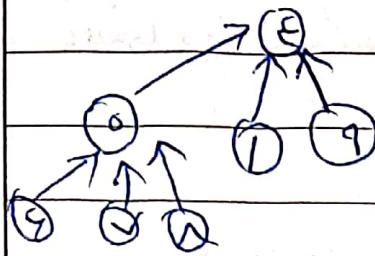
✗

✗

✗

✗

✗



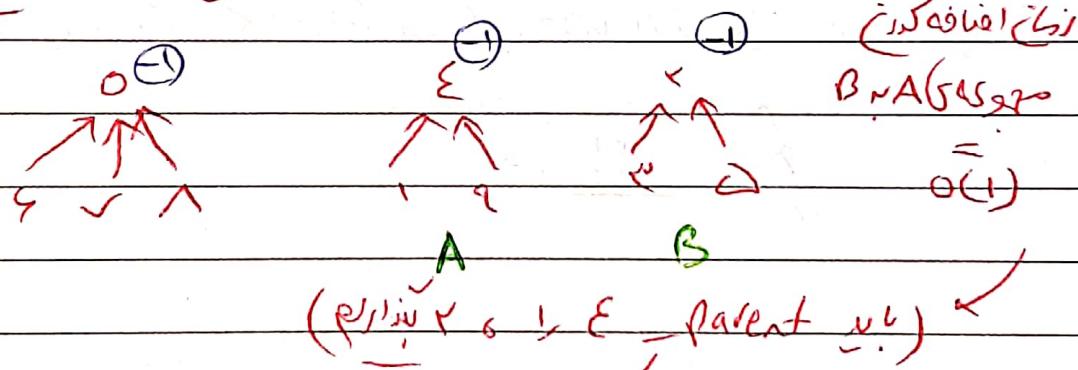
باید که ریشه که تعاریف نداشته باشد
و همچنان که پس از آن می‌باشد که
که قدرتی دارد.

جواب سه گزینه به دوران ارتی

۱) اینجا بگوییم که این کامپیوئن می‌توان از اینها استفاده کرد
۲) می‌دانیم که در اینجا ۰ و ۱ را می‌توان از اینها استفاده کرد
۳) می‌دانیم که در اینجا ۰ و ۱ را می‌توان از اینها استفاده کرد

i	[.]	[1]	[r]	[e]	[s]	[d]	[v]	[h]	[g]
parent	-1	E	-1	-1	r	0	0	0	E

خوبی رسمی اند



(جواب)

void UnionT(int i, int j)

parent[i] = j;

y

int FindT(int i)

for(; parent[i] >= 0; i = parent[i]);

return i;

parent < 0

ارجع

o(h)

Union Find

جزوی و متعال توانع

$$O(n^2) = \text{بررسیک} \cdot \text{خواهش} \quad (1)$$



با این قانون خوب صرفاً این پیوستگی را کاهش دار $\Omega(n^2)$ \rightarrow $O(n)$

union(0) \rightarrow find(0)

union(0) \rightarrow find(0)



Union($n < m < 1$) \rightarrow find(0)

$O(n^2)$ مجموع

union(i,j) \rightarrow (نیز) \rightarrow (قانون خوب)

اگر هزار کوکا در درخت باشیم؛ که از نعداد کوکا در درخت باید $\Omega(n^2)$ باشد

لذا از ورنسایی میتوانیم $\Omega(n^2)$ را بسیار کم کنیم

$\text{Ans}_i - \text{parent} = \text{Q} \rightarrow$ (نیز) \rightarrow (نیز)

این سیستم پس از اینکه هزار کوکا در درخت باشیم $\Omega(n)$ باشد

هر کوکا در درخت میتواند مادر خود را در صفت parent معرفی کند

weighted Union \rightarrow (نیز) \rightarrow (نیز) \rightarrow (نیز) \rightarrow (نیز) \rightarrow (نیز)

$\text{parent}[i] = \text{parent}[j]$
 $i \rightarrow \text{parent}[i]$
 $j \rightarrow \text{parent}[j]$

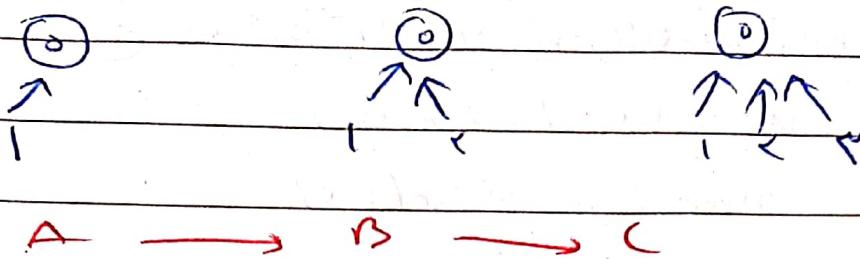
void weightedUnion(int i, int j)

 int temp = parent[i] + parent[j];
 if (parent[i] > parent[j]) \rightarrow نیز \rightarrow نیز \rightarrow نیز
 parent[i] = j;

 parent[j] = temp; \rightarrow نیز \rightarrow نیز

 else \rightarrow نیز \rightarrow نیز
 parent[j] = i;
 parent[i] = temp;

weighted union(α) و weighted union(β)



الآن (نحو) weighted union (الجتمع) و union (التحان) از

$\Theta(\log n)$ \rightarrow $\Theta(\log n)$ \rightarrow $\Theta(\log n)$

$\Theta(\log n)$ \rightarrow $\Theta(\log n)$ \rightarrow $\Theta(\log n)$

$= \Theta(\log n)$ \rightarrow $\Theta(\log n)$ \rightarrow $\Theta(\log n)$

$\Theta(u + f * \log n)$ \rightarrow $\Theta(\log u)$

$\Theta(\log n)$ \rightarrow $\Theta(\log n)$ \rightarrow $\Theta(\log n)$

$\Theta(\log n)$

$\Theta(1)$ $\Theta(1)$ $\Theta(1)$ $\Theta(1)$ $\Theta(1)$ $\Theta(1)$ $\Theta(1)$ $\Theta(1)$

$\Theta(1)$ $\Theta(1)$ $\Theta(1)$ $\Theta(1)$ $\Theta(1)$ $\Theta(1)$ $\Theta(1)$ $\Theta(1)$



Collapsing rule

نیز 2010

$\forall i \text{ Parent}[i] \neq \text{root}(i)$, $i \rightarrow j$ (جی کو اسے تیار کرنا ہے جو i کو j کی نئی نئی root(i) کا parent(j) کے لئے ہے)

ایسے ہے سے کیا کرنا ہے find(j) کا نتیجہ میں i کو find(j) کا نتیجہ میں کھو کر i کو j کا parent(j) کے لئے ہے

ایسے ہے کیا کرنا ہے find(j) کا نتیجہ

int CollapsingFind(int i) {

 int root, trail, lead;

 for(root = i; parent[root] >= 0; root = parent[root])

 for(trail = i; trail != root; trail = parent[trail])

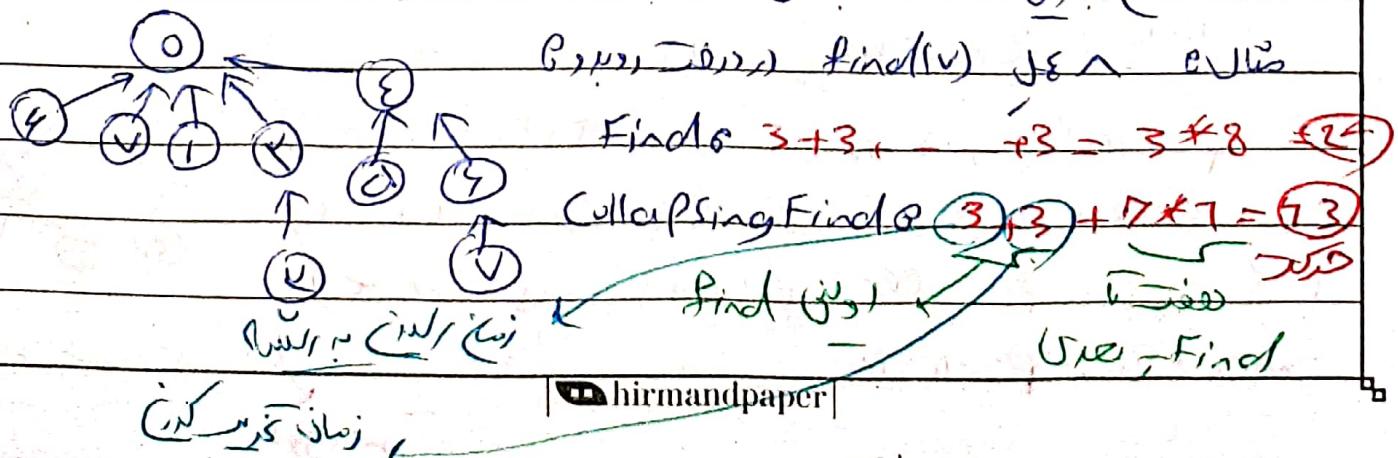
 lead = parent[trail];

 parent[trail] = root; lead = trail

 return root;

 O(h) \rightarrow انجام دہنے کا وقت

ایسے ہے اسے find کو کیا کرنا ہے اسے find(i) کا نتیجہ میں i کو j کا parent(j) کے لئے ہے



پیش (Pre) دقت دهی (دودوی)

ساده‌ترین صفات که داریم را خلاصه کنید

۱) بعد دقت دهی (دودوی) متنزه باشد

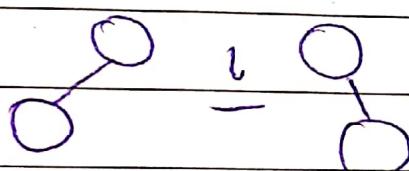
۲) هر دوی از اعماق آنرا که تولید شده است را در جای اصلی خود باشد

۳) دوی دقت دهی (دودوی) متنزه باشد

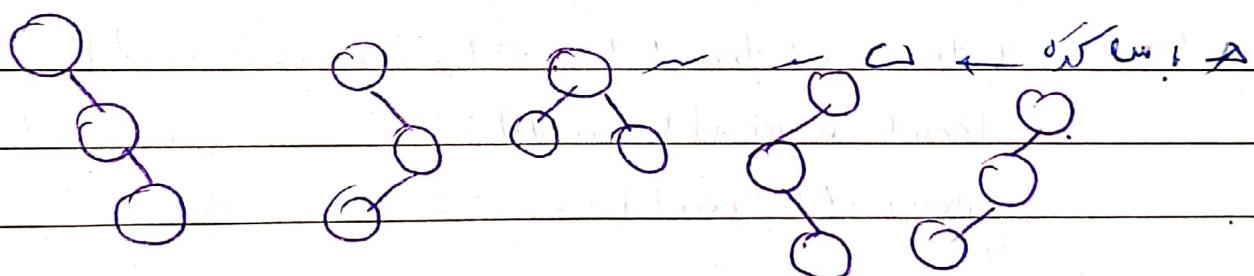
(دقت دهی (دودوی) متنزه)



با صفتی داشته باشند و دور دار



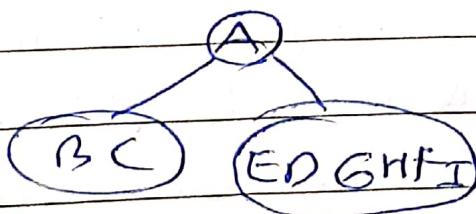
برای دوکردن از درست (درستی متنزه و دور دار)



میان (In) دقت (دودوی) از (بنای دهی) میان مرتب و پس ترسی

preorder A B C D E F G H I

inorder B C A D E F G H I

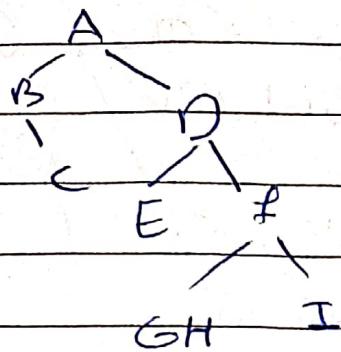


preorder A B C (D E F G H I)

inorder B C A (E D G H F I)

