$i \backslash j$	0	1	2	3	4		k
0	1						
1	1	1					
2	1	2	1				
3	1	3	3	1			
n 4	1	4	6	4	1		
	$\vdots$					$\ddots$	1
$n-1$	1						
n	1						

$\emptyset$

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$

$\binom{n-1}{k-1}$ $\binom{n-1}{k}$

$\binom{n}{k}$

$O(nk)$ if $k = O(n)$

$\downarrow$

$O(n^2)$

V_{ij} = value for the optimal knapsack involving elements $a_1 \dots a_i$ with a weight capacity j

Inputs

items: $a_1 \dots a_n$

wts: $w_1 \dots w_n$

values v_1 v_n

capacity: W

Final Problem Solution:

$$V_{\boxed{n}\,\boxed{W}}$$

$V_{10,100}$

$= ?$

$n = 200$

$W = 1000$

$i = 0, 1, 2, \ldots n$

$j = 0 \ldots W$

Item	Weight	Value
a_1	5	10
a_2	2	5
a_3	4	8
a_4	2	7
a_5	3	7

$W = 7$ $n = 5$

V_{ij} = value for the optimal knapsack involving elements $a_1 \ldots a_i$ with a weight capacity j

take no elements

$i = 0, 1, 2, \ldots n$

take everything

$j = 0, 1, 2, \ldots W$

choose nothing

full capacity

Base Cases:

$i = 0$ value $= 0$

$j = 0$ value $= 0$

final optimal value:

$V_{n,W}$

V_{ij} = you are now considering including or excluding item a_i (wt: w_i value: v_i)

- is taking a_i even feasible

$j - w_i \geq 0$ feasible → take a_i / → don't take a_i

$j - w_i < 0$ infeasible → no choice, you cannot take a_i

$\underbrace{a_1\ a_2 \cdots a_{i-1}}$ a_i

don't take a_i

$\Rightarrow$ wt capacity is still j

$V_{i-1,j}$ = value of the optimal solution involving $a_1 \cdots a_{i-1}$, wt $= j$

take a_i:

$j \longrightarrow j - w_i$ (remaining capacity)

