

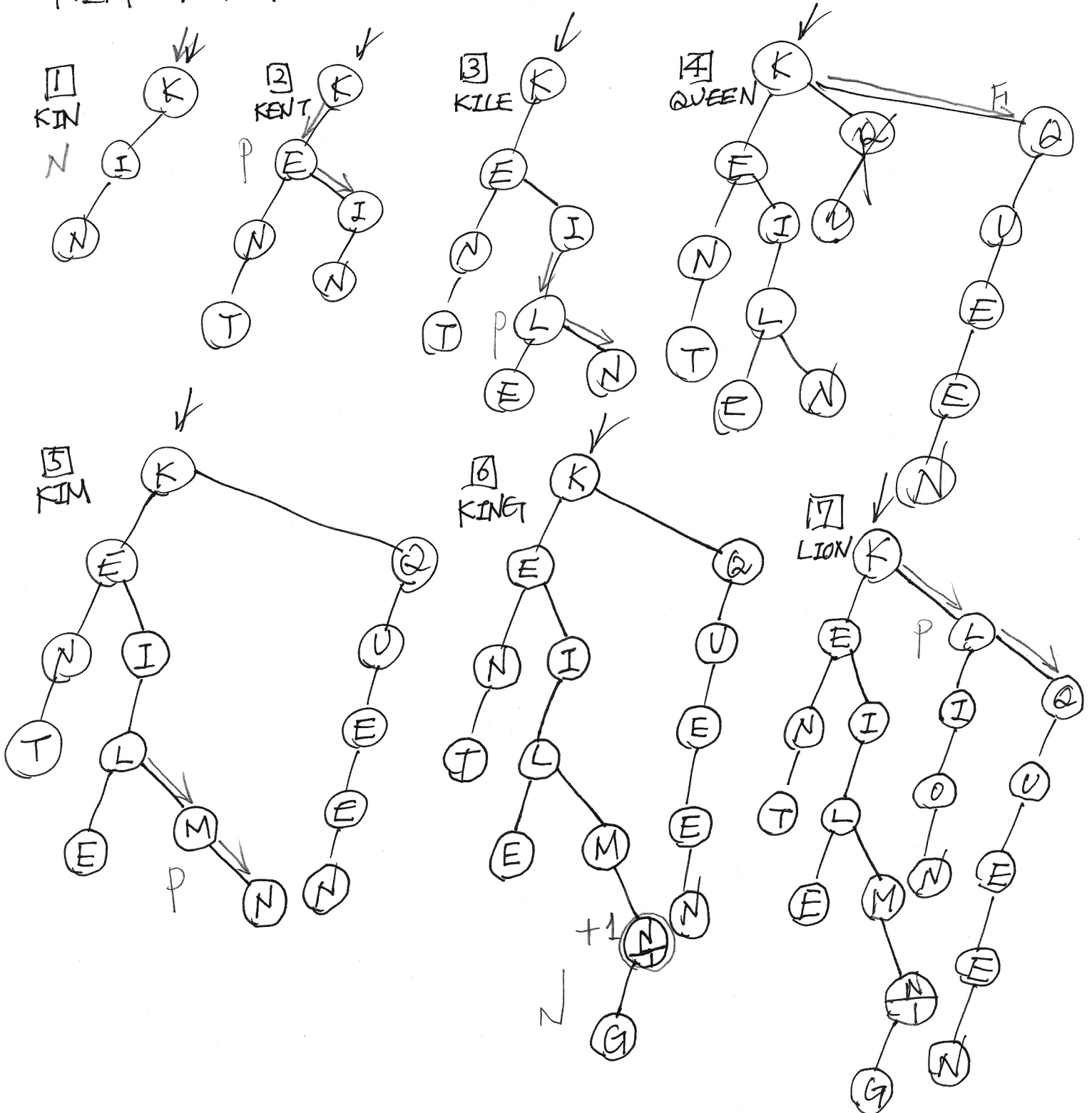
2/27/2007

Dr. Dong-Keun Shin
the greatest computer scientist in the world

checked with PD-code
on 4/24/2007

Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea

KIN KENT KILE QUEEN
KIM KING LION KIND JADE

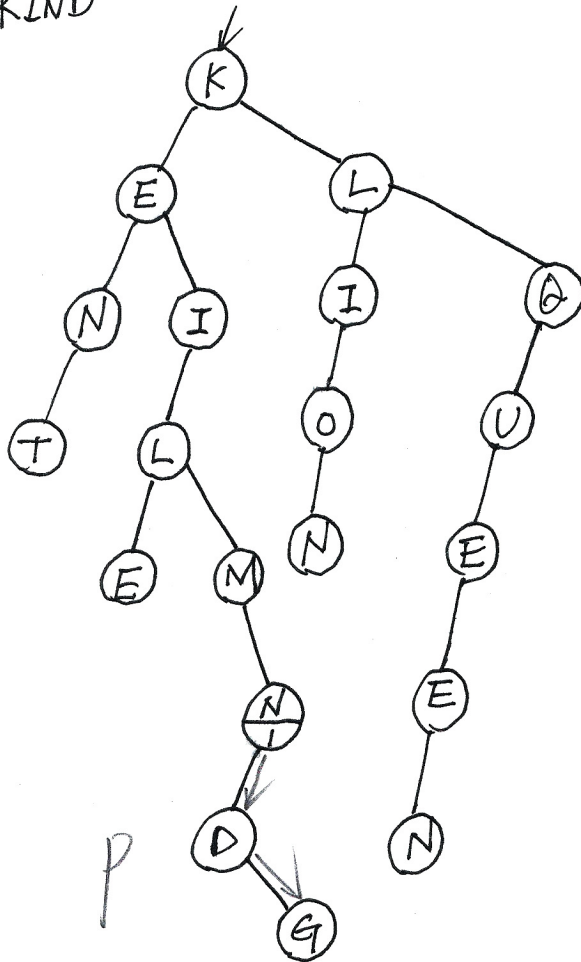


2/17/2007

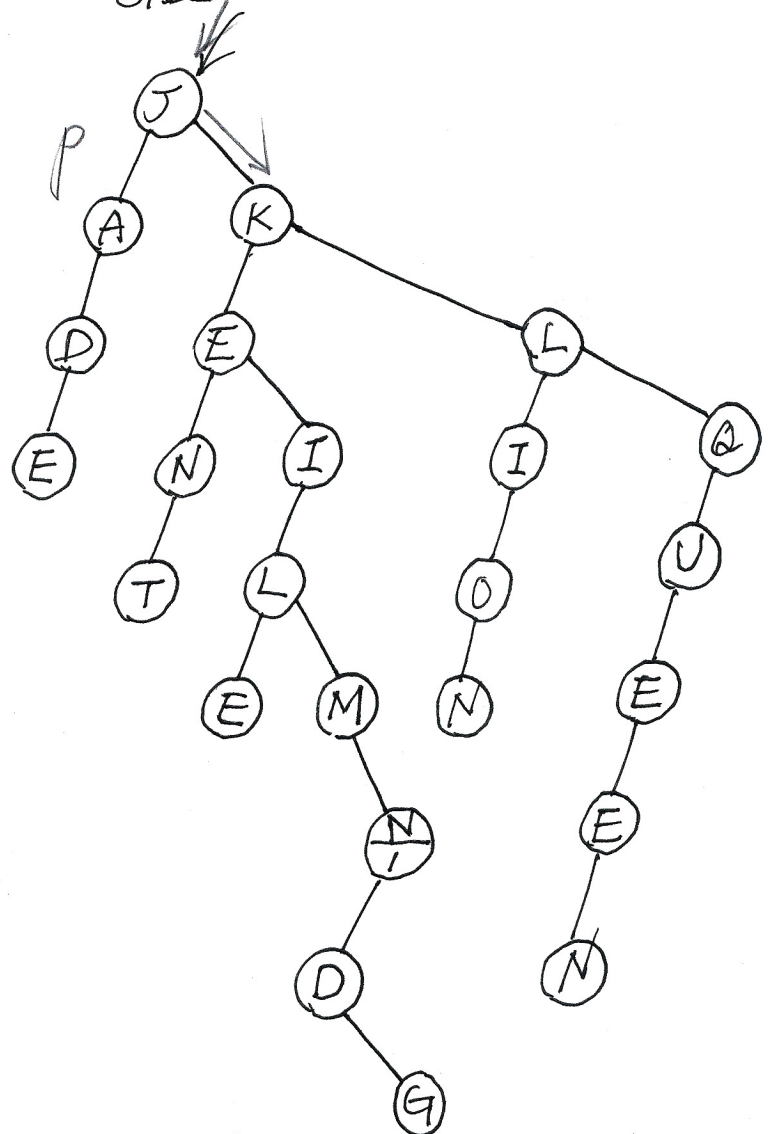
Dr. Dong-Keun Shin
the greatest computer scientist in the world

Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea

[8]
KIND



[9]
JADE



JADE KENT KILE KIM KIN KIND KING
LION QUEEN

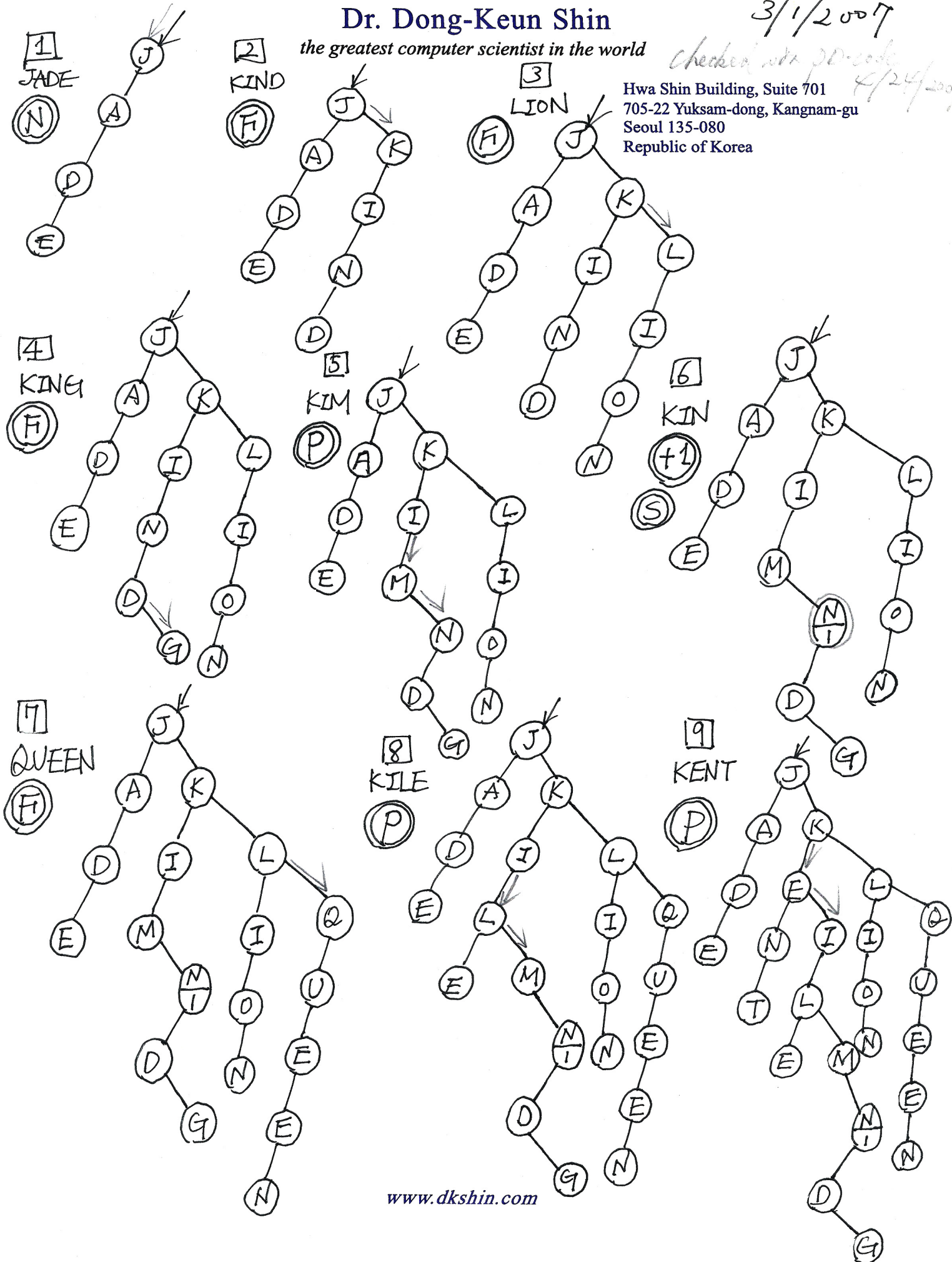
JADE KIND LION KING KIM
KIN QUEEN KILE KENT