(پس ترسی preoder (پیش دهی دهی))

نیز پس ترسی



تعدد حادثتیت (مای) صیغه‌نگار ک با اسئال این را به داخل (ست) و خارج (ر) کام دوست (مای) چونکه به (د) صراحت مادر با تعداد (دوست-مای) (علوی) صیغه‌نگار باشد

pre
inorder

لذا خواسته شد که $b_1 = 1$, $b_2 = 2$, $b_3 = \omega$ باشد.

$$\text{M}_\cdot^* \text{M}_\cdot^* - \text{Mn}$$

$$n = 5 \quad (m_1 * m_2)$$

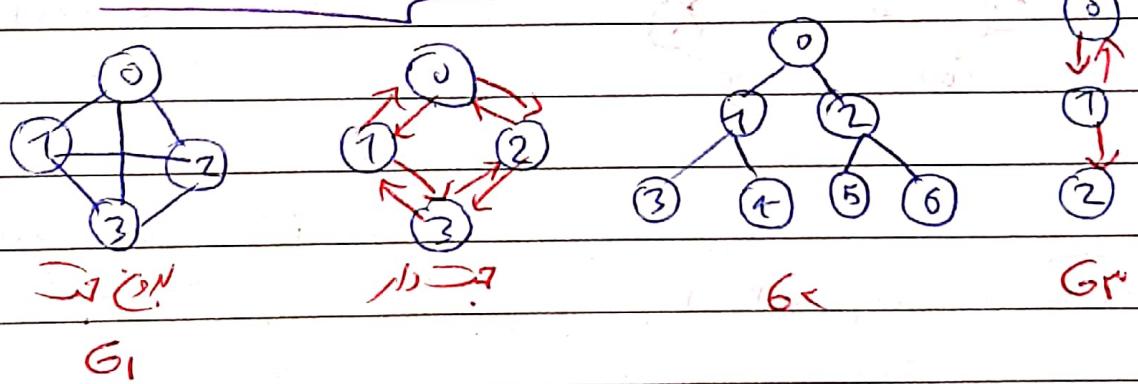
$$n = l \cdot (m_1 * m_2) * m_p \quad m_1 * (m_2 * m_p)$$

$$n = \varepsilon \cdot ((m_1 * m_2) * m_3) * m_4$$

$$(\min(m^*, \max(m^*, m^{\dagger})) + m^*)$$

تجدد حالي (جدي) كمصدر ثانٍ (أو مصادر ثالثة) را

$$((m_1 * m_2) * (m_3 * m_4)) \underset{\text{comm}}{=} ((m_1 * m_3) * (m_2 * m_4))$$



$$\checkmark(G_1) = \{0, 1, 2, 3\}$$

$$\checkmark(G_5) = \{0, 1, 2, 3, 4, 5\}$$

$$\checkmark(6\leftrightarrow)=\{0,1,5\}$$

$$E(G_1) = \{(0,1), (0,2), (0,4), (1,2), (1,3), (2,4)\}$$

$$E(G) = \{(0,1), (0,2), (1,3), (1,4), (2,3), (2,4)\}$$

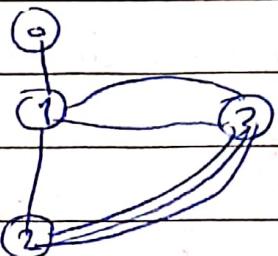
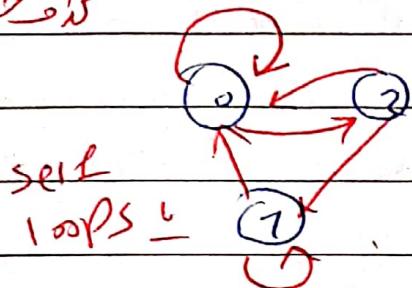
$$E(G^*) = \{ \langle 0,1 \rangle, \langle 1,0 \rangle, \langle 1,2 \rangle \}$$

مفع (Self-Contain Loop)

self loops

* دوست گرفتہ مزتف (یا کسی رسمی سہ خوبی) دللت. جسے یا لکھا کیا
باختہ ای صورت - کواد خوبی (ای صورت) بند کرنا.

* دوست گرفتہ مزتف (یا کسی لکھا کیا) حینہ بار ظاہر ہے باختہ ای صورت
کواد خوبی



feedback loops

اگر (L و M) میں یا لکھا کیوں نہیں تو

ایک جگہ ایک جگہ (L و M) میں یا لکھا کیوں نہیں تو



اگر (L و M) میں یا لکھا کیوں نہیں تو

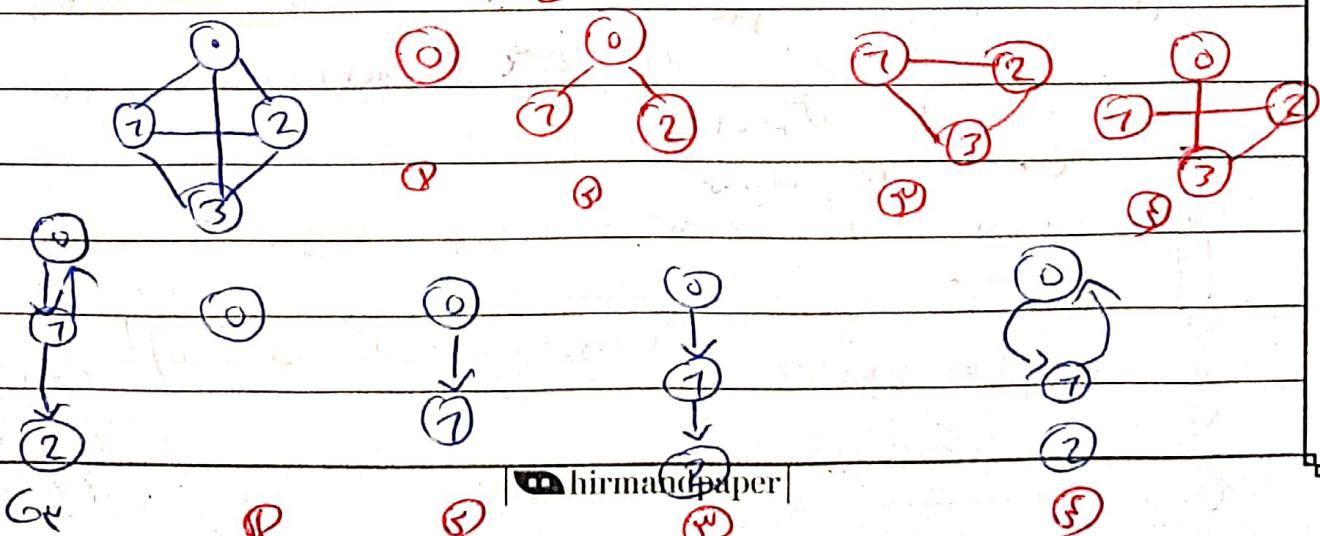
ایک جگہ ایک جگہ (L و M) میں یا لکھا کیوں نہیں تو

ایک جگہ ایک جگہ (L و M) میں یا لکھا کیوں نہیں تو

ایک جگہ ایک جگہ (L و M) میں یا لکھا کیوں نہیں تو

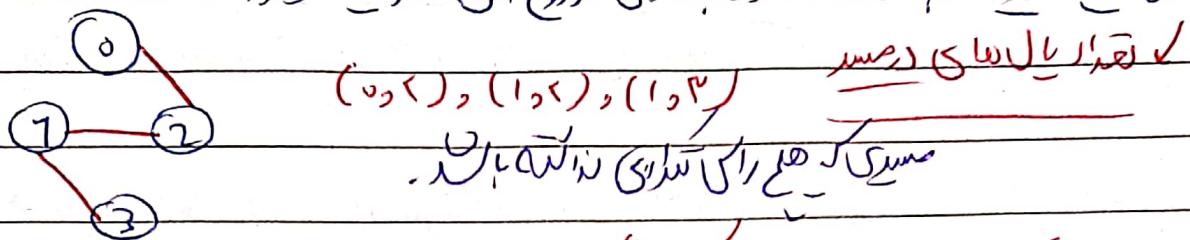
$V(G') \subseteq V(G)$ and $E(G') \subseteq E(G)$

یا لکھا کیوں نہیں تو



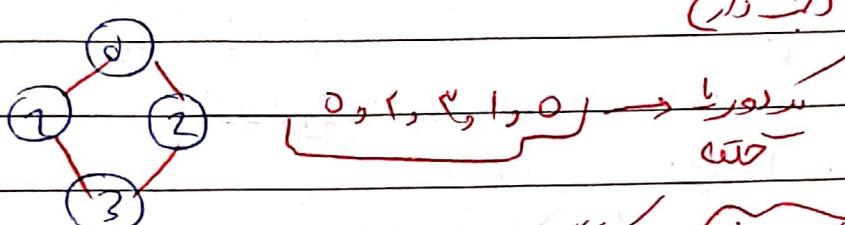
سونو (بینا) از پالک (نباتی از رود) که یار (استیل) (پالک)

حول مسار مدور - تعدد الزوايا (دفع آن تغيره) ص ٩٦



در مسنون که ایشان می‌کنند

مسنونات بحسب المعايير المعمولية



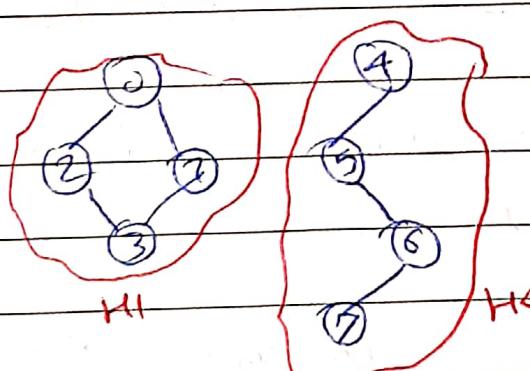
(۳) نکات مفید در گردشگری / عکس میراث گردشگری اکادمی

صـ (جـ 6) مـ (جـ 1) لـ (جـ 2) مـ (جـ 3) لـ (جـ 4) مـ (جـ 5) لـ (جـ 6)

کار و فن اکار سینمہ درج ہے

in Φ $\exists \exists y_1 \exists x_1 \exists y_2$ $\neg \exists x_2 \exists y_3 \exists x_3 \exists y_4$ Φ $\exists x_4 \exists y_5 \exists x_5 \exists y_6$

• $\exists i, j \in \mathbb{N}$ $i \neq j$ $\underline{\underline{z_i = z_j}}$



بِرَكَتِي زَيْلَدْو + H16H5

8 Geographie

النحو (المعنى) (المعنى)

544

مُوَكَّل (جَنِيدْ قَوْقَجْ) سَعْدُ الْأَسْدِ وَ مُوسَى الْأَسْدِ ازْمَانٌ عَنْهُمْ

ل) نزدیک ترین زندگانی که فرم معمولی است

نریس نہیں ہے

دیگر دوستی را که می‌خواهیم بخواهیم

از ای خانواده

خوب

دیگر دوستی را که می‌خواهیم بخواهیم



$$d_{0 \rightarrow 1} = 1$$

$$\sim d_{1 \rightarrow 2} = 2$$

میانگین درجه در گراف

$$e = \left(\sum_{i=0}^{n-1} d_i \right) / n$$

میانگین درجه در گراف

میانگین درجه در گراف

میانگین درجه در گراف

$$\begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

میانگین درجه در گراف

$$\begin{matrix} 0 & \xrightarrow{\quad} & [1/1] & \xrightarrow{\quad} & 1 & \xrightarrow{\quad} & 2 \\ 1 & \xrightarrow{\quad} & [2/1] & \xrightarrow{\quad} & & & \end{matrix}$$

میانگین درجه در گراف

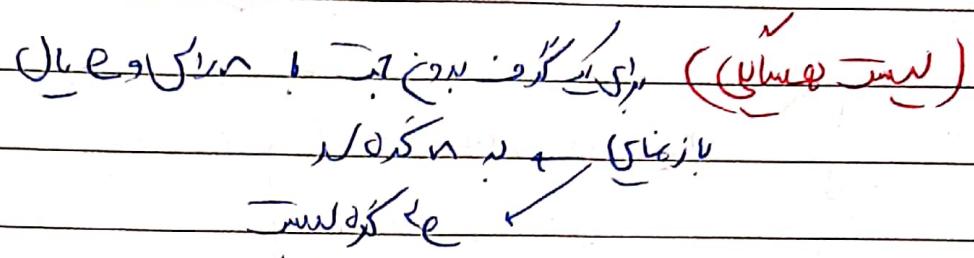
$$\sum_{j=0}^{n-1} \text{adj}_i \text{mat}[i][j]$$

میانگین درجه در گراف

$$\text{ind}(v_i) = \sum_{j=0}^{n-1} A[j][i] \quad \text{outd}(v_i) = \sum_{j=0}^{n-1} A[i][j]$$

لیست کنکسیو (Concise List) یا لیست مترادف (Adjacency List) یک روش برای نمایش گراف است که هر یکی از گرهات را به عنوان یک عضو از یک آرایه معرفی کرد و مقدار این عضو را با یک لیست مترادف (Adjacency List) معرفی کرد.