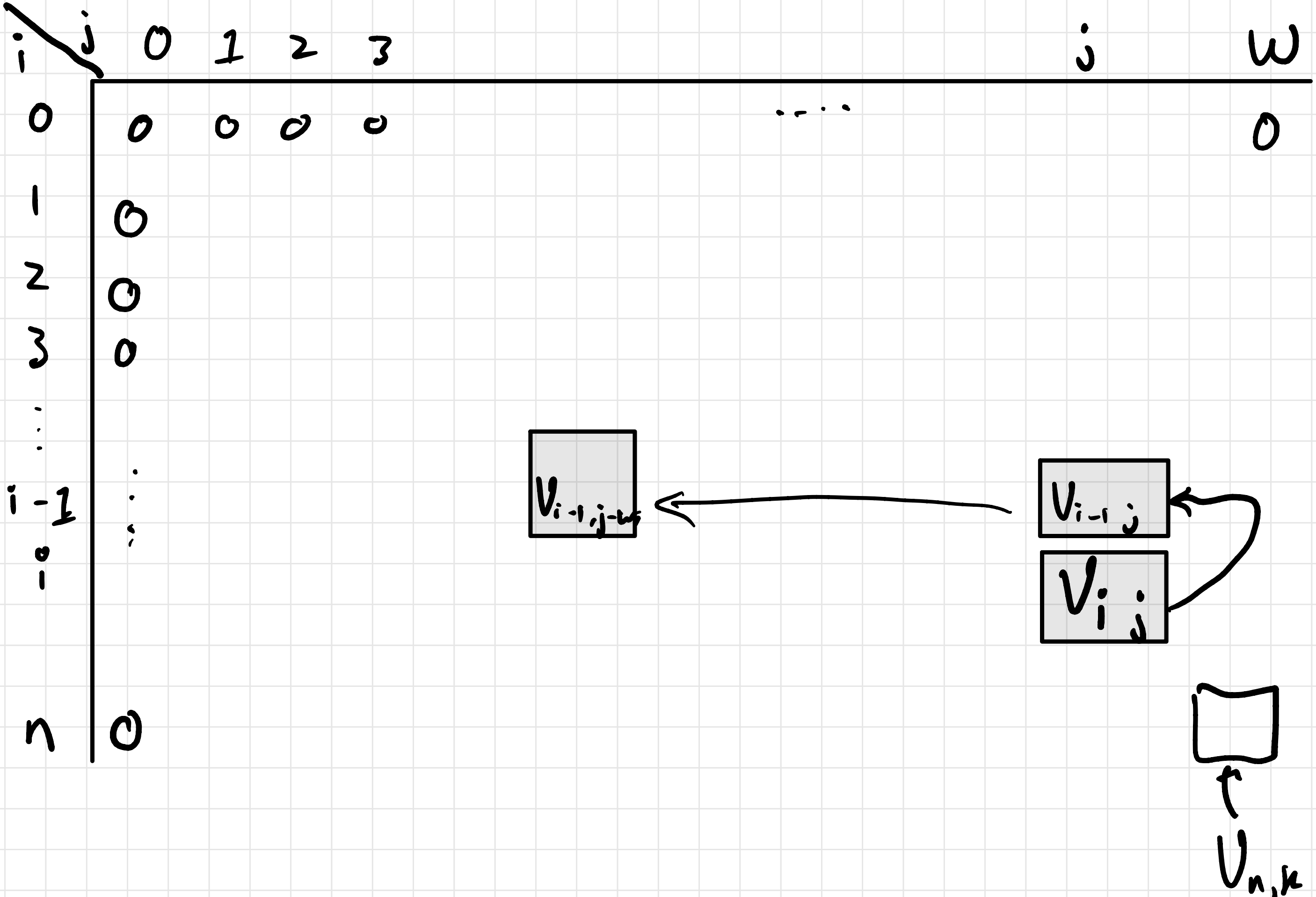
$a_1 a_2 \cdots a_{i-1} \boxed{a_i}$

$V_{i-1,\, j-w_i} + v_i$

get the value of a_i

$$V_{ij} = \begin{cases} \max\{\underbrace{V_{i-1,j} \quad V_{i-1,j-w_i}}_{\text{choice}}\} & \text{if } j-w_i \geq 0 \\ V_{i-1,j} & \text{if } j-w_i < 0 \end{cases}$$

no choice (infeasible)


i
j
0 1 2 3 j W
0
0 0 0 0 0
1
2
3
i-1
i
n
0
0
0
0
$V_{i-1,j-w}$
$V_{i-1,j}$
$V_{i,j}$
$V_{n,k}$

for $i = 0 \dots n$

for $j = 0 \dots W$

if $i = 0$ or $j = 0$

$V_{ij} \leftarrow 0$

else if $j - w_i < 0$

$V_{ij} \leftarrow V_{i-1,j}$

else if $V_{i-1,j} > V_{i-1,j-w_i} + v_i$ value of the a_i item

$V_{ij} \leftarrow V_{i-1,j}$ Don't take a_i

else

$V_{ij} \leftarrow V_{i-1,j-w_i} + v_i$

$i \backslash j$	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	10	10	10
2	0	0	5	5	5	10	10	15
3	0	0	5	5	8	10	13	15
4	0	0	7	7	12	12	15	17
5	0	0	7	7	12	14	15	19

Item	Weight	Value
a_1	5	10
a_2	2	5
a_3	4	8
a_4	2	7
a_5	3	7

$V_{4,5} =$ 1) feasible? $j - w_i = 5 - 2 \geq 0$ feasible

max: take it: $V_{i-1,\, j-w_i} + v_i$

$= V_{3\,3} + v_4 = 5 + 7 = 12$

or omit

$V_{i-1,j} = V_{3,5} = 10$

$i \backslash j$	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	10	10	10
2	0	0	5	5	5	10	10	15
3	0	0	5	5	$\boxed{8}$	10	13	15
4	0	0	7	7	$\boxed{12}$	12	15	17
5	0	0	7	7	12	14	15	$\boxed{19}$

Item	Weight	Value
a_1	5	10
$\boxed{a_2}$	$\boxed{2}$	5
~~a_3~~	4	8
$\boxed{a_4}$	$\boxed{2}$	7
$\boxed{a_5}$	$\boxed{3}$	7

≠ ≠ ≠

≠ => take a_5

→ wt => 4

= => did not take a_3

=> take a_4

=> take a_2

Solution:

$\{a_2\ a_4\ a_5\}$

Val: 19 wt: 7

Input: Tableau V of 0-1 knapsack

Output: Optimal knapsack $S \subseteq A = \{a_1 \cdots a_n\}$

$S \leftarrow \emptyset$

$i \leftarrow n$ // start at the bottom

$j \leftarrow W$ // start at the last column.