Dr. Dong-Keun Shin

the greatest computer scientist in the world

3/1/2007

Hwa Shin Building, Suite 701
705-22 Yulsam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea



JADE KIND LION KIM KING KENT
KIN QUEEN KILE

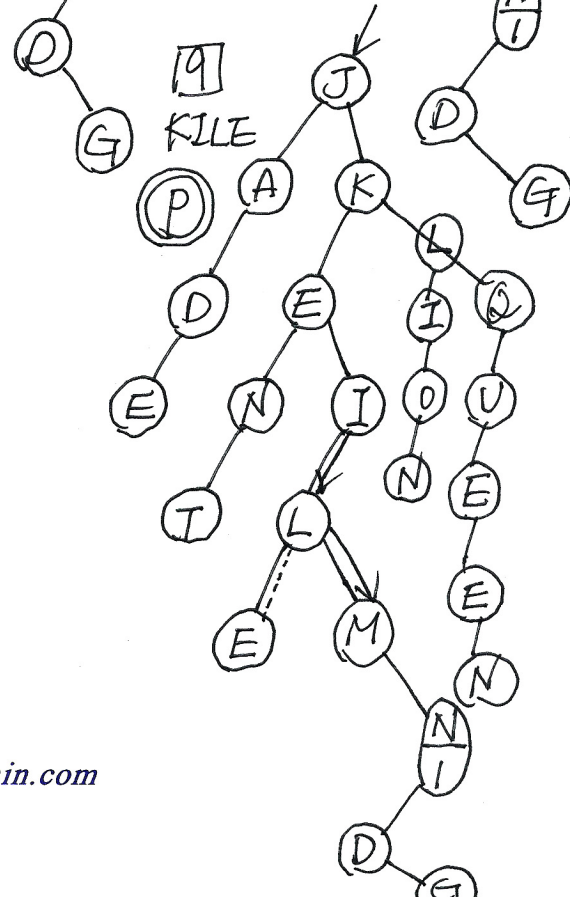
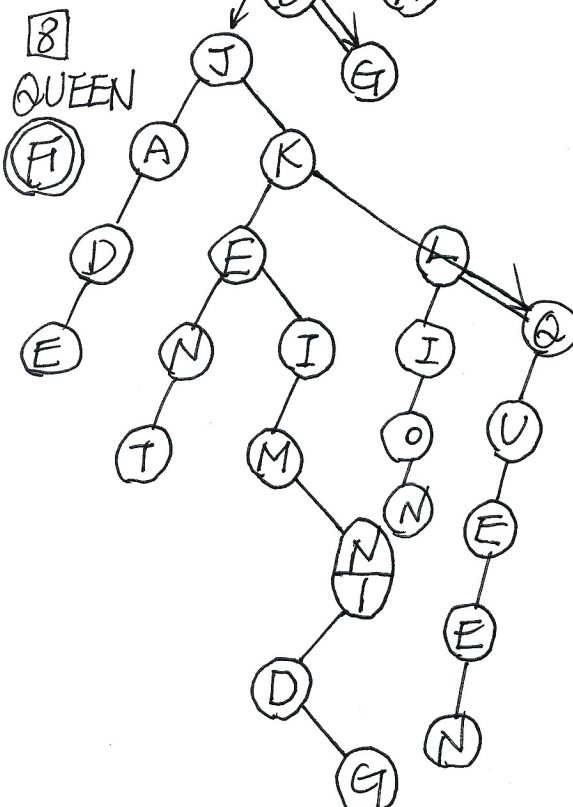
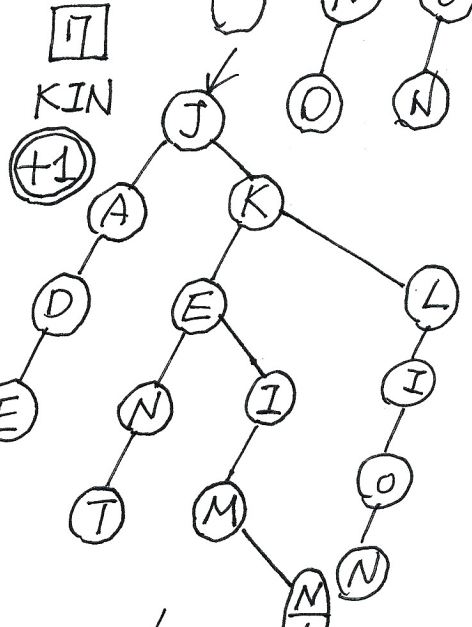
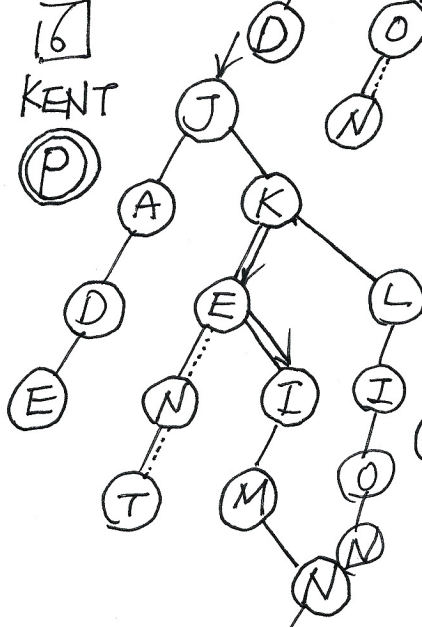
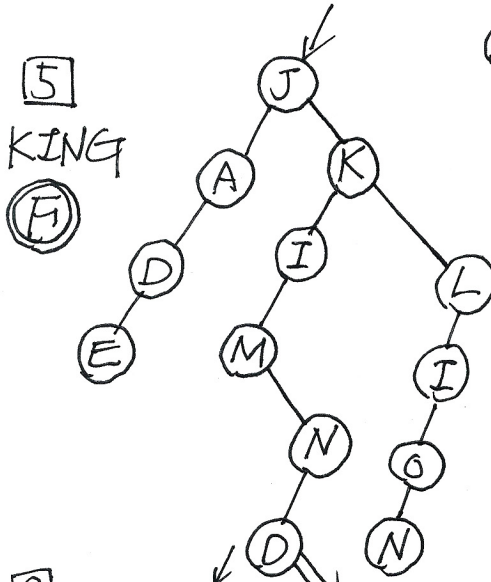
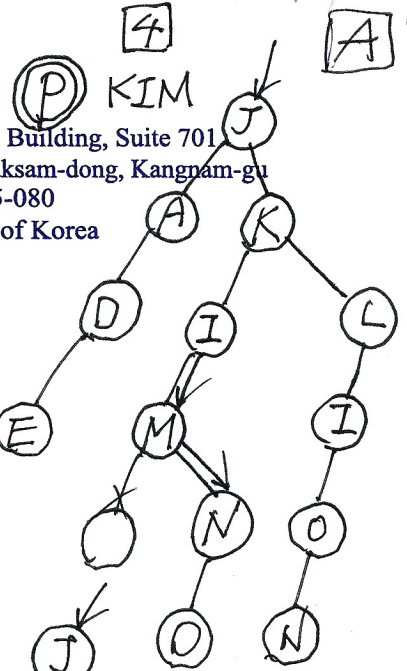
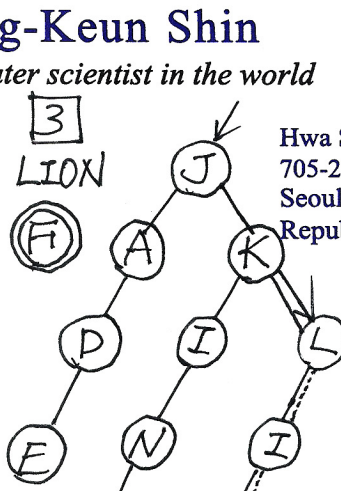
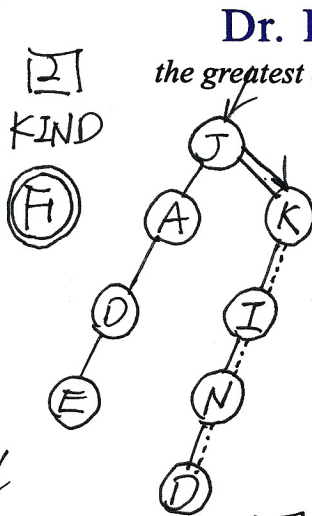
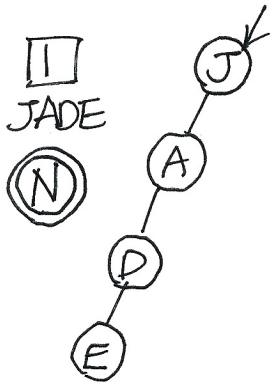
3/2/2007

chkd 4/5/2007

Dr. Dong-Keun Shin

the greatest computer scientist in the world

Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea



KING KIND KENT KIN KILE KIM

3/2/2007

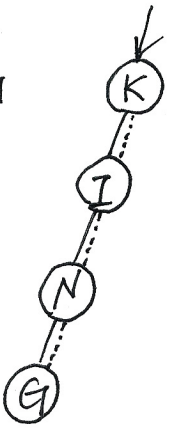
LION QUEEN JADE

Dr. Dong-Keun Shin

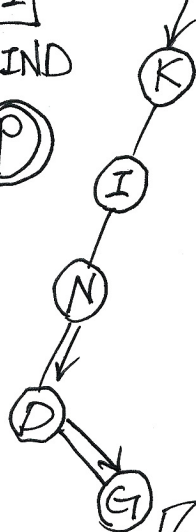
the greatest computer scientist in the world

Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea

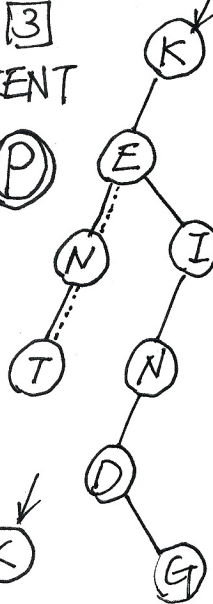
[1]
KING
(N)



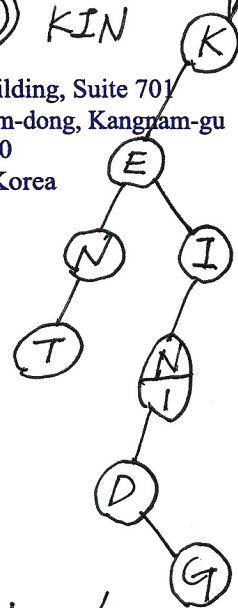
[2]
KIND
(P)



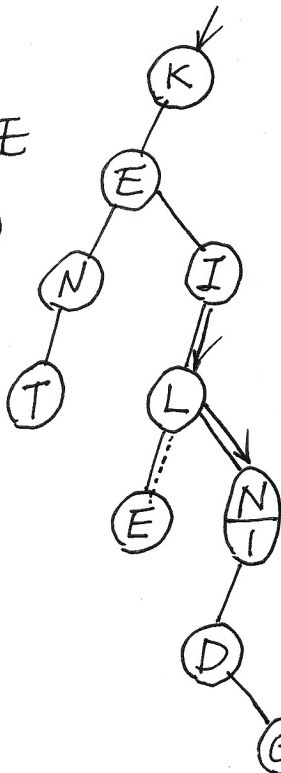
[3]
KENT
(P)



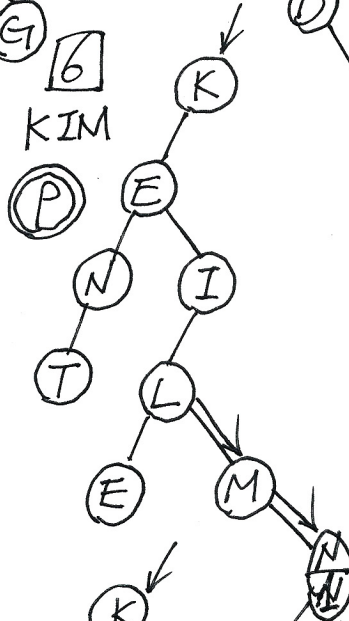
[4]
KIN
(+)



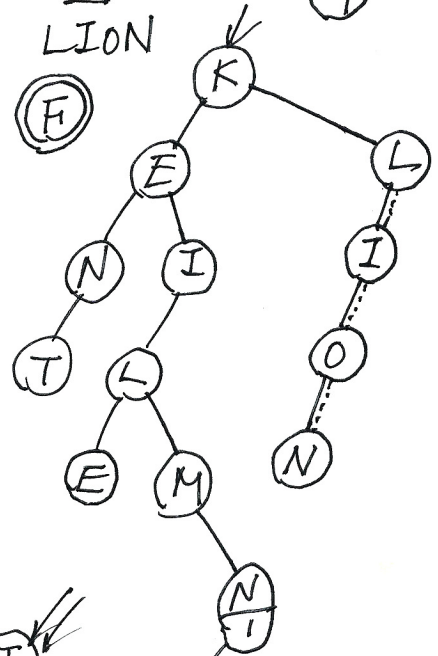
[5]
KILE
(P)



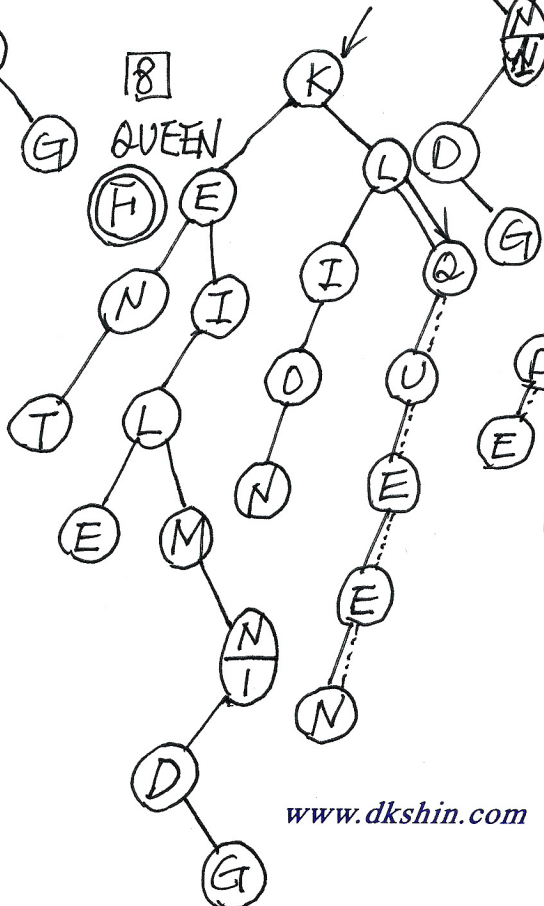
[6]
KIM
(P)



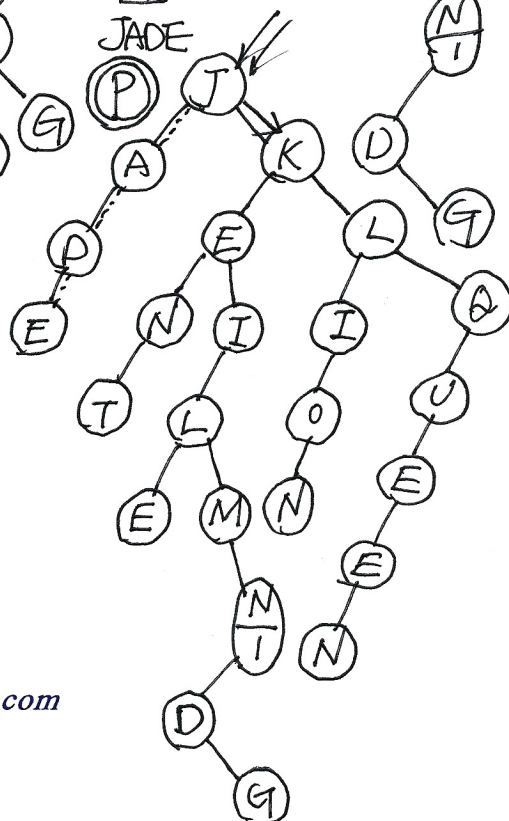
[7]
LION
(F)



[8]
QUEEN
(H)



[9]
JADE
(P)



KIN, KENT, KILE, KIND, KIM, KING,
QUEEN, JADE, LION

3/2/2007

Chkd 4/5/2007

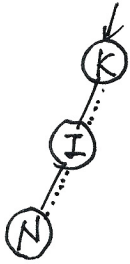
Dr. Dong-Keun Shin

the greatest computer scientist in the world

Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea

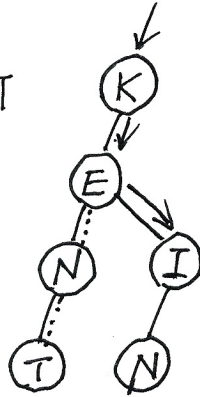
1
KIN

(N)



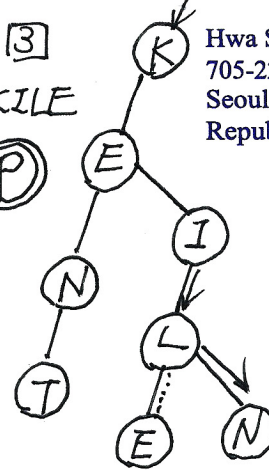
2
KENT

(P)



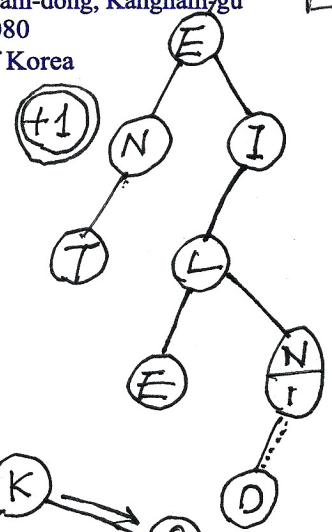
3
KILE

(P)



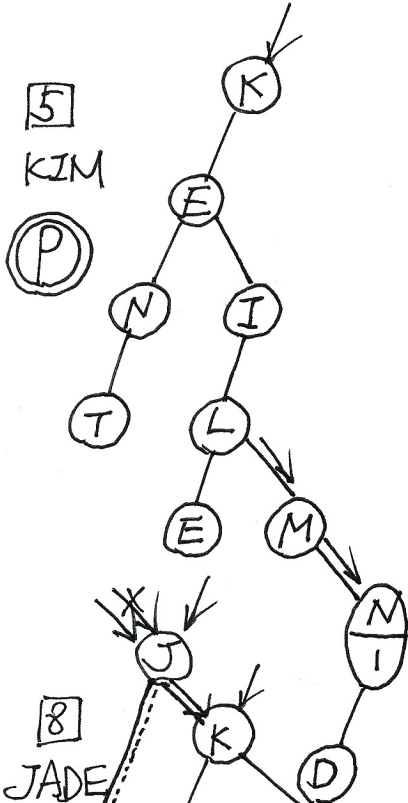
4
KIND

(C)



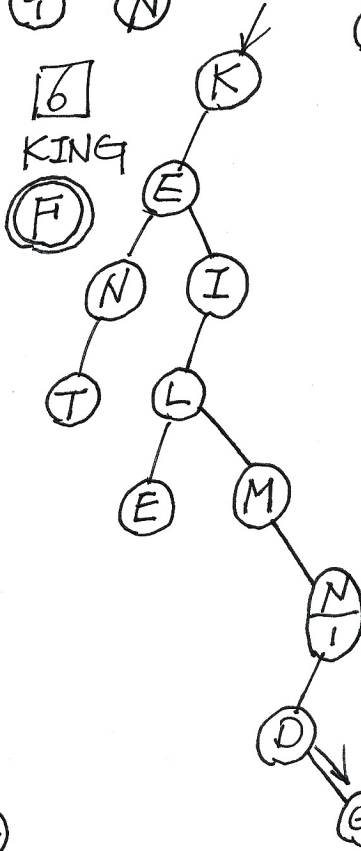
5
KIM

(P)