(n) ۰, ۱ =



typedef struct node *node_pointer;

typedef struct node {
 int vertex; → بارهای
 struct node *link; };

struct node *link;

y;

node_pointer graph[MAX_VERTICES];

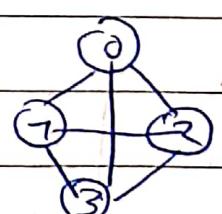
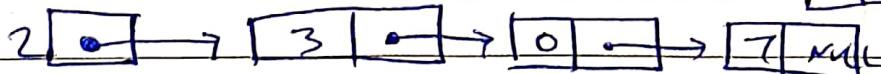
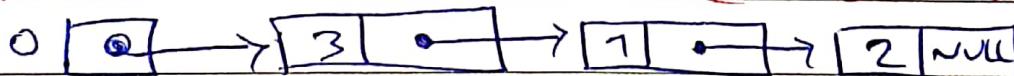
int n = ۰; گرفت

Head nodes

vertex links

(رسانی‌نوار) (رسانی‌نوار)

بازگشتی گرفت



۰

۱ نور

۲

۳

۴

۵

۶

۷

۸

۹

۱۰

۱۱

۱۲

۱۳

۱۴

۱۵

۱۶

۱۷

۱۸

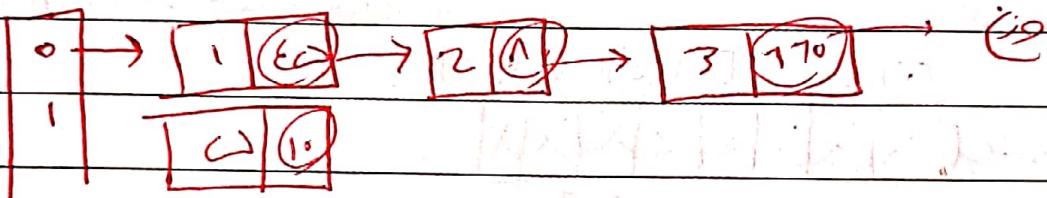
۱۹

گلسا کی قیمتی را در $\text{adj mat}[i][j]$ می بینیم

و این را ایجاد کریں

ایجاد ماتریس مجاہدین $\text{adj mat}[i][j]$ کی طرف

و این را ایجاد کریں



اول

(۱) (۲) (۳) (۴) (۵) (۶)

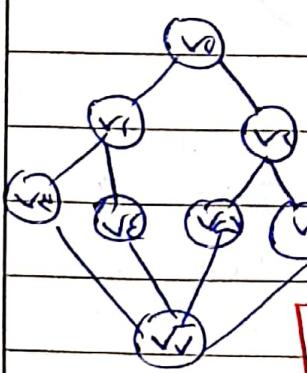
Depth first search (DFS)

preorder traversal

Breadth First search

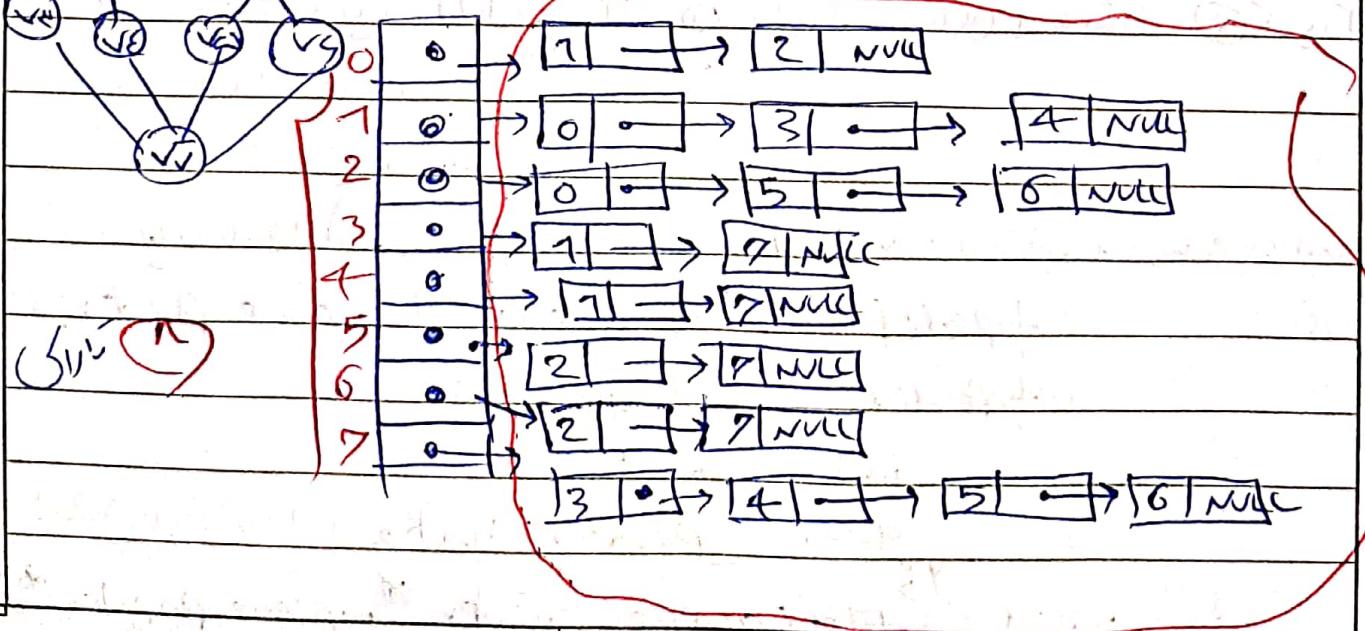
(۰, ۱, ۲)

levelorder traversal



DFS: V0, V1, V3, V4, V2, V5, V6, V0, V4

BFS: V0, V1, V2, V3, V4, V5, V6, V0



لے کر خانہ میں دشمنی کے ساتھ بیٹھنے کا انتہا تھا۔

```

void dfs(int v)           adjacency list (l)
{
    nod_pointer w;
    visited[v] = TRUE;   ~ ~ matrix (n)
    printf("%d,%d,%d);\n", v, l[v], n);
    for(w=graph[v]; w; w=w->link)  > dts
        if(!visited[w->vertex])      > w
            dfs(w->vertex);       > in
}
short int visited[MAX];

```

[.]	[i]	[j]	[e]	[ɛ]	[ə]	[ɔ]	[u]
visited	X	X	X	X	X	X	X

outputs 0 1 3 7 4 5 2 6 due, you

$O(n^2)$ ← مرور فیلپینی و متریک

(حراك) و فقط سار

(۱۰۷)

جسٹیس (دیکھ دین) (اولنگ) ۱ نومبر ۲۰۰۵ء کے بعد کسی کو

ادسے visited (سیوریز) (SIVORY) (سیوریز) (SIVORY)

typedef struct queue *queue_pointer;

typedef struct queue P

int vertex:

queue pointer links

```
void *alloc(generic pointer *); generic pointer *int, *  
int deallocate(generic pointer *);
```


adjacency list o(n+e)
matrix o(n²)

void connected (void) // determine the connected
int i;

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

if (!visited[i])

dfs(i);

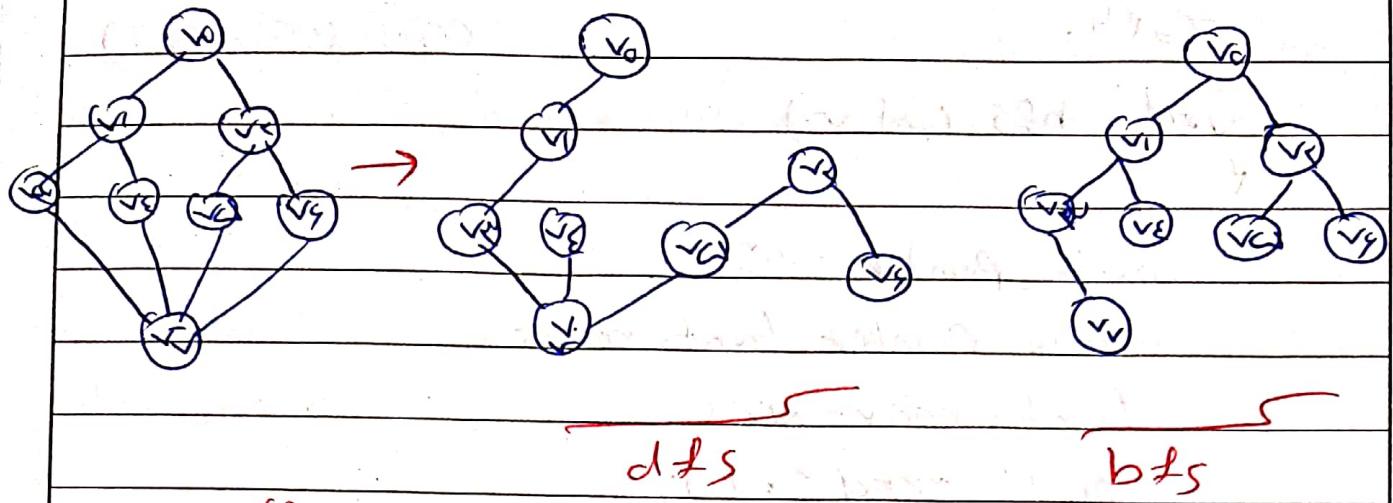
printf("\n");

Y

Y

لطفاً (V₁-i₁)
لطفاً (V₂-i₂)
لطفاً (V₃-i₃)
لطفاً (V₄-i₄)
لطفاً (V₅-i₅)
لطفاً (V₆-i₆)
لطفاً (V₇-i₇)
لطفاً (V₈-i₈)
لطفاً (V₉-i₉)
لطفاً (V₁₀-i₁₀)
لطفاً (V₁₁-i₁₁)
لطفاً (V₁₂-i₁₂)
لطفاً (V₁₃-i₁₃)
لطفاً (V₁₄-i₁₄)
لطفاً (V₁₅-i₁₅)
لطفاً (V₁₆-i₁₆)