while $i \geq 0$ and $j \geq 0$ // stay in bounds of the table

while $i \geq 1$ and $V_{ij} = V_{i-1,j}$:

$i--$

$S \leftarrow S \cup \{a_i\}$

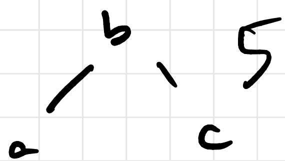
$j = j - w_i$

$i = i - 1$

Output S

Keys	probs	
a	.6	→ 60% you search for a
b	.3	
c	.1	→ 10%

c



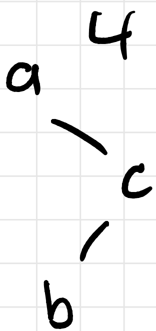
prob		# comps
.6	x	2
.3	x	1
.1	x	2

1.7

key comparisons on average (expected # of comparisons)

a	.6
b	.3
c	.1

$a < b < c$

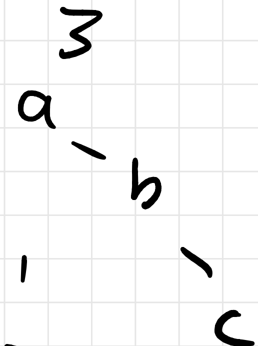


prob		# comps
.6	x	1
.3	x	3
.1	x	2

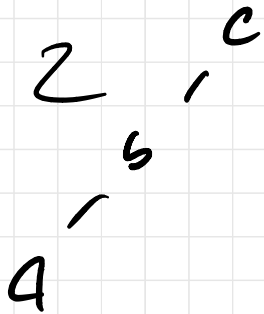
1.7

.6	x	1
.3	x	2
.1	x	3

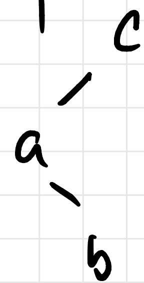
1.5



2



1



Catalan Numbers

$$C_n = \frac{1}{n+1}\binom{2n}{n} \in O\left(\frac{4^n}{n^{1.5}}\right)$$

= number of possible BSTs with n keys

$$n = 3 \qquad = \frac{1}{4}\binom{6}{3}$$

$$= \frac{1}{\cancel{4}} \cdot \frac{6!}{3!\,3!} = \frac{\cancel{6}\;5\;\cancel{4}}{\cancel{6}} = 5$$

C_{ij} = expected number of comps for OBST involving keys $k_i \ldots k_j$

- Full solution: $C_{1,n}$ = OBST $k_1 \ldots k_n$
- i, j will run from --- $0 \ldots n$
- C_{11} = OBST $k_1 \ldots k_1 = k_1$