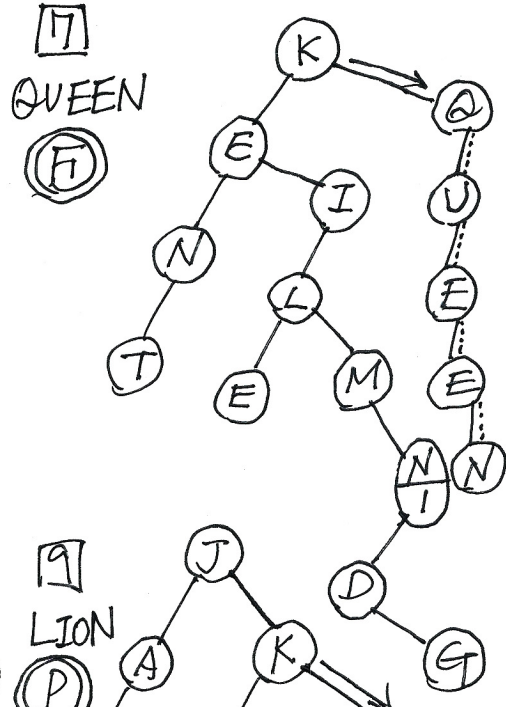
6
KING

(F)



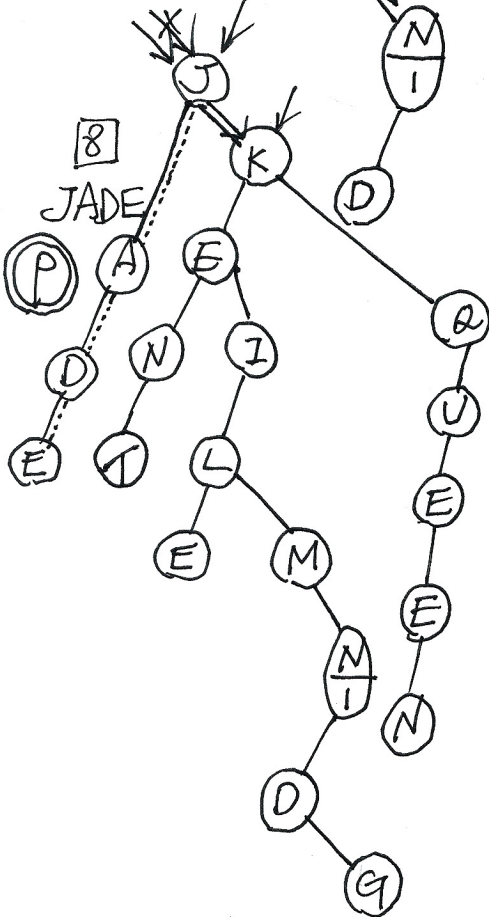
7
QUEEN

(F)



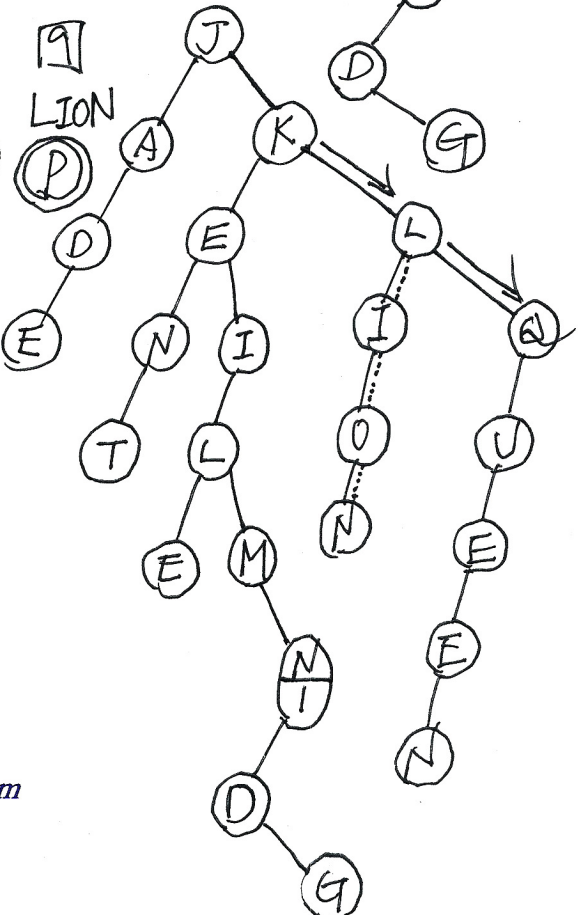
8
JADE

(P)



9
LION

(P)



LEE, KIM, KING, KITE, KENT, KIN, KANT, KILE,
 KIMMY, KIMBERLY, KITE, KID, KINS, JACOB, KANG,
 KITE

Dr. Dong-Keun Shin

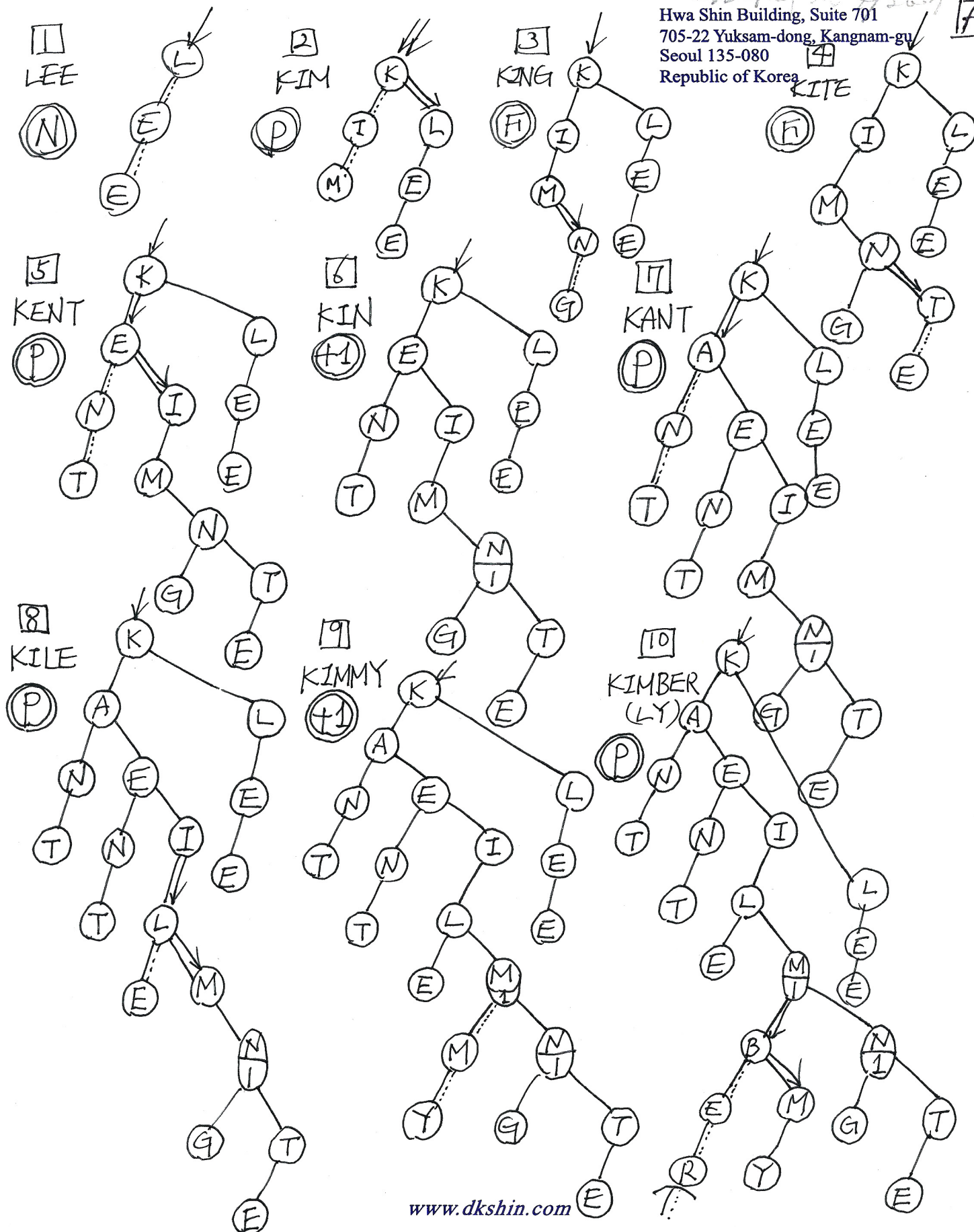
the greatest computer scientist in the world

~~KITE~~

3/5/2007

Hwa Shin Building, Suite 701
 705-22 Yuksam-dong, Kangnam-gu
 Seoul 135-080
 Republic of Korea