لطفاً (V₁₇-i₁₇)
لطفاً (V₁₈-i₁₈)
لطفاً (V₁₉-i₁₉)
لطفاً (V₂₀-i₂₀)
لطفاً (V₂₁-i₂₁)
لطفاً (V₂₂-i₂₂)
لطفاً (V₂₃-i₂₃)
لطفاً (V₂₄-i₂₄)
لطفاً (V₂₅-i₂₅)
لطفاً (V₂₆-i₂₆)
لطفاً (V₂₇-i₂₇)
لطفاً (V₂₈-i₂₈)
لطفاً (V₂₉-i₂₉)
لطفاً (V₃₀-i₃₀)
لطفاً (V₃₁-i₃₁)
لطفاً (V₃₂-i₃₂)
لطفاً (V₃₃-i₃₃)
لطفاً (V₃₄-i₃₄)
لطفاً (V₃₅-i₃₅)
لطفاً (V₃₆-i₃₆)
لطفاً (V₃₇-i₃₇)
لطفاً (V₃₈-i₃₈)
لطفاً (V₃₉-i₃₉)
لطفاً (V₄₀-i₄₀)
لطفاً (V₄₁-i₄₁)
لطفاً (V₄₂-i₄₂)
لطفاً (V₄₃-i₄₃)
لطفاً (V₄₄-i₄₄)
لطفاً (V₄₅-i₄₅)
لطفاً (V₄₆-i₄₆)
لطفاً (V₄₇-i₄₇)
لطفاً (V₄₈-i₄₈)
لطفاً (V₄₉-i₄₉)
لطفاً (V₅₀-i₅₀)
لطفاً (V₅₁-i₅₁)
لطفاً (V₅₂-i₅₂)
لطفاً (V₅₃-i₅₃)
لطفاً (V₅₄-i₅₄)
لطفاً (V₅₅-i₅₅)
لطفاً (V₅₆-i₅₆)
لطفاً (V₅₇-i₅₇)
لطفاً (V₅₈-i₅₈)
لطفاً (V₅₉-i₅₉)
لطفاً (V₆₀-i₆₀)
لطفاً (V₆₁-i₆₁)
لطفاً (V₆₂-i₆₂)
لطفاً (V₆₃-i₆₃)
لطفاً (V₆₄-i₆₄)
لطفاً (V₆₅-i₆₅)
لطفاً (V₆₆-i₆₆)
لطفاً (V₆₇-i₆₇)
لطفاً (V₆₈-i₆₈)
لطفاً (V₆₉-i₆₉)
لطفاً (V₇₀-i₇₀)
لطفاً (V₇₁-i₇₁)
لطفاً (V₇₂-i₇₂)
لطفاً (V₇₃-i₇₃)
لطفاً (V₇₄-i₇₄)
لطفاً (V₇₅-i₇₅)
لطفاً (V₇₆-i₇₆)
لطفاً (V₇₇-i₇₇)
لطفاً (V₇₈-i₇₈)
لطفاً (V₇₉-i₇₉)
لطفاً (V₈₀-i₈₀)
لطفاً (V₈₁-i₈₁)
لطفاً (V₈₂-i₈₂)
لطفاً (V₈₃-i₈₃)
لطفاً (V₈₄-i₈₄)
لطفاً (V₈₅-i₈₅)
لطفاً (V₈₆-i₈₆)
لطفاً (V₈₇-i₈₇)
لطفاً (V₈₈-i₈₈)
لطفاً (V₈₉-i₈₉)
لطفاً (V₉₀-i₉₀)
لطفاً (V₉₁-i₉₁)
لطفاً (V₉₂-i₉₂)
لطفاً (V₉₃-i₉₃)
لطفاً (V₉₄-i₉₄)
لطفاً (V₉₅-i₉₅)
لطفاً (V₉₆-i₉₆)
لطفاً (V₉₇-i₉₇)
لطفاً (V₉₈-i₉₈)
لطفاً (V₉₉-i₉₉)
لطفاً (V₁₀₀-i₁₀₀)
لطفاً (V₁₀₁-i₁₀₁)
لطفاً (V₁₀₂-i₁₀₂)
لطفاً (V₁₀₃-i₁₀₃)
لطفاً (V₁₀₄-i₁₀₄)
لطفاً (V₁₀₅-i₁₀₅)
لطفاً (V₁₀₆-i₁₀₆)
لطفاً (V₁₀₇-i₁₀₇)
لطفاً (V₁₀₈-i₁₀₈)
لطفاً (V₁₀₉-i₁₀₉)
لطفاً (V₁₁₀-i₁₁₀)
لطفاً (V₁₁₁-i₁₁₁)
لطفاً (V₁₁₂-i₁₁₂)
لطفاً (V₁₁₃-i₁₁₃)
لطفاً (V₁₁₄-i₁₁₄)
لطفاً (V₁₁₅-i₁₁₅)
لطفاً (V₁₁₆-i₁₁₆)
لطفاً (V₁₁₇-i₁₁₇)
لطفاً (V₁₁₈-i₁₁₈)
لطفاً (V₁₁₉-i₁₁₉)
لطفاً (V₁₂₀-i₁₂₀)
لطفاً (V₁₂₁-i₁₂₁)
لطفاً (V₁₂₂-i₁₂₂)
لطفاً (V₁₂₃-i₁₂₃)
لطفاً (V₁₂₄-i₁₂₄)
لطفاً (V₁₂₅-i₁₂₅)
لطفاً (V₁₂₆-i₁₂₆)
لطفاً (V₁₂₇-i₁₂₇)
لطفاً (V₁₂₈-i₁₂₈)
لطفاً (V₁₂₉-i₁₂₉)
لطفاً (V₁₃₀-i₁₃₀)
لطفاً (V₁₃₁-i₁₃₁)
لطفاً (V₁₃₂-i₁₃₂)
لطفاً (V₁₃₃-i₁₃₃)
لطفاً (V₁₃₄-i₁₃₄)
لطفاً (V₁₃₅-i₁₃₅)
لطفاً (V₁₃₆-i₁₃₆)
لطفاً (V₁₃₇-i₁₃₇)
لطفاً (V₁₃₈-i₁₃₈)
لطفاً (V₁₃₉-i₁₃₉)
لطفاً (V₁₄₀-i₁₄₀)
لطفاً (V₁₄₁-i₁₄₁)
لطفاً (V₁₄₂-i₁₄₂)
لطفاً (V₁₄₃-i₁₄₃)
لطفاً (V₁₄₄-i₁₄₄)
لطفاً (V₁₄₅-i₁₄₅)
لطفاً (V₁₄₆-i₁₄₆)
لطفاً (V₁₄₇-i₁₄₇)
لطفاً (V₁₄₈-i₁₄₈)
لطفاً (V₁₄₉-i₁₄₉)
لطفاً (V₁₅₀-i₁₅₀)
لطفاً (V₁₅₁-i₁₅₁)
لطفاً (V₁₅₂-i₁₅₂)
لطفاً (V₁₅₃-i₁₅₃)
لطفاً (V₁₅₄-i₁₅₄)
لطفاً (V₁₅₅-i₁₅₅)
لطفاً (V₁₅₆-i₁₅₆)
لطفاً (V₁₅₇-i₁₅₇)
لطفاً (V₁₅₈-i₁₅₈)
لطفاً (V₁₅₉-i₁₅₉)
لطفاً (V₁₆₀-i₁₆₀)
لطفاً (V₁₆₁-i₁₆₁)
لطفاً (V₁₆₂-i₁₆₂)
لطفاً (V₁₆₃-i₁₆₃)
لطفاً (V₁₆₄-i₁₆₄)
لطفاً (V₁₆₅-i₁₆₅)
لطفاً (V₁₆₆-i₁₆₆)
لطفاً (V₁₆₇-i₁₆₇)
لطفاً (V₁₆₈-i₁₆₈)
لطفاً (V₁₆₉-i₁₆₉)
لطفاً (V₁₇₀-i₁₇₀)
لطفاً (V₁₇₁-i₁₇₁)
لطفاً (V₁₇₂-i₁₇₂)
لطفاً (V₁₇₃-i₁₇₃)
لطفاً (V₁₇₄-i₁₇₄)
لطفاً (V₁₇₅-i₁₇₅)
لطفاً (V₁₇₆-i₁₇₆)
لطفاً (V₁₇₇-i₁₇₇)
لطفاً (V₁₇₈-i₁₇₈)
لطفاً (V₁₇₉-i₁₇₉)
لطفاً (V₁₈₀-i₁₈₀)
لطفاً (V₁₈₁-i₁₈₁)
لطفاً (V₁₈₂-i₁₈₂)
لطفاً (V₁₈₃-i₁₈₃)
لطفاً (V₁₈₄-i₁₈₄)
لطفاً (V₁₈₅-i₁₈₅)
لطفاً (V₁₈₆-i₁₈₆)
لطفاً (V₁₈₇-i₁₈₇)
لطفاً (V₁₈₈-i₁₈₈)
لطفاً (V₁₈₉-i₁₈₉)
لطفاً (V₁₉₀-i₁₉₀)
لطفاً (V₁₉₁-i₁₉₁)
لطفاً (V₁₉₂-i₁₉₂)
لطفاً (V₁₉₃-i₁₉₃)
لطفاً (V₁₉₄-i₁₉₄)
لطفاً (V₁₉₅-i₁₉₅)
لطفاً (V₁₉₆-i₁₉₆)
لطفاً (V₁₉₇-i₁₉₇)
لطفاً (V₁₉₈-i₁₉₈)
لطفاً (V₁₉₉-i₁₉₉)
لطفاً (V₂₀₀-i₂₀₀)
لطفاً (V₂₀₁-i₂₀₁)
لطفاً (V₂₀₂-i₂₀₂)
لطفاً (V₂₀₃-i₂₀₃)
لطفاً (V₂₀₄-i₂₀₄)
لطفاً (V₂₀₅-i₂₀₅)
لطفاً (V₂₀₆-i₂₀₆)
لطفاً (V₂₀₇-i₂₀₇)
لطفاً (V₂₀₈-i₂₀₈)
لطفاً (V₂₀₉-i₂₀₉)
لطفاً (V₂₁₀-i₂₁₀)
لطفاً (V₂₁₁-i₂₁₁)
لطفاً (V₂₁₂-i₂₁₂)
لطفاً (V₂₁₃-i₂₁₃)
لطفاً (V₂₁₄-i₂₁₄)
لطفاً (V₂₁₅-i₂₁₅)
لطفاً (V₂₁₆-i₂₁₆)
لطفاً (V₂₁₇-i₂₁₇)
لطفاً (V₂₁₈-i₂₁₈)
لطفاً (V₂₁₉-i₂₁₉)
لطفاً (V₂₂₀-i₂₂₀)
لطفاً (V₂₂₁-i₂₂₁)
لطفاً (V₂₂₂-i₂₂₂)
لطفاً (V₂₂₃-i₂₂₃)
لطفاً (V₂₂₄-i₂₂₄)
لطفاً (V₂₂₅-i₂₂₅)
لطفاً (V₂₂₆-i₂₂₆)
لطفاً (V₂₂₇-i₂₂₇)
لطفاً (V₂₂₈-i₂₂₈)
لطفاً (V₂₂₉-i₂₂₉)
لطفاً (V₂₃₀-i₂₃₀)
لطفاً (V₂₃₁-i₂₃₁)
لطفاً (V₂₃₂-i₂₃₂)
لطفاً (V₂₃₃-i₂₃₃)
لطفاً (V₂₃₄-i₂₃₄)
لطفاً (V₂₃₅-i₂₃₅)
لطفاً (V₂₃₆-i₂₃₆)
لطفاً (V₂₃₇-i₂₃₇)
لطفاً (V₂₃₈-i₂₃₈)
لطفاً (V₂₃₉-i₂₃₉)
لطفاً (V₂₄₀-i₂₄₀)
لطفاً (V₂₄₁-i₂₄₁)
لطفاً (V₂₄₂-i₂₄₂)
لطفاً (V₂₄₃-i₂₄₃)
لطفاً (V₂₄₄-i₂₄₄)
لطفاً (V₂₄₅-i₂₄₅)
لطفاً (V₂₄₆-i₂₄₆)
لطفاً (V₂₄₇-i₂₄₇)
لطفاً (V₂₄₈-i₂₄₈)
لطفاً (V₂₄₉-i₂₄₉)
لطفاً (V₂₅₀-i₂₅₀)
لطفاً (V₂₅₁-i₂₅₁)
لطفاً (V₂₅₂-i₂₅₂)
لطفاً (V₂₅₃-i₂₅₃)
لطفاً (V₂₅₄-i₂₅₄)
لطفاً (V₂₅₅-i₂₅₅)
لطفاً (V₂₅₆-i₂₅₆)
لطفاً (V₂₅₇-i₂₅₇)
لطفاً (V₂₅₈-i₂₅₈)
لطفاً (V₂₅₉-i₂₅₉)
لطفاً (V₂₆₀-i₂₆₀)
لطفاً (V₂₆₁-i₂₆₁)
لطفاً (V₂₆₂-i₂₆₂)
لطفاً (V₂₆₃-i₂₆₃)
لطفاً (V₂₆₄-i₂₆₄)
لطفاً (V₂₆₅-i₂₆₅)
لطفاً (V₂₆₆-i₂₆₆)
لطفاً (V₂₆₇-i₂₆₇)
لطفاً (V₂₆₈-i₂₆₈)
لطفاً (V₂₆₉-i₂₆₉)
لطفاً (V₂₇₀-i₂₇₀)
لطفاً (V₂₇₁-i₂₇₁)
لطفاً (V₂₇₂-i₂₇₂)
لطفاً (V₂₇₃-i₂₇₃)
لطفاً (V₂₇₄-i₂₇₄)
لطفاً (V₂₇₅-i₂₇₅)
لطفاً (V₂₇₆-i₂₇₆)
لطفاً (V₂₇₇-i₂₇₇)
لطفاً (V₂₇₈-i₂₇₈)
لطفاً (V₂₇₉-i₂₇₉)
لطفاً (V₂₈₀-i₂₈₀)
لطفاً (V₂₈₁-i₂₈₁)
لطفاً (V₂₈₂-i₂₈₂)
لطفاً (V₂₈₃-i₂₈₃)
لطفاً (V₂₈₄-i₂₈₄)
لطفاً (V₂₈₅-i₂₈₅)
لطفاً (V₂₈₆-i₂₈₆)
لطفاً (V₂₈₇-i₂₈₇)
لطفاً (V₂₈₈-i₂₈₈)
لطفاً (V₂₈₉-i₂₈₉)
لطفاً (V₂₉₀-i₂₉₀)
لطفاً (V₂₉₁-i₂₉₁)
لطفاً (V₂₉₂-i₂₉₂)
لطفاً (V₂₉₃-i₂₉₃)
لطفاً (V₂₉₄-i₂₉₄)
لطفاً (V₂₉₅-i₂₉₅)
لطفاً (V₂₉₆-i₂₉₆)
لطفاً (V₂₉₇-i₂₉₇)
لطفاً (V₂₉₈-i₂₉₈)
لطفاً (V₂₉₉-i₂₉₉)
لطفاً (V₃₀₀-i₃₀₀)
لطفاً (V₃₀₁-i₃₀₁)
لطفاً (V₃₀₂-i₃₀₂)
لطفاً (V₃₀₃-i₃₀₃)
لطفاً (V₃₀₄-i₃₀₄)
لطفاً (V₃₀₅-i₃₀₅)
لطفاً (V₃₀₆-i₃₀₆)
لطفاً (V₃₀₇-i₃₀₇)
لطفاً (V₃₀₈-i₃₀₈)
لطفاً (V₃₀₉-i₃₀₉)
لطفاً (V₃₁₀-i₃₁₀)
لطفاً (V₃₁₁-i₃₁₁)
لطفاً (V₃₁₂-i₃₁₂)
لطفاً (V₃₁₃-i₃₁₃)
لطفاً (V₃₁₄-i₃₁₄)
لطفاً (V₃₁₅-i₃₁₅)
لطفاً (V₃₁₆-i₃₁₆)
لطفاً (V₃₁₇-i₃₁₇)
لطفاً (V₃₁₈-i₃₁₈)
لطفاً (V₃₁₉-i₃₁₉)
لطفاً (V₃₂₀-i₃₂₀)
لطفاً (V₃₂₁-i₃₂₁)
لطفاً (V₃₂₂-i₃₂₂)
لطفاً (V₃₂₃-i₃₂₃)
لطفاً (V₃₂₄-i₃₂₄)
لطفاً (V₃₂₅-i₃₂₅)
لطفاً (V₃₂₆-i₃₂₆)
لطفاً (V₃₂₇-i₃₂₇)
لطفاً (V₃₂₈-i₃₂₈)
لطفاً (V₃₂₉-i₃₂₉)
لطفاً (V₃₃₀-i₃₃₀)
لطفاً (V₃₃₁-i₃₃₁)
لطفاً (V₃₃₂-i₃₃₂)
لطفاً (V₃₃₃-i₃₃₃)
لطفاً (V₃₃₄-i₃₃₄)
لطفاً (V₃₃₅-i₃₃₅)
لطفاً (V₃₃₆-i₃₃₆)
لطفاً (V₃₃₇-i₃₃₇)
لطفاً (V₃₃₈-i₃₃₈)
لطفاً (V₃₃₉-i₃₃₉)
لطفاً (V₃₄₀-i₃₄₀)
لطفاً (V₃₄₁-i₃₄₁)
لطفاً (V₃₄₂-i₃₄₂)
لطفاً (V₃₄₃-i₃₄₃)
لطفاً (V₃₄₄-i₃₄₄)
لطفاً (V₃₄₅-i₃₄₅)
لطفاً (V₃₄₆-i₃₄₆)
لطفاً (V₃₄₇-i₃₄₇)
لطفاً (V₃₄₈-i₃₄₈)
لطفاً (V₃₄₉-i₃₄₉)
لطفاً (V₃₅₀-i₃₅₀)
لطفاً (V₃₅₁-i₃₅₁)
لطفاً (V₃₅₂-i₃₅₂)
لطفاً (V₃₅₃-i₃₅₃)
لطفاً (V₃₅₄-i₃₅₄)
لطفاً (V₃₅₅-i₃₅₅)
لطفاً (V₃₅₆-i₃₅₆)
لطفاً (V₃₅₇-i₃₅₇)
لطفاً (V₃₅₈-i₃₅₈)
لطفاً (V₃₅₉-i₃₅₉)
لطفاً (V₃₆₀-i₃₆₀)
لطفاً (V₃₆₁-i₃₆₁)
لطفاً (V₃₆₂-i₃₆₂)
لطفاً (V₃₆₃-i₃₆₃)
لطفاً (V₃₆₄-i₃₆₄)
لطفاً (V₃₆₅-i₃₆₅)
لطفاً (V₃₆₆-i₃₆₆)
لطفاً (V₃₆₇-i₃₆₇)
لطفاً (V₃₆₈-i₃₆₈)
لطفاً (V₃₆₉-i₃₆₉)
لطفاً (V₃₇₀-i₃₇₀)
لطفاً (V₃₇₁-i₃₇₁)
لطفاً (V₃₇₂-i₃₇₂)
لطفاً (V₃₇₃-i₃₇₃)
لطفاً (V₃₇₄-i₃₇₄)
لطفاً (V₃₇₅-i₃₇₅)
لطفاً (V₃₇₆-i₃₇₆)
لطفاً (V₃₇₇-i₃₇₇)
لطفاً (V₃₇₈-i₃₇₈)
لطفاً (V₃₇₉-i₃₇₉)
لطفاً (V₃₈₀-i₃₈₀)
لطفاً (V₃₈₁-i₃₈₁)
لطفاً (V₃₈₂-i₃₈₂)
لطفاً (V₃₈₃-i₃₈₃)
لطفاً (V₃₈₄-i₃₈₄)
لطفاً (V₃₈₅-i₃₈₅)
لطفاً (V₃₈₆-i₃₈₆)
لطفاً (V₃₈₇-i₃₈₇)
لطفاً (V₃₈₈-i₃₈₈)
لطفاً (V₃₈₉-i₃₈₉)
لطفاً (V₃₉₀-i₃₉₀)
لطفاً (V₃₉₁-i₃₉₁)
لطفاً (V₃₉₂-i₃₉₂)
لطفاً (V₃₉₃-i₃₉₃)
لطفاً (V_{394</sub}