k_1 cost = 1 comparison $\times p_1$

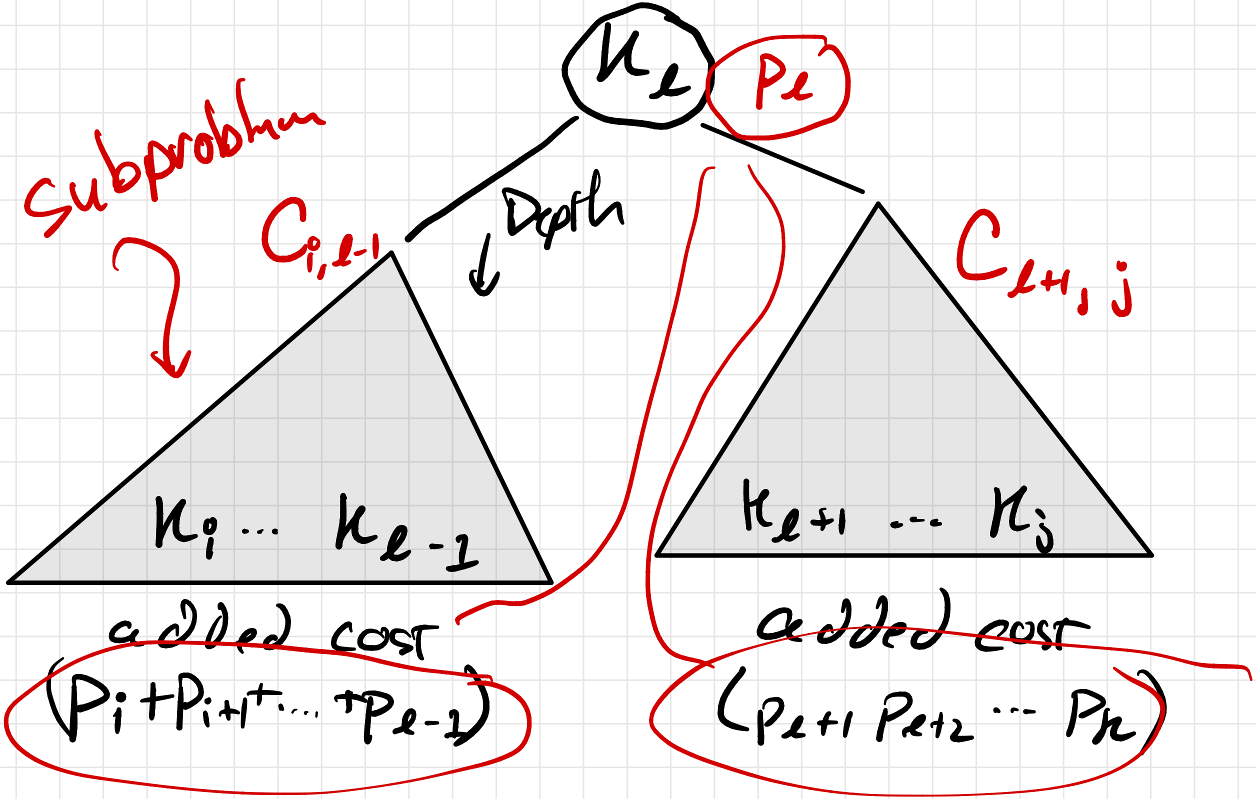
C_{22} C_{33} ...

- C_{ii} = OBST of only K_i = p_i

- C_{00} = empty tree = 0

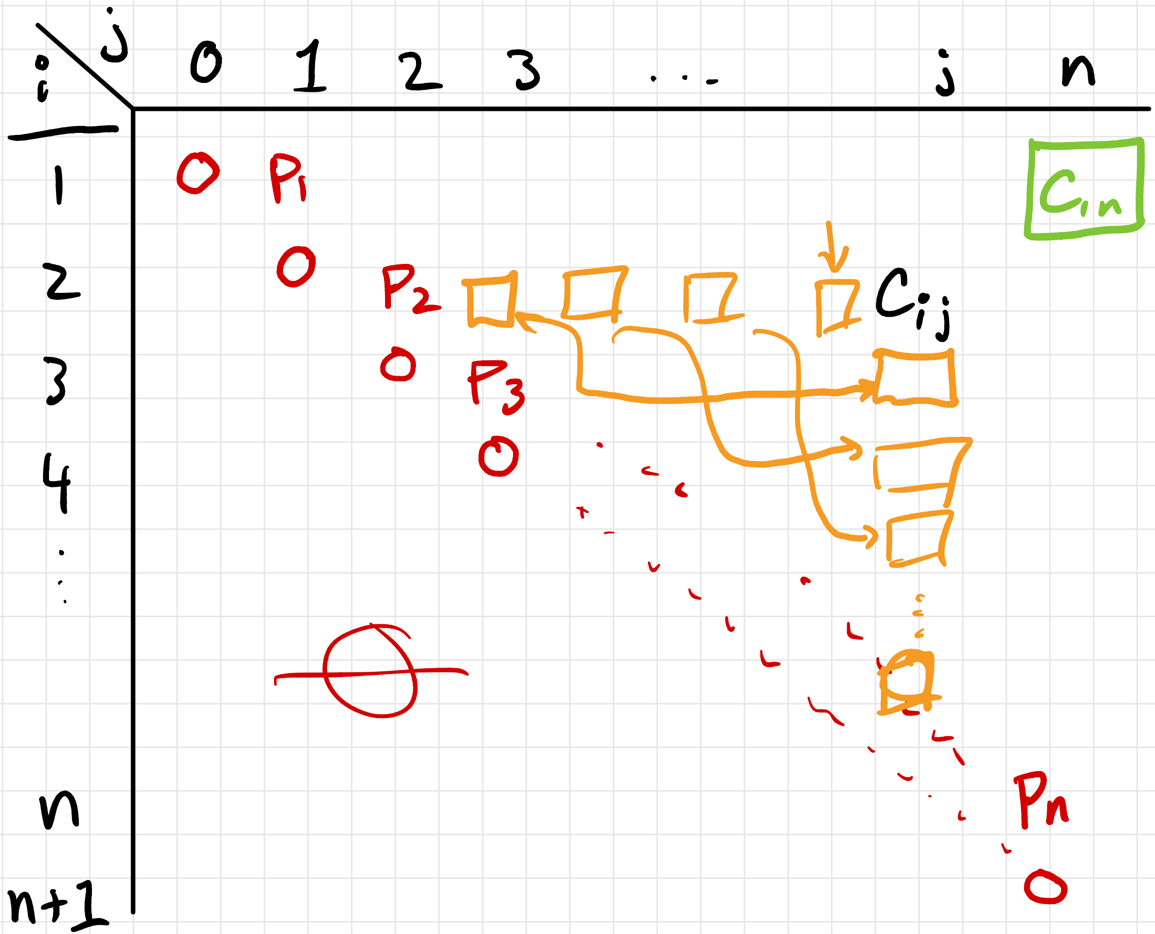
- C_{10} = $K_1 \dots K_0$ = empty tree = 0