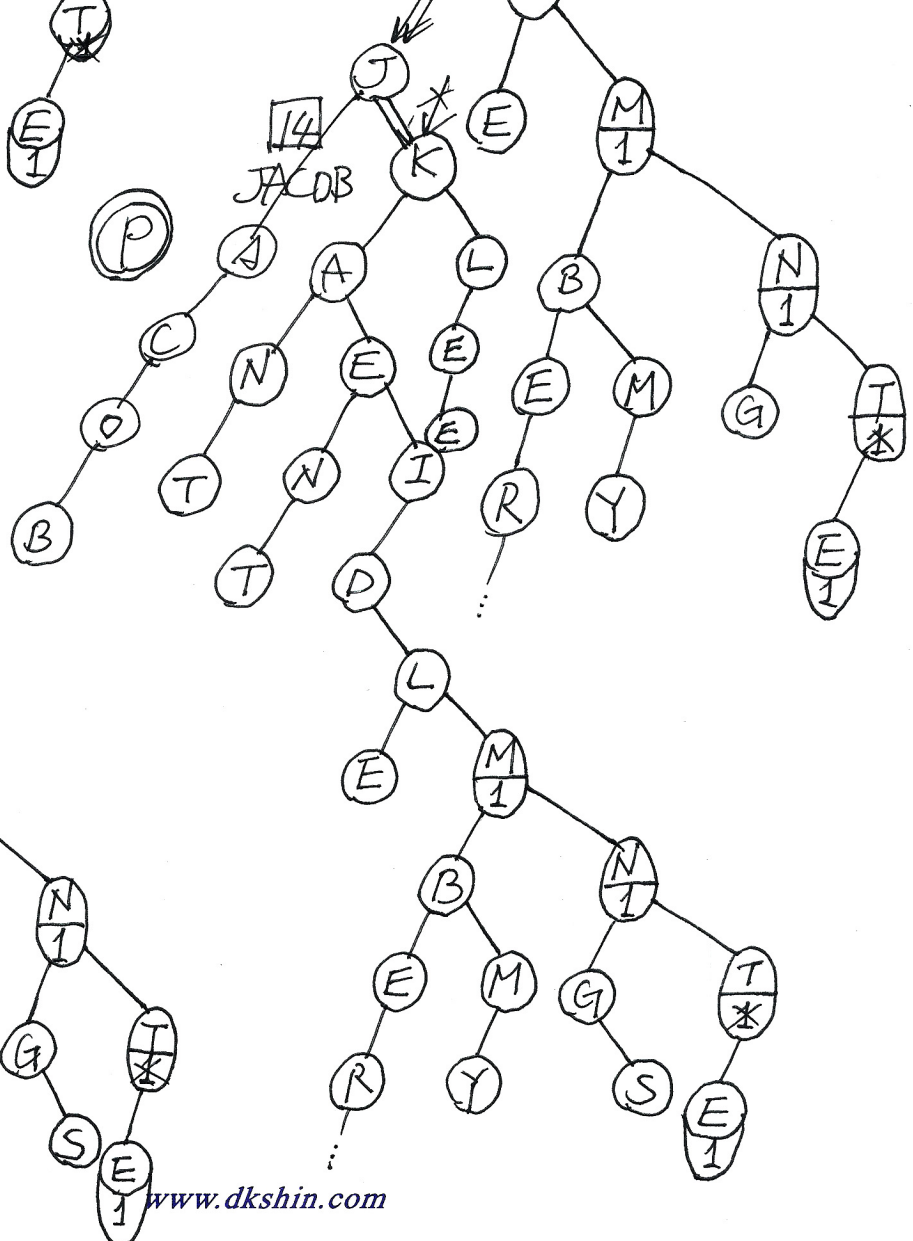
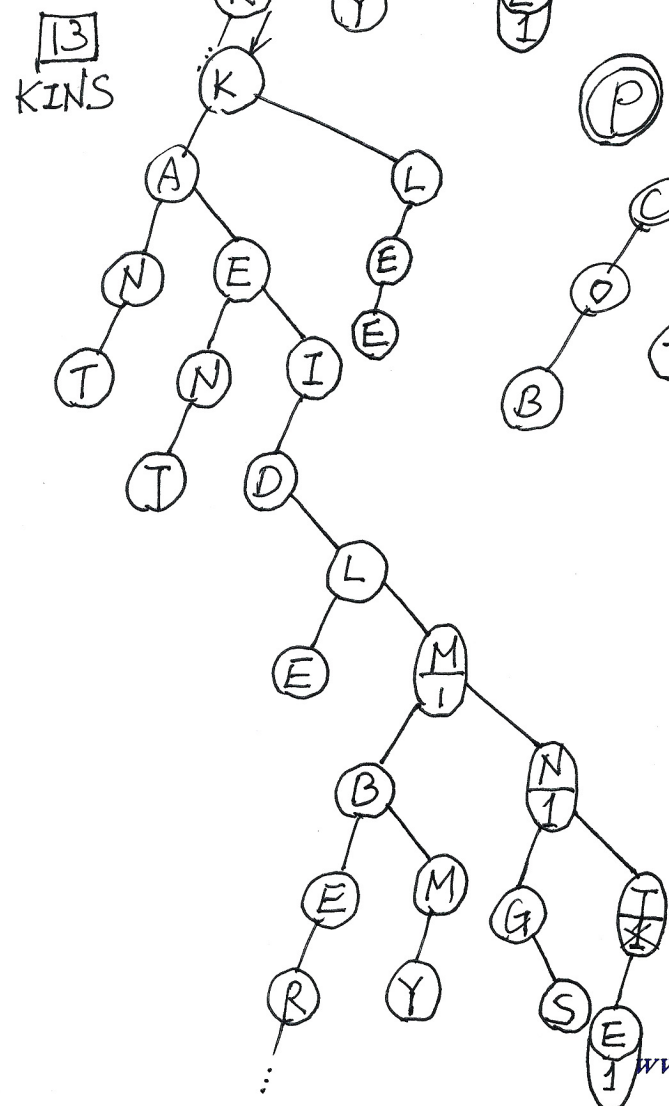
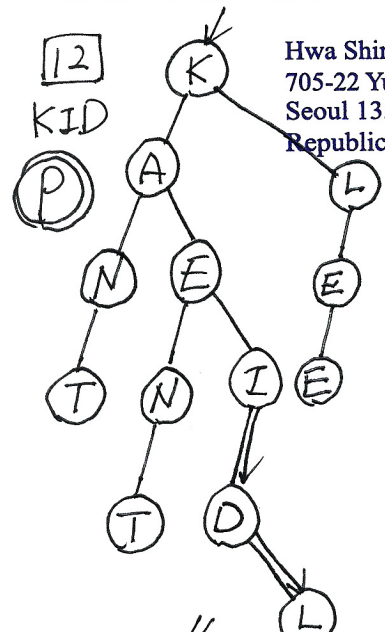
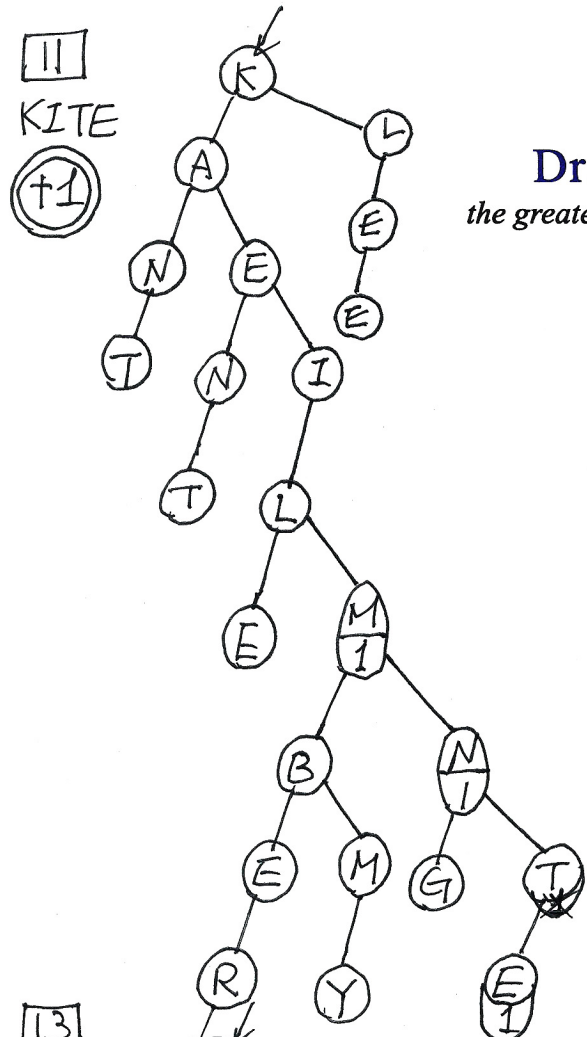
[A]



3/5/2007

Dr. Dong-Keun Shin
the greatest computer scientist in the world

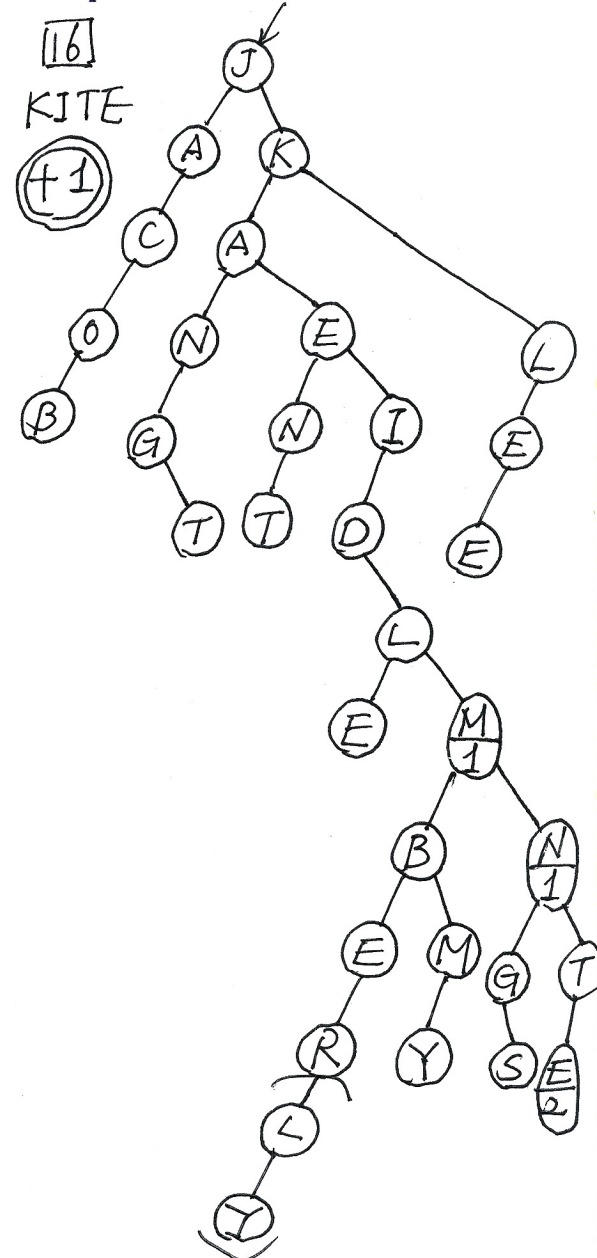
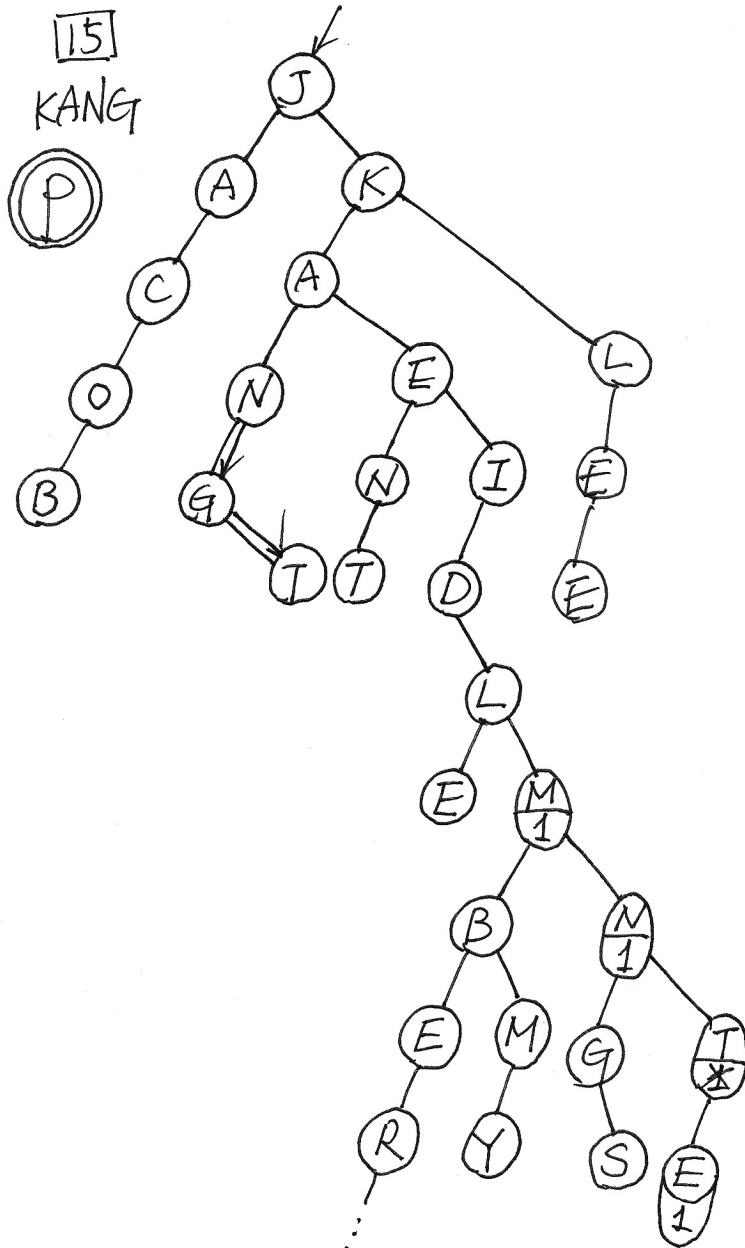
Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea



3/5/2007

Dr. Dong-Keun Shin
the greatest computer scientist in the world

Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea



JACOB, KANG, KANT, KENT, KID, KILE, KIM,
KIMBER(LY), KIMMY, KING, KINS, KITE, KITE, KITE,
LEE

KIN ~~KIN~~

316, 273, 4329, 9837, 16, 274, 107, 273, 5, 272

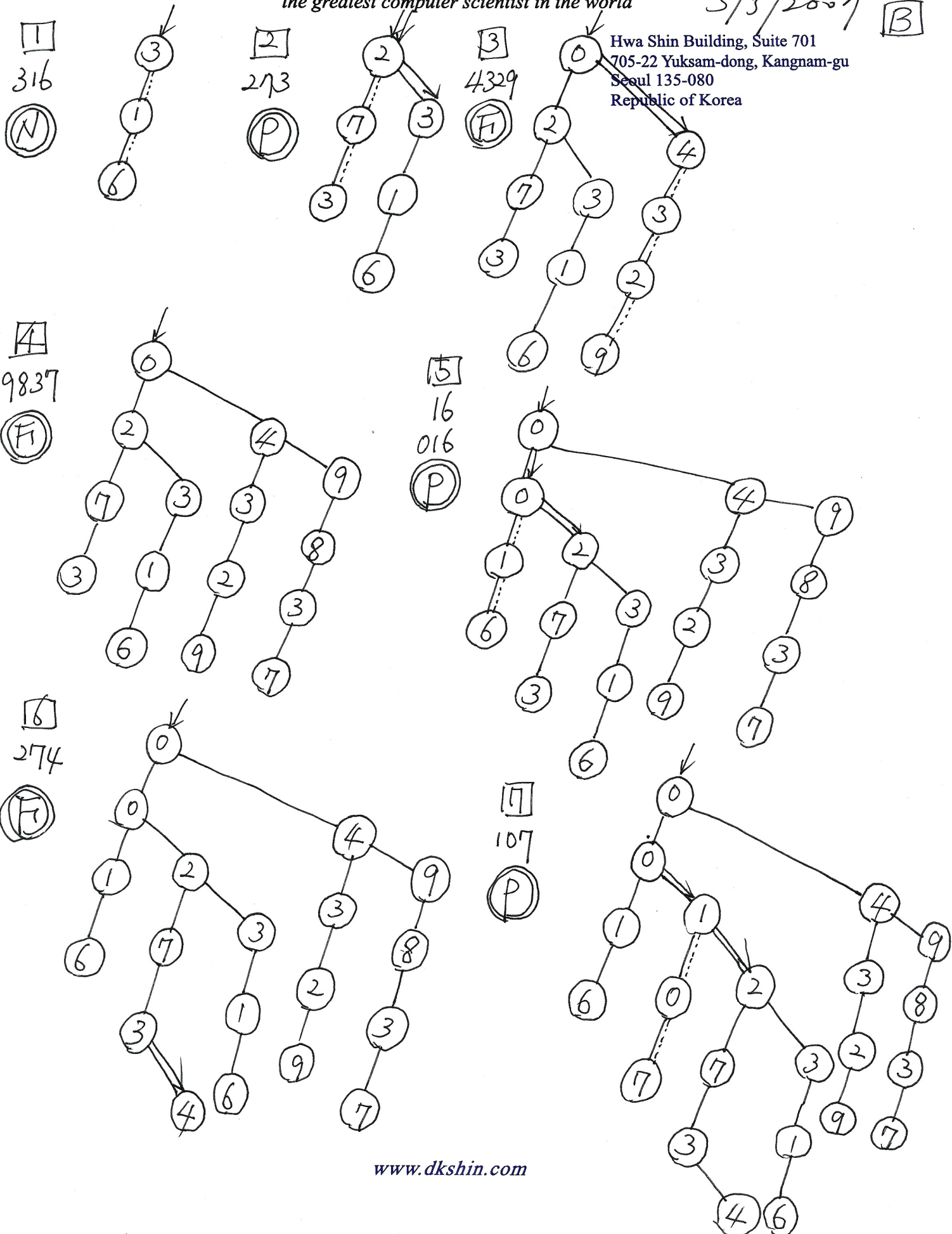
(10 input)

3/5/2007 [B]

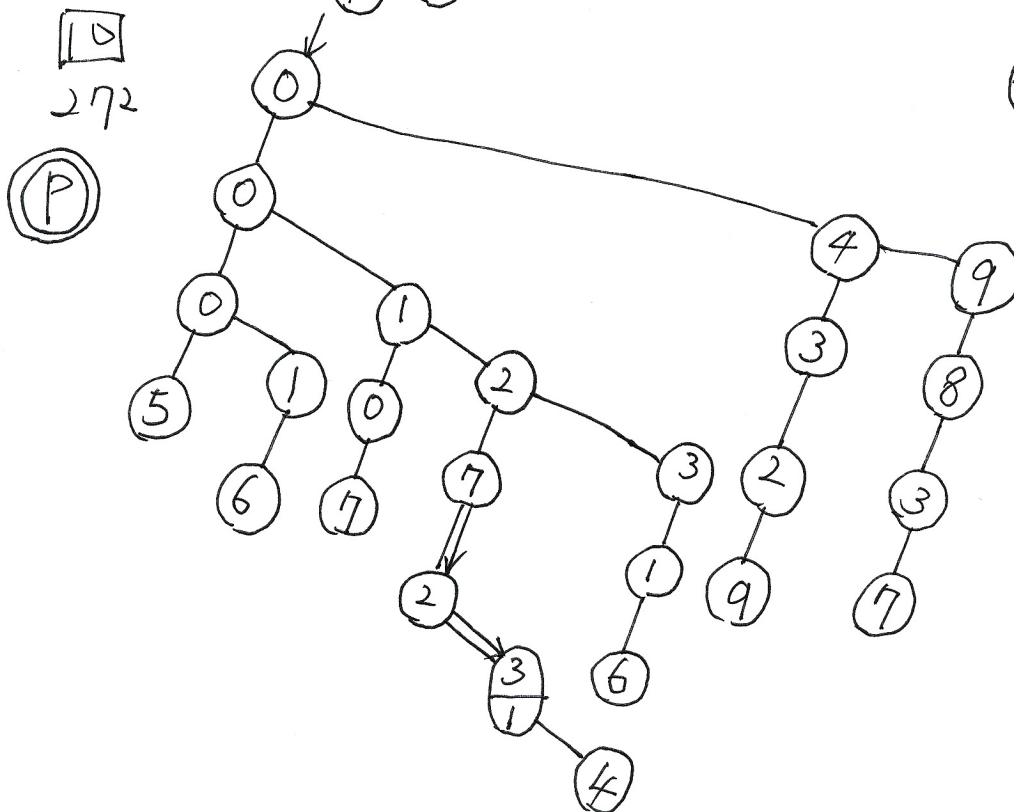
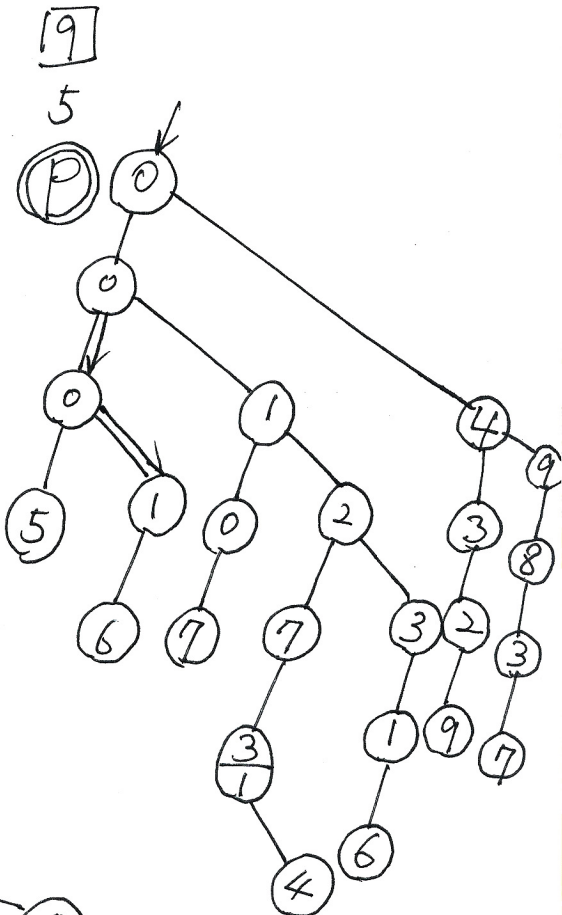
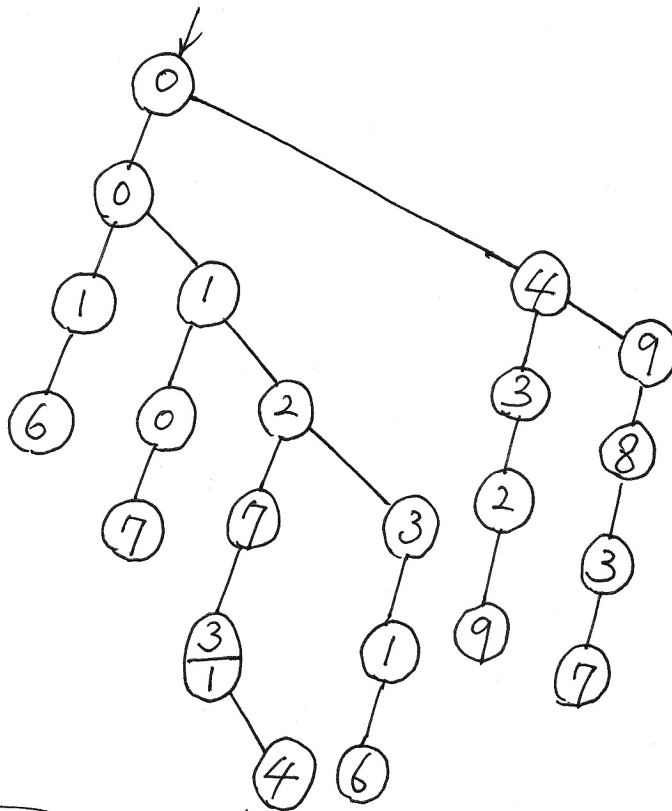
Dr. Dong-Keun Shin

the greatest computer scientist in the world

Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea



3/5/2007

$$\begin{array}{r} 18 \\ 273 \\ \hline +1 \end{array}$$


5, 16, 107, 272, 273, 273, 274, 316, 4329, 9837

Integer: fill up sufficient no. zeros
Ndx on 3rd

4/29

inner
Inpt pt ← Ptd Trie

처음 Inpt는 무조건 Set Up할까?