dfs

bfs

دیپث فارسی

void dfs(int v)

T = ٢٤

node pointer w; *ویکنگ ویکنگ*

visited[v] = TRUE;

printf("%d ", v);

for (w=graph[v]; w!=NULL; w=w->link)

if (!visited[w->vertex])

dfs(w); *دیپث فارسی*

end if

dfs

end loop

dfs(w->vertex);

T = T+1(v,w);

v,w

(parent, child) *پدر و فرزند*

$T = \emptyset;$

((جزء (S₂, T₂))

void bfs (int v)

{

node_pointer w;

queue_pointer front, rear;

front = rear = NULL;

printf ("%.5d ", v);

visited[v] = TRUE;

addq (&front, &rear, v);

while (front)

v = deleteq (&front);

for (w = graph[v]; w; w = w->link)

if (!visited[w->vertex])

printf ("%.5d ", w->vertex);

addq (&front, &rear, w->vertex);

visited[w->vertex] = TRUE;

$T = T \cup \{v, w\}$

نیکنی (اجراهی تعدادی)

}

$v(G) = \sqrt{\text{حدیقی} G} + \text{حدیقی} G$

$v(G)$

گوشه هایی که در اینجا نموداری نشان داده شده اند

جواب داده شده است و اینجا نموداری نشان داده شده است

جواب داده شده است

که می بینید اینجا نموداری نشان داده شده است

اکسل

(عمر = اوری کوئٹھا)

گراف لینہ دوسری (Biconnected) و نیکے چوری (articulation points)

خرفتی ہے گراف میں جو فونڈیل اے اگر فونڈیل اے تو ریکارڈ 6 نیکے چوری اے (Superior)

چوری اے وہ چالہی کہ باہم تکمیل کرنے والے ریکارڈ 6 نیکے چوری اے (وصولیہ)

گراف کو دوسری طرفہ کے چونقٹی چوری کے نتالیہ اے

وصولیہ چونقٹی طرفہ اے اگر فونڈیل 6 نیکے چوری کے نتالیہ اے

از 6 صریح اے

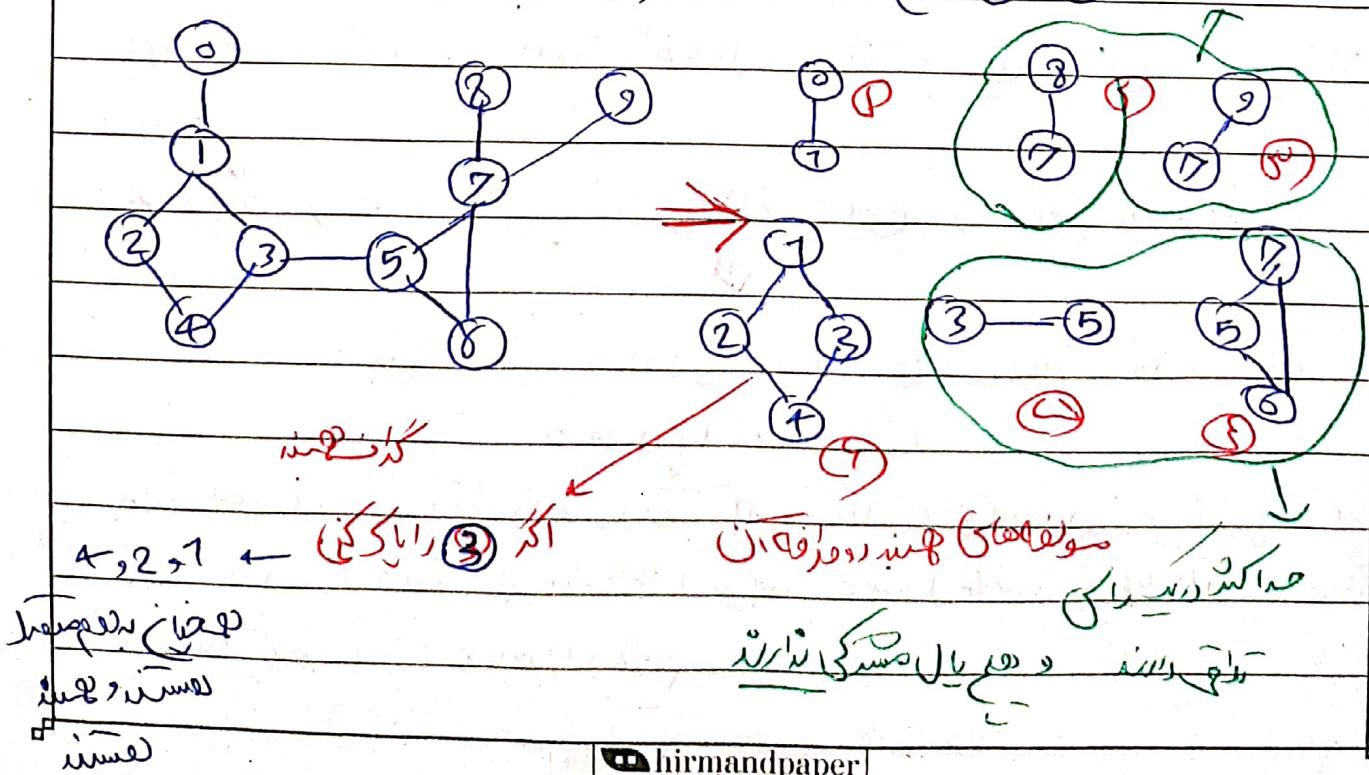
نیکسٹنی ہے 6 حادی چونقٹی کے نتالیہ کو فونڈیل کوں دوسری طرفہ بولنے والا مل ہے