C_{ij} : $k_i \cdots k_j$ make k_ℓ the root $\ell = i, i+1, \ldots j$



$$\sum_{m=i}^{j} p_m$$

$$C_{ij} = \min_{i \le l \le j} \left\{ C_{i,l-1} + C_{l+1,j} + \sum_{s=i}^{j} p_s \right\}$$



i
j
0
1
2
3
...
j
n
1
2
3
4
n
n+1
0
P1
0
P2
0
P3
0
Pn
0
Cij
Cin

$$
\begin{bmatrix}
 & & & & & & & \vdots & \\
\cdots & C_{i,i-1} & C_{i,i} & C_{i,i+1} & \cdots & C_{i,j-2} & C_{i,j-1} & \boxed{C_{i,j}} & \cdots \\
 & & & & & & & C_{i+1,j} & \\
 & & & & & & & C_{i+2,j} & \\
 & & & & & & & C_{i+3,j} & \\
 & & & & & & & \vdots & \\
 & & & & & & & C_{j,j} & \\
 & & & & & & & C_{j+1,j} & \\
 & & & & & & & \vdots &
\end{bmatrix}
$$

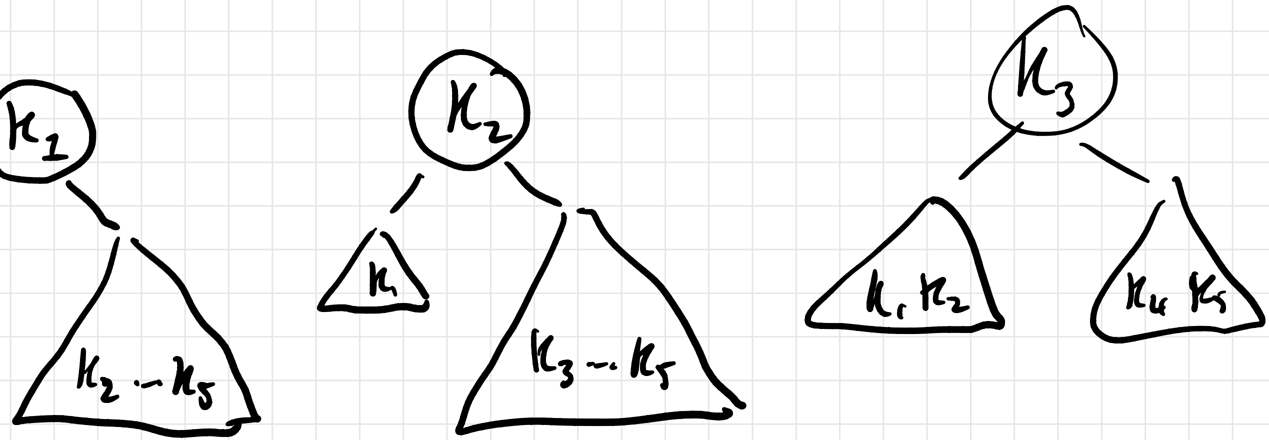
Algorithm 37: Optimal Binary Search Tree