Pt Trie가 Dull 이면 준성이는 Hdr Pt (Paul or Lucas Son or dull) ← Currnt Inpt Nd

남은 Inpt Chars로 무조건 길게 엮어 만들고 끝내기;

Ptd Inpt Ch 가 Ptd Trie Ch 와 Eq이면 준성이는 무조건 "Just stop the calling" 을 하고 영난다.

(1) Hdr Pt ← Trie Pt Paul Lft 하고

(2) Trie Pt ← Trie Nd's Paul Lft Chld Pt

(3) if Paul Lft = dull or No More Inpt Chars, Left, Cntr gets incremtd by 1

else Inpt Pt gets Incremtd by 1. 그리고는 리턴

Ptd Inpt Ch 가 Ptd Trie Ch 보다 Lss[P] 면 준성이는 "Who peaches the..."

라고 묻는다.

(1) Hdr Trie Pt (Lucas Rgt or Paul Lft) gets Inpt Ch Nd 를 가르키게 바꾸고

(2) Inpt Ch Nd의 Rit Chld를 Ptd Trie Ch를 가르키게 하고

(3) 남은 Inpt Chs들을 왼쪽으로 엮어주고 나서는 리턴.

Ptd Inpt Ch 가 Ptd Trie Ch 보다 Grt면 준성이는 "Go ahead your infingement" 라고 말한

(1) Hdr Pt ← Trie Pt Lucas Rgt 하고

(2) Trie Pt ← Nd's Rht Chld Pt 그리고는 리턴

Hdr pt's Cntr Nd.

Then no more recursion call

Inpt Chs Nd's Son

Trie pt gets what no more dull

No call afterwards more recursive

① Set-Inpt- and Ptn

② Hdr ← Root Pt

Root pt = Dull or m

Inpt pt = Root pt
Root pt ← Inpt pt
no call

③

Root pt's char <

Inpt's char then OK call

Inpt's char then change

Inpt's char then stay OK call

Inpt Rgt chld ← Root Pt

Root pt ← Input pt (1st char)

do not call routine Inpt pt ← Rht

Trie pt ← Root pt

call routine

Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea

www.dkshin.com

from the laboratory of:
Dr. Dong-Keun Shin

Date: 6/20/2007

Several Primitive Operations on Shin's Tree

1. Search a key in the Shin tree
 2. Print keys in sorted order
 3. Insert a key into the tree
-
4. Traverse the Shin tree
 - a) preorder
 - b) snorder
 - c) inorder
 - d) postorder
 5. Convert Shin tree to reverse one
vice versa



Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea

www.dkshin.com

from the laboratory of:

Dr. Dong-Keun Shin

Search 5PM

Date:

6/20/2007

① Search a key

② printing in order

③ traversing ^{preorder} ^{inorder} ^{postorder}

④ reverse shin ^{reverse}

⑤ insert key to tree

flag set

count > 0 then found else not found

① Tree-pt = nil and Key-pt = nil

② vice versa

Call R-R

only nil and found

③ nil nil

found counter

check

≠ 0

Call R-Routine (Tree-pt, key-pt)

if Key-pt.ch > Tree-pt.ch then

Tree-pt := Tree-pt.rightson

Call Recursive Routine (Tree-pt, key-pt)

if Key-pt.ch < Tree-pt.ch then

writeln (The Key is not found);

Try "KINE" in the sample tree.



Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea

www.dkshin.com

from the laboratory of:

Dr. Dong-Keun Shin

8PM

Date: 6/20/2007

Search on Shin's tree

get Tree-pt from Root-pt and get Key-pt
call Recursive-Search (Tree-pt, Key-pt)

Recursive-Search (Tree-pt: Node-Type,
Key-pt: Node-Type):

if both Tree-pt and Key-pt are nil then Key-Found
else if (Tree-pt = nil) and (Key-pt ≠ nil) then Not-Found
else if (Tree-pt ≠ nil) and (Key-pt = nil) then
if Counter-Flag > 0 then found else Not-Found
else if Key-pt↑.ch = Tree-pt↑.ch then
if Tree-pt↑.counter > 0 then set-Flag else reset
Key-pt := Key-pt↑.LeftSon
Tree-pt := Tree-pt↑.LeftSon
Recursive-Search (Tree-pt, Key-pt) called
else if Key-pt↑.ch > Tree-pt↑.ch then
Tree-pt := Tree-pt↑.RightSon
Recursive-Search (Tree-pt, Key-pt) called
else if Key-pt↑.ch < Tree-pt↑.ch then
Not-Found



Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea

www.dkshin.com

from the laboratory of:

Dr. Dong-Keun Shin

Date: 6/21/2007

```
procedure Print-Sorted-Keys (Tree-pt: Node-Type);  
begin  
  if Tree-pt = nil then  
    Print-out-Key;  
  else  
    begin  
      if Tree-pt↑.LeftSon <> nil then  
        Store-Node-In-Stack  
        Print-Sorted-Keys (Tree-pt↑.LeftSon);  
      Print-Sorted-Keys (Tree-pt↑.RightSon);  
    end;  
end;
```

No Good!



Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea

www.dkshin.com

from the laboratory of:
Dr. Dong-Keun Shin

Date: 6/21/2007

```
if Tree-pt <> nil then
begin
  if Tree-pt.LeftSon <> nil then
    Store-Node-In-Stack
  else
    print-Out-Key;
  Print-Sorted-Keys (Tree-pt.LeftSon);
  pop-Up-Node-In-Stack;
  Print-Sorted-Keys (Tree-pt.RightSon);
end;

check-counter ?
```

DKSHIN.COM

Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea
www.dkshin.com

from the laboratory of:
Dr. Dong-Keun Shin

Date: 6/21/2007 → 6/22/2007

```
if Tree-pt <> nil then
begin
  if (Tree-pt↑.LeftSon <> nil) then
  begin
    store-Node-In-Stack
  end
  if Tree-pt↑.compar > 0 then Print-At-Key-Based-on-Counter
  else
    print-out-Key;
  Print-Sorted-Keys (Tree-pt↑.LeftSon);
  pop-up-Node-In-Stack;
  Print-Sorted-Keys (Tree-pt↑.RightSon);
end;

check-counter ?
```

~~or (Tree-pt↑.compar > 0)~~

~~Print-Sorted-Keys (Tree-pt↑.LeftSon);~~

~~pop-up-Node-In-Stack;~~

~~Print-Sorted-Keys (Tree-pt↑.RightSon);~~



Shin Sort and Search

Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea

www.dkshin.com

from the laboratory of:

Dr. Dong-Keun Shin

Date: 6/22/2009

```
procedure Print-Sorted-Keys (Tree-pt:
                             Node-Type);
begin
  if Tree-pt <> nil then
    begin
      if (Tree-pt ↑. LeftSon <> nil) then
        begin
          store-Node-In-Stack
          if Tree-pt ↑. Counter > 0 then
            Print-Out-Key-Based-On-Counter;
          ← Print-Sorted-Keys (Tree-pt ↑. LeftSon);
        end;
      else
        begin
          Print-Out-Key;
        end;
      ← Pop-Up-Node-In-Stack {one node only};
      ← Print-Sorted-Keys (Tree-pt ↑. RightSon);
    end;
end;
```



Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea

www.dkshin.com

from the laboratory of:
Dr. Dong-Keun Shin

Date: 6/29/2007

```
procedure Print-Sorted-Keys (Tree-Pt: Node_Type);  
begin  
  if Tree-Pt <> nil then  
    begin  
      if (Tree-Pt↑.LeftSon <> nil) then  
        begin  
          Store-Node-In-Stack; {current Node}  
          if Tree-Pt↑.Counter > 0 then  
            Print-Out-Key-Based-On-Counter;  
          Print-Sorted-Keys (Tree-Pt↑.LeftSon);  
        end  
      else  
        begin  
          Print-Out-Key;  
        end  
      Print-Sorted-Keys (Tree-Pt↑.RightSon);  
      Pop-Up-Node-In-Stack; {one node popped}  
    end;  
end;
```



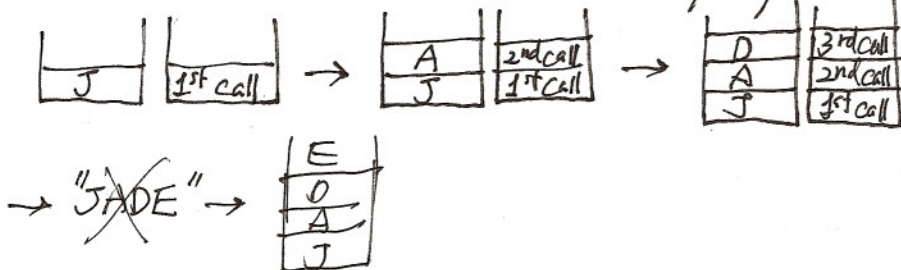
Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea

www.dkshin.com

from the laboratory of:

Dr. Dong-Keun Shin

Date: 6/29/2007



{ 2. Printing Shin Tree}

Procedure Print_Sorted_Keys (Tree_Pt: Node_Type);

begin

if Tree_Pt <> nil then

begin

Store_Node_In_Stack; {Store the David Shin's current node}

if Tree_Pt^.LeftSon = nil then

Print_Out_Key

else

if Tree_Pt^.Counter > 0 then

Print_Out_Key_Based_On_Counter;