وصولیہ چونقٹی طرفہ از 6 گراف، حالہ مردہ انتہا رکھوں

(اکریڈیٹ)

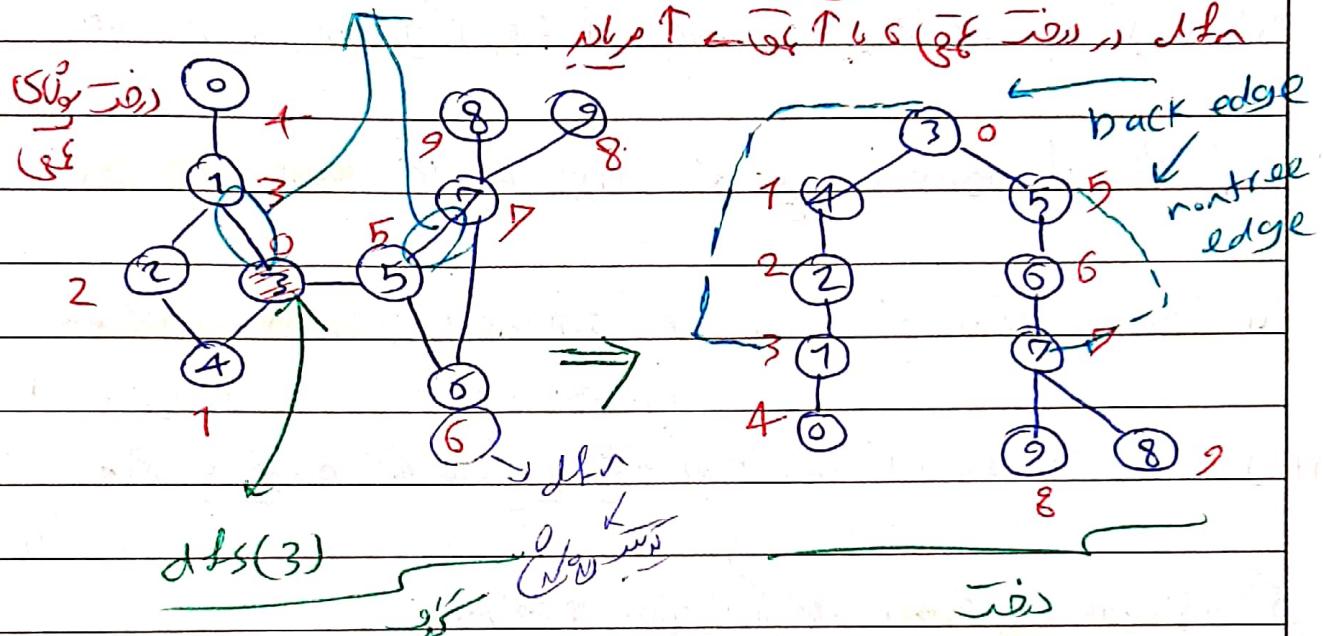
وصولیہ چونقٹی طرفہ

ہے اکی نیکان دوسرے سسک دوہا حصہ چونقٹی
(در 7 تلاعہ ایں) دوسری قرار دالتا ہے



(سکریپت مونیشن) قبضہ درجہ (بالفارسی) میرزا کوہلی بیانیہ (میرزا کوہلی)

دیپھرست نمبر (DFN) e depth first number



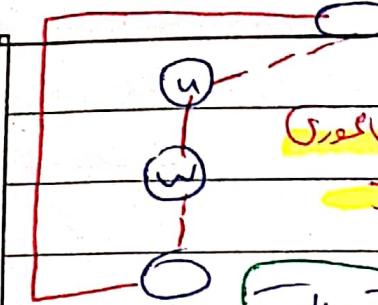
رسیو (Rx) و سیگنال (Signal) را در اینجا معرفی کرد.

مَنْ هُوَ كَفِيلٌ فَرَغَانَ نَبَتْ بِرْدَفَرْدَ بِلَادِ عَجَّيْ أَسْنَ يَا لِمَعَافِعِ دَالِلَهِ أَرْدَ

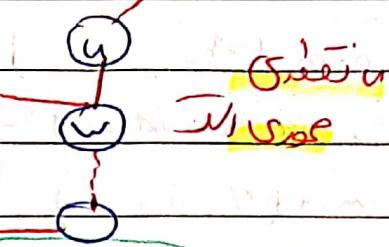
رسیه (РФ) پولی (پولی) هی (نفیا) کوئی ال - اکر و فکا (АКР
حلف (حلف) روسیه (Россія)

وہیں کامنے والے سنتھی (جھوپڑی) اور آگ و ٹھاکر (ڈاری) حداں میں قدرتیہ مانندے سارے بُلڈ برکوں کے قاریتالیم بالعقار اُنہیں صیغہ کے ساتھ تھاں و سُلھاکاں، حداں پر بُلڈ برکسی اسے کہا جاتا ہے۔

از درخت کمترین عرضه ای از u به w



نفعی معرفی
نمایش



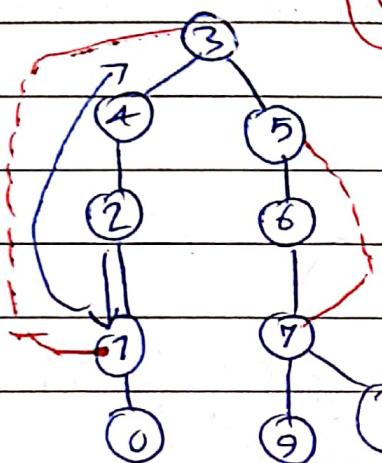
نفعی معرفی
نمایش

دستوراتی که در w ایجاد شده اند $low(w)$ و $dfn(w)$
کسر کوچک است (نیز با انتشاره از w صدای از سرمه و های راه را کم کرده اند) بدلیل اینکه w از u پیش از u میگذرد

$$low(w) < dfn(u)$$

$$low(w) \geq dfn(u)$$

u نیز w



vertex	0	1	2	3	4	5	6	7	8	9
dfn	0	1	2	3	4	5	6	7	8	9
low	0	0	0	3	4	5	6	7	8	9

نفعی معرفی (نیز کمترین عرضه ای از u به w)

۱) در این مسیر دستیاری بوده و در این دستیاری فرزند مانند w است

۲) $low(w) > dfn(u)$ در این دستیاری فرزند مانند w است

حال

برای راس w که بعدها u باشد $low(w) = low(w)$

برای راس v که بعدها u باشد $low(v) = low(v)$

$$low(v) = 0 \rightarrow v < 1 \rightarrow \text{نمایش معرفی}$$

$$dfn(v) = 1 \rightarrow v < 1 \rightarrow \text{نمایش معرفی}$$

نمایش معرفی $\rightarrow low(0) > dfn(1) + 1 \geq 1$

$\Sigma > \Sigma$

$low(w) = \min\{dfn(w), \min\{low(v) | v \text{ is a child of } w\}\}$

$\min\{dfn(v) | (w, v)\}$ is a back edge

$$\text{low}(0) = \min(4, \dots) = 4$$

$$\text{low}(1) = \min(3, 4, 0) = 0$$

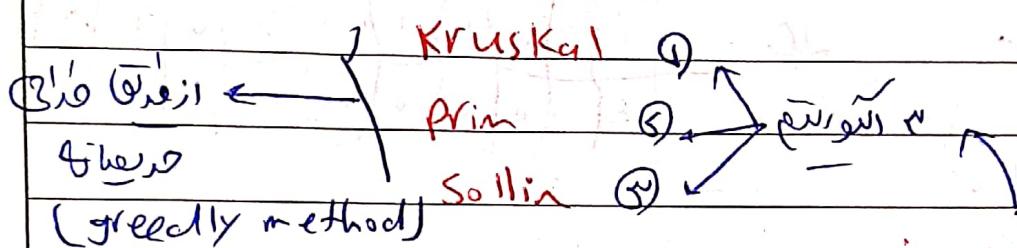
$$\text{low}(2) = \min(2, 0, \dots) = 0$$

$$\text{low}(4) = \min(1, 0, \dots) = 0 \rightarrow \text{کرس} \rightarrow \text{Kruskal}$$

\Rightarrow جزوی از low

$$\text{low}(9) = \min(\dots, \dots) = \dots$$

$$\text{low}(c) = \min(c, 0, \dots) = c$$



۱) نمایند از این راهی گرفتاره کنم

۲) بعد دستورات را امالي رفته کنم

۳) از این راهی که در ایجاد رکته بناد رفته باشد

(Kruskal)

۱) (درخت بُونا با کمترین هزینه را با اضافه کردن) میل در هر صورت مجاز است

۲) ایجادی اضافه شوند به تعبیر سایر ترکی هزینه ای ایند برقرار

۳) میل به اضافه صورت میشود به اینه که قبل از آن تیکلار در تیکل از اینه داشت

تیکل نداشت

۴) گرفتار شوند و ... تیکل ای اضافه شوند

۵) همچنان که گرفتار شوند تیکل ای اضافه شوند

Time complexity is $O(E \log E)$

$T = \emptyset$

Kruskal كروسكال

while (T contains less than $n-1$ edges & E is not empty)

choose a least cost edge (v, w) from E ;

delete (v, w) from E ; ~~if it is in T~~

if (v, w) does not create a cycle in T

add (v, w) ;

else

discard (v, w) .

y

if (T contains fewer than $n-1$ edges)

print ("No spanning tree in ");

مختار (edge) (v, w) مختار (v, w) 1 111

كرر (v, w) في المين هيلپ مختار (v, w)

o(e) \leftarrow heap (v, w)

o(log e) \leftarrow heap (v, w)