INPUT : A set of keys $k_1, \ldots, k_n$ and a probability distribution p on the keys
OUTPUT : An optimal Binary Search Tree

```
FOR i = 1, ..., n DO                      initialization
    C[i,i-1] ← 0                          empty tree = no cost
    C[i,i] ← p(k_i)                       single node trees = p_i
    [R[i,i] ← i]                          Root Table keeps track of the node that gave us the min
C[n+1,n] ← 0
FOR d = 1, ..., (n − 1) DO
    FOR i = 1, ..., (n − d) DO
        j ← i + d
        min ← ∞
        FOR ℓ = i, ..., j DO              each n_ℓ is our root
            q ← C[i,ℓ-1] + C[ℓ+1,j]
            IF q < min THEN
                min ← q
                R[i,j] ← ℓ
        C[i,j] ← min + Σ_{s=i}^{j} p(k_s)
output C[1,n], R
```

$$C_{i,i-1} \leftarrow 0,\quad C_{i,i} \leftarrow p(k_i),\quad \boxed{R_{i,i} \leftarrow i},\quad C_{n+1,n} \leftarrow 0,\quad C_{i,j} \leftarrow min + \sum_{s=i}^{j} p(k_s)$$

$K_1 \dots 5$



$O(n^2)$

$i \backslash j$	0	1	2	3	4	5
1	0	0.213	0.253	1.033	$\boxed{1.233}$	1.573
2		0	0.020	0.587	0.787	1.127
3			0	0.547	0.747	1.087
4				0	0.100	0.320
5					0	0.120
6						0

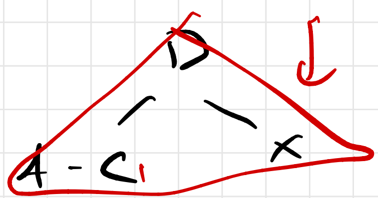
$i \backslash j$	1	2	3	4	5
1	1	1	3	3	3
2		2	3	3	3
3			3	3	3
4				4	5
5					5

	Key	Probability
1	A	.213
2	B	.020
3	C	.547
4	D	.100
5	E	.120

A B C D

$l = 1, 2, 3, 4$

$$C_{ij} = \min_{i \le l \le j} \left\{ C_{i,l-1} + C_{l+1,j} + \sum_{s=i}^{j} p_s \right\}$$



$$
\begin{array}{r}
.88 \\
+\ .353 \\
\hline
1.233
\end{array}
$$

~~$l = 1 \quad C_{1,0} + C_{24} = .787$~~

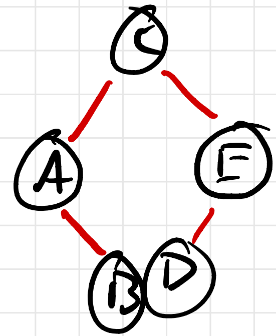
~~$l = 2 \quad C_{1,1} + C_{34} = .213 + .744$~~

$l = 3: \quad C_{12} + C_{4,4} = .253 + .1$

~~$l = 4: \quad C_{13} + C_{54} = 1.033 + 0$~~

4	5
33	1.573
37	1.127
47	1.087
00	0.320
0	0.120
	0

$i \backslash j$	1	2	3	4	5
1	1	1	3	3	3
2		2	3	3	3
3			3	3	3
4				4	5
5					5



Key	Probability
A	.213
B	.020
C	.547
D	.100
E	.120

key	prob	#comps		
A	.213	× 2	=	.426
B	.020	× 3		.060
C	.547	× 1		.547
D	.100	× 3		.300
E	.120	× 2		.240

.486

1.087

1.573

Input: keys $k_1 \cdots k_n$, root table R $O(n)$

Output: root node of the built OBST

root ← new node

root.key ← $R_{1,n}$

S ← init a stack

S.push(root, 1, n)

node, i, j
↑ root of keys $k_i \cdots k_j$

while S is not empty

(u, i, j) ← S.pop()

k ← R_{ij} // key of u

if k < j // build Right tree...

v ← new node

v.key ← $R_{k+1,j}$

u.rightChild ← v

S.push(v, k+1, j)

$u \longrightarrow key = K_k$

$K_i - - K_j$

$k < j \Rightarrow$ is a right

$= \Rightarrow$ no child

Right child

$$K_k = K_j$$

⋮

if $i < k$

// Build the left Child

$v \leftarrow$ new node

$v.key \leftarrow R_{i,k-1}$

$u.leftChild \leftarrow v$

$S.push(v, i, k-1)$

output root