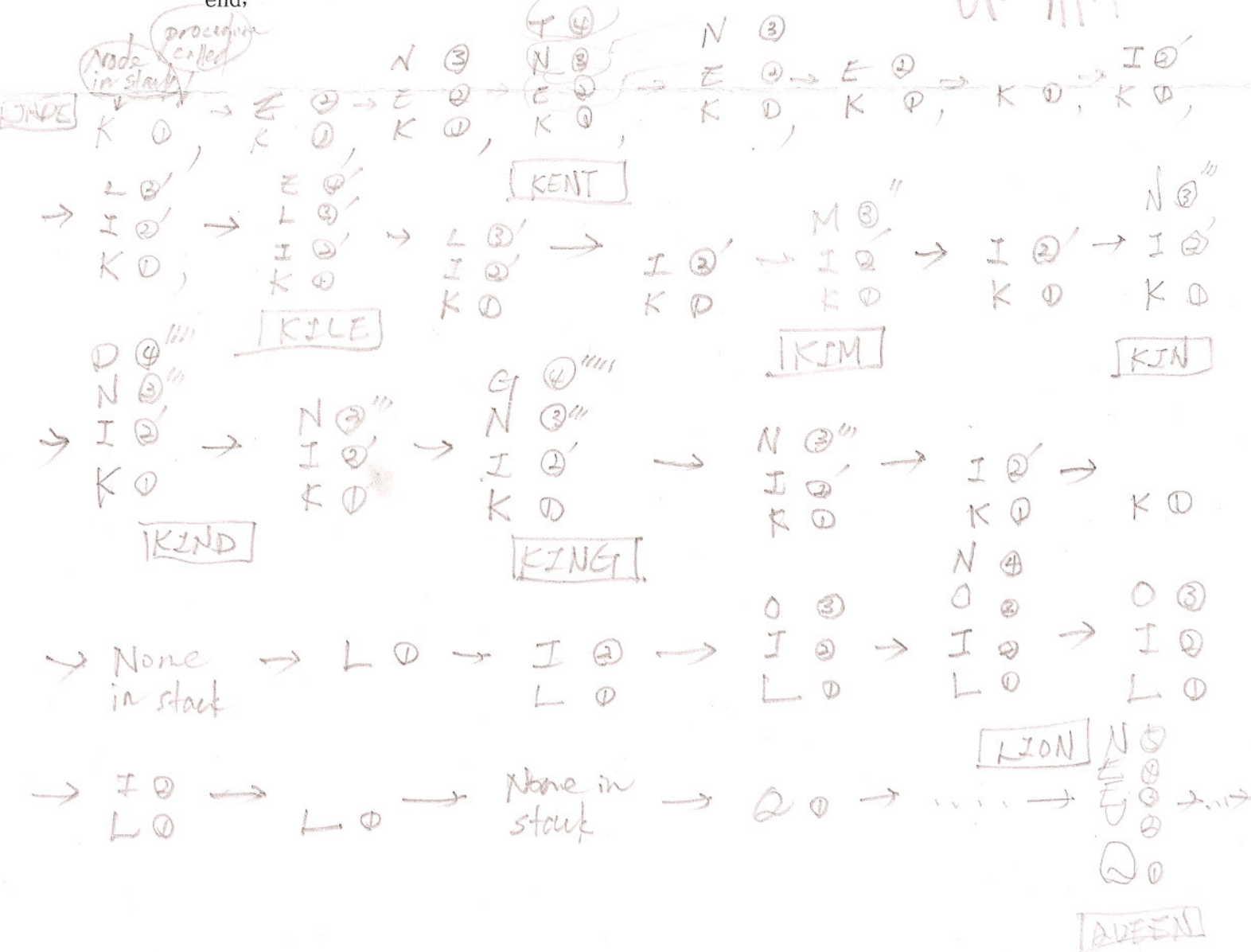
Print_Sorted_Keys (Tree_Pt^.LeftSon);

Print_Sorted_Keys(Tree_Pt^.RightSon);

Pop_Up_Node_In_Stack; {Only one Shin's node popped up}

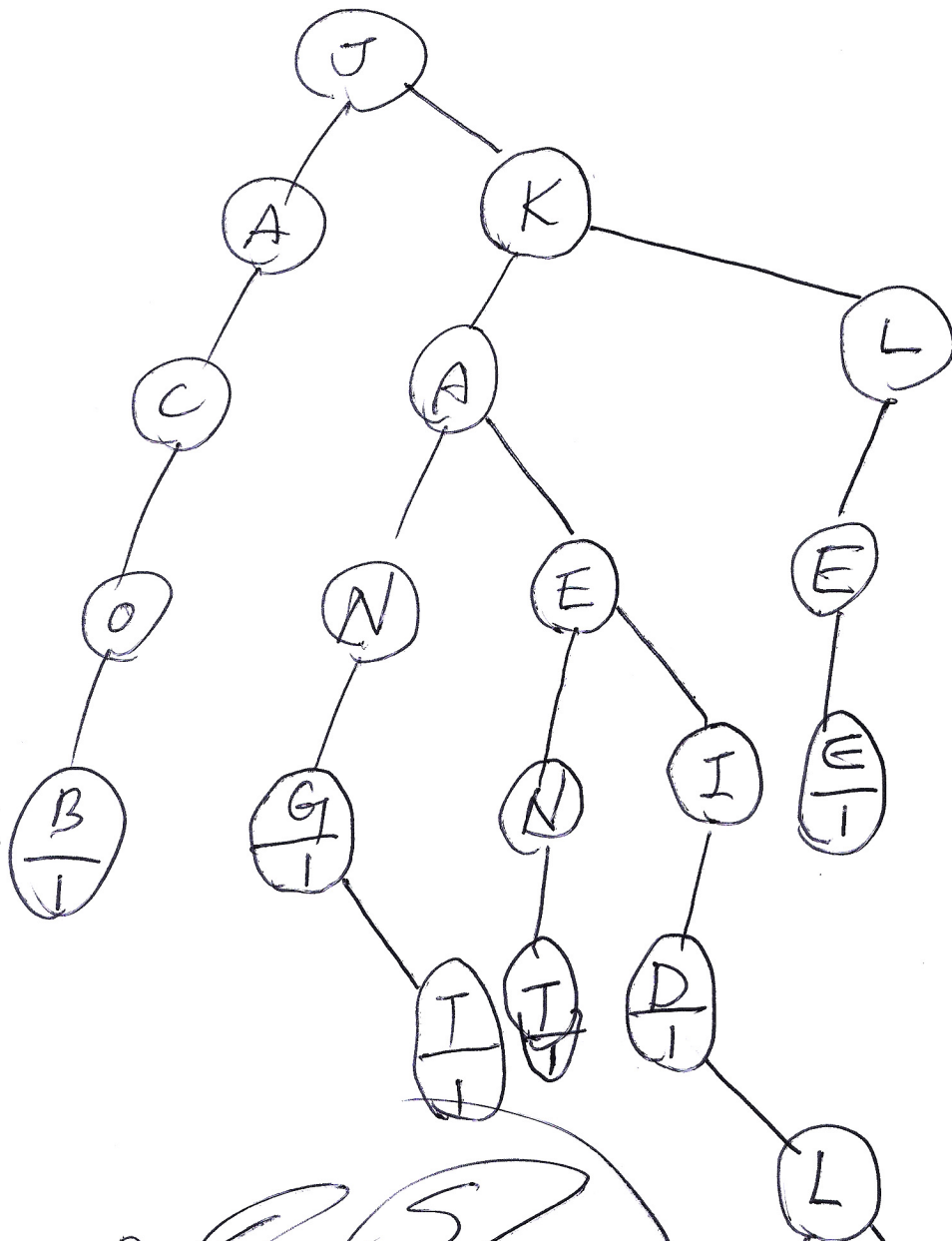
end;

end;



JACOB
 KANG
 KANT
 KENT
 KID
 KILE
 KIM
 KIMBERLY
 KIMMY
 KIN
 KING
 KINS
 KITE
 KITE
 KITE
 LEE

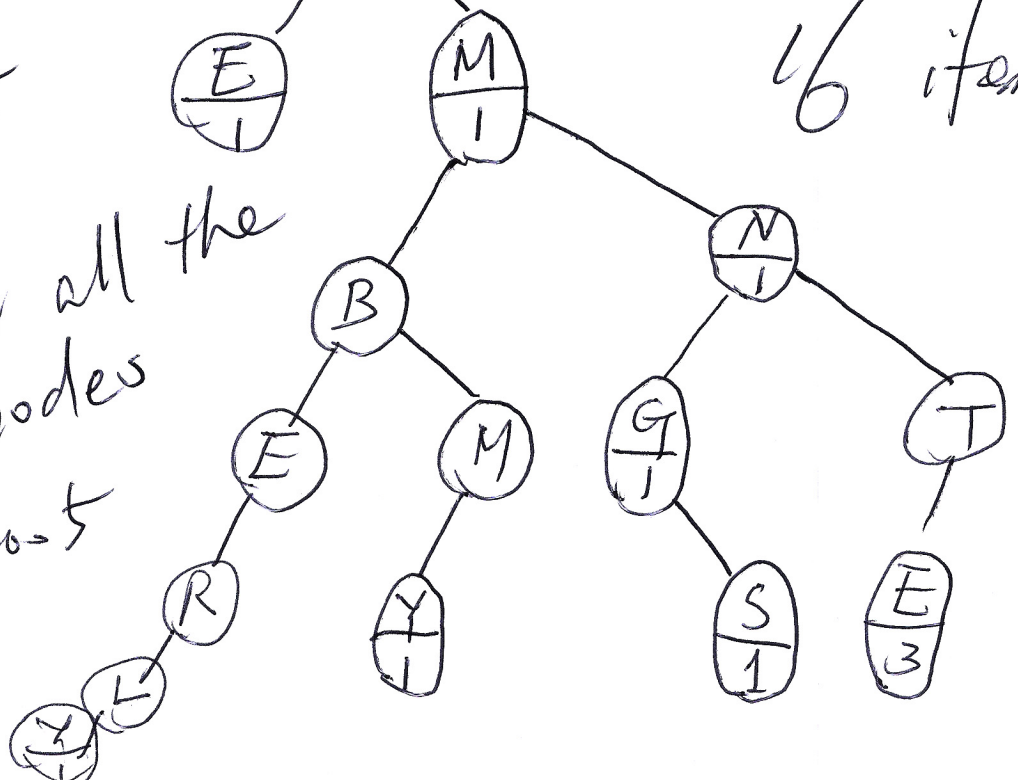
16 items



[Signature]

7/15/2005

Add "1" in all the
 terminal nodes
 July 15, 2005



Hwa Shin Building, Suite 701
705-22 Yuksam-dong, Kangnam-gu
Seoul 135-080
Republic of Korea

www.dkshin.com

from the laboratory of:

Dr. Dong-Keun Shin

Date: 12/26/2007

Del-Node-Record = record

Node_ptr :
searched :
end;

Del-Node-Array =

array [1..MAX_NAME] of
Del-Node-Record

이전 counter > 0?

(Lft) is tree node

Lft Son = Nil

이런 바로 된 pt (searched key)

Rgt Son = Nil

로 올라가 그 pt 를 Nil 로 만들고
2이전부터 searched key 의 한칸씩을 올라감.

Tree Node로

Lft Son = Nil

이런 바로 된으로 올라가

Rgt Son ≠ Nil

pt := its Rgt Son 으로 만들고
searched key 의 한칸씩을 올라감.

(처음일때) counter = counter - 1
if counter > 0

Left Son ≠ Nil

이런 그냥 내려서 두는 가능하면

Rgt Son = Nil

searched key 의 한칸씩을 올라감.

Left Son ≠ Nil

Rgt Son ≠ Nil

이런 그대로 놔두고 가능하면
searched key 의 한칸씩을
올라감.

12/26/2007