$T \leftarrow T \cup (v, w)$

و 111 ت $\leftarrow T \cup (v, w) \leftarrow$ 111

فيما $T \cup (v, w)$ \leftarrow اضافي \leftarrow ③

وي $T \cup (v, w)$ \leftarrow ④

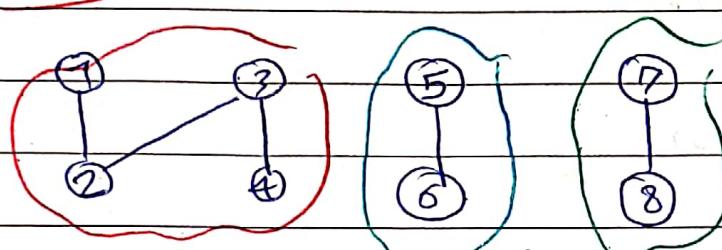
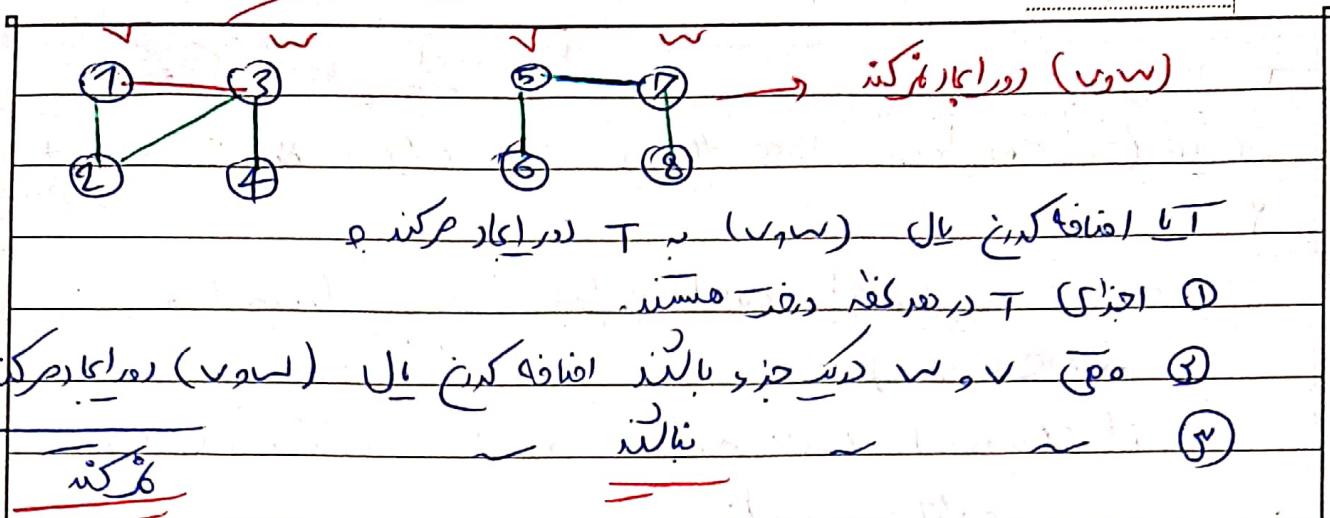
و $T \cup (v, w)$ \leftarrow ⑤

o(1) \leftarrow 1 111 \leftarrow 1 111 \leftarrow 1 111 \leftarrow 1 111 \leftarrow ①

وي $T \cup (v, w)$ \leftarrow ②

و $T \cup (v, w)$ \leftarrow ③

(در ایجاد مکن)



۱۶ حدود ت بارگویی (معنی آن صفحه) مرکز

۱۷ حدود را بمحض (لیس) آن نشان مراد

۱۸، ۲، ۳، ۴، ۵، ۶، ۷، ۸

۱۹ هنوز که ل (سر) به ت اضافه می‌وره و جنبی کامل شود بود اما باهم ترکس صفحه تا سه جزء را نشان اند.

۲۰ حداچی و صفحه های برای بزرگی اجزای ت بارگذاری می‌شوند و این این اجتماع مرکز

$$17, 2, 3, 4, 5 + 15, 6, 7 \Rightarrow 17, 2, 3, 4, 5, 6, 7$$

۲۱ ۱ ۳ ۵ ۷
۲ ۴ ۶ ۸

۲۲ ۱ ۳ ۵ ۷

۲۳ ۱ ۳ ۵ ۷ ۹ ۱۱ ۱۳ ۱۵ ۱۷ ۱۹ ۲۱ ۲۳ ۲۵ ۲۷ ۲۹ ۳۱ ۳۳ ۳۵ ۳۷ ۳۹ ۴۱ ۴۳ ۴۵ ۴۷ ۴۹ ۵۱ ۵۳ ۵۵ ۵۷ ۵۹ ۶۱ ۶۳ ۶۵ ۶۷ ۶۹ ۷۱ ۷۳ ۷۵ ۷۷ ۷۹ ۸۱ ۸۳ ۸۵ ۸۷ ۸۹ ۹۱ ۹۳ ۹۵ ۹۷ ۹۹

۲۴ اضافه کردن یا ل (سر) به ت در ایجاد مکن اکنون که اتفاق

$\text{if } (s_1 \neq s_2) \text{ Union } (s_1, s_2);$

$s_1 = \text{Find}(v);$

$s_2 = \text{Find}(w);$

النهاية انترواج مع بدل (داله) $O(n)$

$O(n)$ او $O(n \log n)$

$O(n + e)$ خلا ترس $\Theta(n + e)$, find vertex into

min heap (داله) $O(n \log n)$

$O(e \log e)$

$O(n + e \log e)$ Kruskal $\Theta(n \log n)$

(Prim) التوريم

1) خلا بدل (داله) هر دوی، با اضافه کردن به دل (داله) هر دوی

5) دل (داله) هر دوی اضافه (داله) به ت به دل (داله) هر دوی (داله) هر دوی

3) در عالم صراحی التوريم مجموع (داله) هر دوی (داله) هر دوی

برای اینکه در جمله Kruskal $\Theta(n \log n)$ است

پس در کل کمتر کم $\Theta(n \log n)$

برای درست باشد از مدل اضافه کردن کمترین وزنی دل (داله) هر دوی

5) $\Theta(n^2)$

6) $\Theta(n^2)$ اضافه کردن دل (داله) هر دوی

Prim's algorithm

(Prim) $\Theta(n^2)$

$T = \emptyset$;

$T_v = \emptyset$; /* start with vertex v_0 and no edges */

while (T contains fewer than $n - 1$ edges) /*

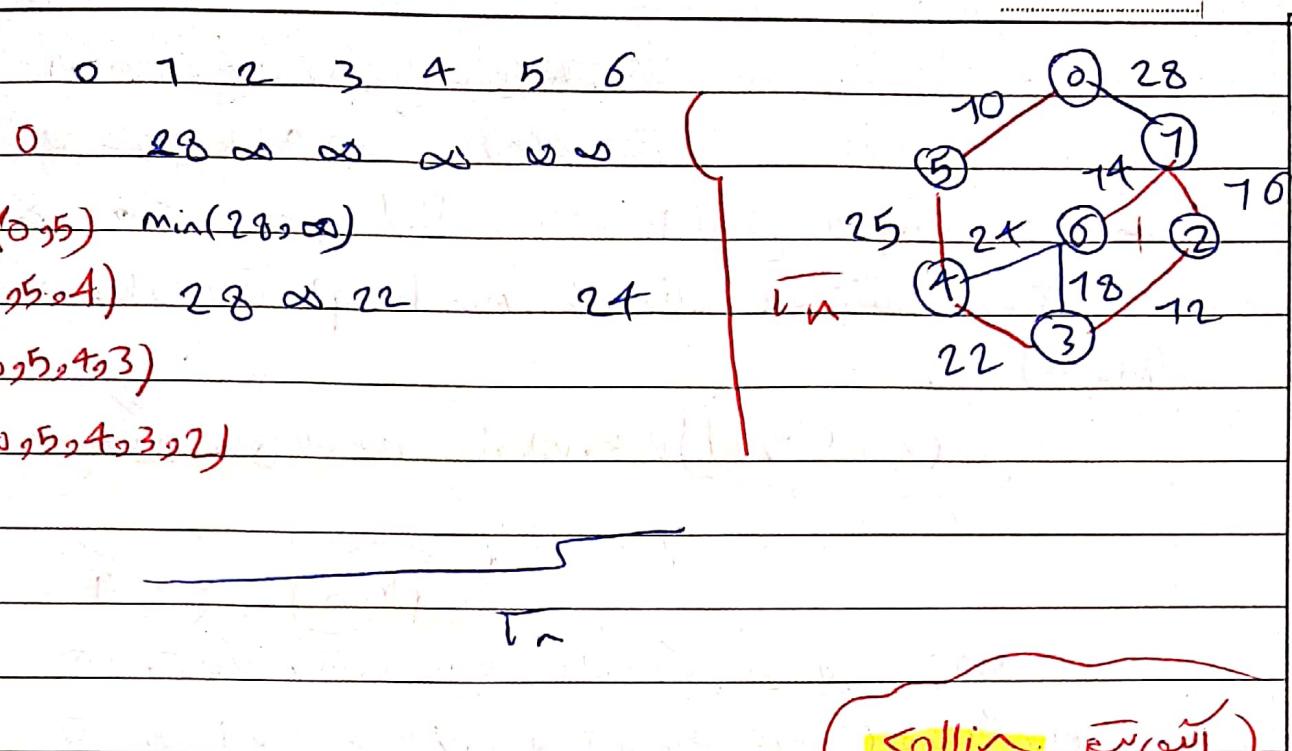
$O(n^2)$ let (u, v) be a least cost edge such that $u \in T_v$ and $v \notin T_v$.

if (there is no such edge) break;

add v to T_v ;

y add (u, v) to T ;

if (T contains fewer than $n - 1$ edges) \rightarrow handpaper
print ("No spanning tree")



Sollin (الخوارزمية)

- ① در مرحله اول مجموعه ای با یک عضو انتخاب شود.
- ② در مرحله دوم با کمترین مجموعه ای از آن صادف شوند و آن را از مجموعه حذف کنند.
- ③ در مرحله سوم مجموعه ای با یک عضو انتخاب شود که با مجموعه هایی که حذف شده اند بزرگتر باشد.
- ④ این انتخاب را تکرار کردن تا در مرحله ای که مجموعه ای با یک عضو باشد اتمام یابد.

Stage 18 $(0,5), (1,6), (2,3), (3,2), (4,3), (5,0), (6,7)$
 $\Rightarrow \{ (0,5) \}, \{ (1,6) \}, \{ (2,3) \}, \{ (3,2) \}, \{ (4,3) \}, \{ (5,0) \}, \{ (6,7) \}$
 در مرحله اول رید کردن

Stage 28 $\{ (0,5), (5,4), (7,6) \}, (7,2) \},$
 $\{ (2,3) \}, (4,3), (1,2) \}$

result $\{ (0,5), (5,4), (7,6), (1,2), (2,3), (4,3) \}$

پس از کسر خواهد بود که مجموعه هایی که حذف شده اند.

۱) فرد دیگری را انتخاب کردن که در این مرحله مجموعه ای باشد که از مجموعه هایی که حذف شده اند بزرگتر باشد.

۲) این مجموعه را از مجموعه هایی که حذف شده اند حذف کنید.

(٣) انتخاب های ساری خواهند بود و فقط از آنها باقی ماند.

(٤) همان که گفته بود میتوانیم با استفاده از این ایده دور دار

الْأَوْلَى \star Prim's ($\Theta(n^2)$) (برع تری روش)

الآن با الگوریتم Djikstra باید کوچکترین مسافت را با انتقالات مینیمیز کنیم.

Fibonacci heap با انتقالات O($n \log n$)

Union-Find باید از Kruskal از دست داشت $\Theta(n^2)$ *
* مقدار کمترین ممکن است این را با این ترتیب کند $O(n + \alpha(n))$

صربیل (Bubble Sort)

لیست از کوچکترین (R₀, R₁, ..., R_{n-1}) صربیل

او را که کوچکترین ممکن است از اینجا کشیده و آنرا کمترین ممکن کرده و آنرا در لیست اضافه کرد.

اگر R_i = R_j و R_{i+1} صربیل کند این را که جای خود را کرده و این را در لیست اضافه کرد.

روایتی برای صربیل کننده این که بقدر کاخ کوچک شد که حافظه ای ام از اینجا

صربیل (Bubble Sort) صربیل از اینجا آغاز میشود

خواهد بود که این را که روی لیست های بزرگ بکار نمیبرد.

صربیل (Bubble Sort)

Flag = true;

Pass = 1;

while (pass <= n) && flag

flag = false;

for (j = 1 ; j <= n ; j++)
if (n[j] > n[j+1]) flag = true;

swap (n[j], n[j+1]); j++;

pass = pass + 1;

در این ایجی دستrib کن معرفتی در اولین صریح و نزدیکی بخود را این فراهم کنید
 نزدیکی که خود را
 bubble sort

صریح نازدیک تعلوف صریح کنند
 در قبول هر حرکت در عویل برداشته شده باشد (عیوب)
 مفاسد همچوں واردی و خروجی لزوم جایگاه را
 در این صریح کنند [۱] و [۲] به صورت دفع
 مررگ فرستاد
 مردی بخوبی این نوع صریح (۱) برداشته را آغاز نمایند
 (۲) صریح کر کریمه ایشان
 همچوں صریح کنند
 همچوں صریح کنند
 همچوں صریح کنند

insertion sort (۳) (۴) (۵) (۶)

این دفعه نیز دستrib کار کنید
 این دفعه نیز دستrib کار کنید

این دفعه نیز دستrib کار کنید

این دفعه نیز دستrib کار کنید

این دفعه نیز دستrib کار کنید

void insertion sort(element list[], int n)

int i,j;

element next;

$O(n^2)$

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

next = list[i];

for(j=i-1; j >= 0 & next.key < list[j].key; j--)

list[j+1] = list[j];

list[j+1] = next;

$O(n^2)$

حالات صيغة المدخلات

$O(n^2)$

حالات صيغة المدخلات

$O(n^2)$ صيغة المدخلات

$\underline{O(i)}$ + $O(n^2)$ صيغة المدخلات

II IA C IV I9

II IA C IV I9

صيغة المدخلات

II IA C IV I9

صيغة المدخلات

C II IA C IV I9

الإدخالات

C II IA IV I9

صيغة المدخلات

(دروی) \rightarrow (رئیس اولیه) \rightarrow صورت ملکی کنندگان ایران

بجهات آن صریح نمایند جلسه - جلسه (دروی) برای رسیدگان خارج از کشور

صریح (SIV)

اعطا کنند

تبار رکوردهای خارج از کشور نباید تخصیص مطابق

لیست و گفتگوی (نماینده) را با این ریاست دیواندی کاپی کردن

میکنند

(تبار رکوردهای خارج از کشور = صورت

باید جلسه - جلسه نهادن برای این اتفاق) اعلیٰ کنفرانس بـ ۱۹/۱۶

صریح (SIV)

برخی از این رئیس اولیه ملکی صورتگذار

برخی از این رئیس اولیه خارج از کشور نیارعنه صورتگذار

نامه (Pivot) و چندین کاپی (Pivot) ایجاد کنند

خارج از کشور که خود را در این Pivot نمایند

دانند. و چندین کاپی هم نهادن صورتگذار

بازگشی

Quick (element *list, int left, int right)

↓ int j;

if(left < right)

↓ j = partition (list, left, right);

Quick (list, left, j - 1);

Quick (list, j + 1, right);

↓

j

① →

← ②

R₀ R₁ R₂ R₃ R₄ R₅ R₆ R₇ R₈ R₉

26 5 37 7 67 77 59 75 48 19

77 5 79 175 26 59 67 48 37

quick(0, 4)

quick(5, 9) (end)

1 5 77 79 15 26 59 67 48 37

1 5 77 79 15 26

1 5 77 15 26

→ 48 37 59 67

1 5 77 15 26 37 48 59 67 0

j i ↗

sort

partition

first

new pivot

i, j

list[i] > pivot

list[j] < pivot

list[i] < list[j]

i < j

```

void quickSort(element list[], int left, int right)
{
    int pivot, i, j;
    element temp;

    if (left < right) {
        i = left + 1;
        j = right - 1;
        Pivot = list[left].key;

        do {
            do {
                i++;
            } while (list[i].key <= pivot);

            do {
                j--;
            } while (list[j].key >= pivot);

            if (i < j) {
                swap(list[i], list[j], temp);
            }
        } while (i < j);

        swap(list[left], list[i], temp);
        quicksort(list, left, i - 1);
        quicksort(list, i + 1, right);
    }
}

 $T(n) = \underbrace{o(n)}_{\text{Partition}} + \underbrace{kT(\frac{n}{k})}_{\text{Non-Pivot + Pivot Sublist}}$ 

 $\frac{T(n)}{T} = \frac{o(n) + kT(\frac{n}{k})}{T}$ 
 $T(n) = o(n) + kT(\frac{n}{k})$ 
 $T(n) = o(n) + T(n-k) + T(k)$ 
 $T(n) = o(n) + T(n-k) + T(k)$ 
 $T(n) = o(n)$ 

```

اکنہ لور بارنی حالت خود را Pivot کرے

لر ز

1 4 77 25 48 50 میں کسی $O(n)$ کے سے
ایک ایسا کام کرنے کا سعی دیں

ایک ایسا کام کرنے کا سعی دیں

$T(n-1) \rightarrow O(n)$
صرفاً

کس لور بارنی کو ایسا کر سوئے جائے کہ

50 48 73 77 4 7 اگر ممکن میں کریں تو

$O(n)$

(O(n)) میں 50 بے اخراج

$O(n)$ \rightarrow بڑی ترین $T(n-1)$

از نفع آتی وہ کہ نقصم دلیل اور کا لذت بازی وہ فنازی

صلالہ، ایسا صیغہ اور یہ حتم کرنے

ایسی دو ایسا صیغہ بھی نہیں کہ جسے

انقدر سائنس زبان احتمال میں کریں (O(n) رکھو)

11 10 17

a

b

c

بڑی نسبت

11 10 17

a

b

c

$O(n \log n)$

$O(n^2)$

بڑی نسبت

$O(n^2)$

بڑی نسبت

فروض کریں کہ $T(n)$ میں n کا ایک بارہ کار

ایک ایسا کام کرنے کا سعی دیں

$T(n) \leq kn \log n$

جستجوی سریع (Quicksort) (برخراحت صربی لانگ)

(ج) کففعه به حافظه (خناق) نباشد

برخ) حالات ممکن و ممکن نباشند

حالات عقیق بازگشایی

حافظه صورتیاز شدن

برخ) حالات

حذف از اجرای بازگشایی به زیر می‌رود (حالات)

حذفیات را در (ج) ب حل صفر بگیرید

حذف عقیق بازگشایی

حافظه صورتیاز شدن

صربی لانگ) نسبتاً معنادار از میانه بین اول و آخر است

$$\text{Pivot} = \text{median } P_{K_1, K_{(1+r)/2}, K_{r+1}}$$

$$\text{median } P_{7, 7, 7} = 7$$

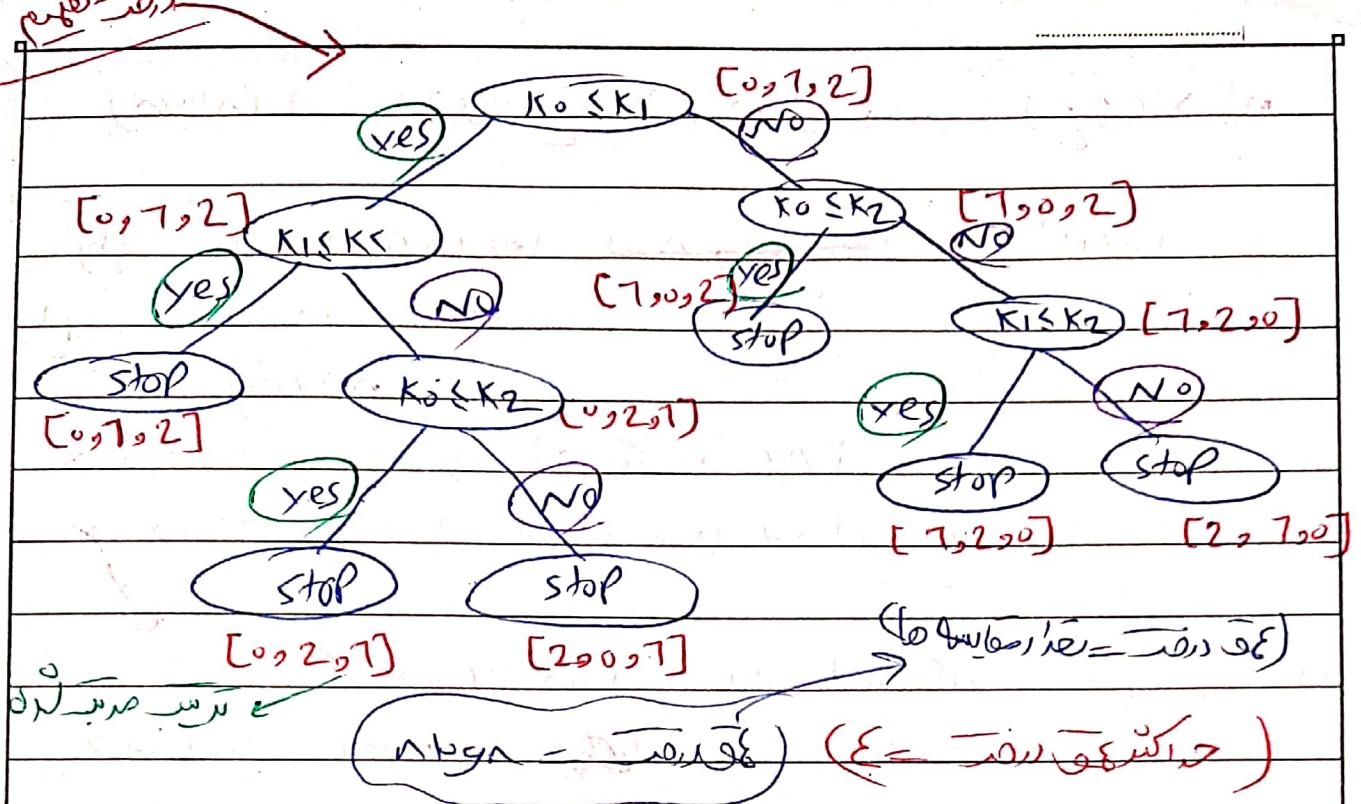
$$\text{median } P_{7, 5, 7} = 7$$

(ج) نسبتاً معنادار از میانه

برخ) نسبتاً معنادار از میانه

برخ) نسبتاً معنادار از میانه

صربی لانگ) این مرحله باید در این مرحله انجام شود



K0 ≤ K1 ≤ K2

پیش می‌گیریم که $K_0 \leq K_1$ دلیل

$K_0 \leq K_2 \wedge K_1 \leq K_2$ نیز

پیش می‌گیریم که $K_1 \leq K_2$

پس از $K_1 \leq K_2$

حافظ $n!$ کریم

برای $n! \log(n)$ از $n^{\log n}$ کمتر است

$\log n + \text{ثابت}$

هر عدد بزرگ باشد از $n^{\log n}$ بزرگ است (باشد و بزرگ)

اختلاف ای امر کند $n^{\log n} > n^{\log(n+1)}$ \Rightarrow با صدای $n^{\log(n+1)} - n^{\log n}$

حافظ $n!$ نمایم

این بزرگ داشته باشد $n^{\log n}$ از $n^{\log(n+1)}$ کمتر است

دقت دوستی که سرشار از $n!$ نمایم $n^{\log(n+1)}$ از $n^{\log n}$ بزرگ داشته باشد

$$n! \cdot 2^{k-1} \rightarrow k \geq \log(n!) + 7$$

$$n! > \left(\frac{n}{2}\right)^{\frac{n}{2}} \rightarrow \log(n!) > \frac{n}{2} \log \frac{n}{2} = \underline{n \log n}$$

$$\log n! = \cancel{\log(1x2x3x\ldots x)} + \log(nx(n-1)x(n-2)\ldots x1)$$

$$= \log n + \log n - 1 + \dots + \underbrace{+ 1}_{(n \log n)}$$

* درجه قدر باش حافظ ! هر بگر دلله بخواه

* درجه درجه دویم ! هر بگر دسته بخواه و خود را

هر آنونی که گفت این مقدار خوبها صد کند و بیچیزی رسانی نمایند

حال آنون (n log n) سه مرتبه

درجه درجه دویم) هر بگر مطمئنی مطلقاً سر از رسیده بگر از n log n سه مرتبه

متوجه
متوجه
متوجه

که زمان اجر $n \log n$
که زمان اجر n^2 که زمان اجر n^3

که زمان اجر n^2 که زمان اجر n^3 که زمان اجر n^4

که زمان اجر n^3 که زمان اجر n^4 که زمان اجر n^5

که زمان اجر n^4 که زمان اجر n^5 که زمان اجر n^6

اگه آنونی و بتری زمان اجر بسیار بزرگ شود

(n^6) همراه

اگه همچو افلاکاتی (ربا، ربا) همراه

بازم اما اگه افرادان سه

زدایم اما اگه افرادان سه

بازم اما اگه افرادان سه