Wednesday, August 17, 2022 12:26 PM

10, 5, 20, 15, 25, 3, 4, 0

10

5 } 10

5 } 10, 20

5 } 10, 15, 20

5, 10 } 15, 20

5, 10 | 15, 20, 25

3, 5, 10 | 15, 20, 25

3, 4, 5, 10 } 15, 20, 25

0, 3, 4, 5, 10 } 15, 20, 25

0, 3, 4, 5 | 10, 15, 2, 25

Hash map

Array

Key index 0 to n-1

Value arr[idx]

Key <String, Integer>

Chammi = 10

Bittu = 20

Pintu = ~~18~~ 15.

Kakke = 12

← O(1)

A - 10

B - 20

C - 30

D - 40

put (O(1))

	0	1	2	3	
K	A	B	C	D	
V	10	20	30	40	

map.get(C)

```
int[] arr1 = { 30, 20, 40, 50, 70, 30, 20, 20, 50, 50 };
        int[] arr2 = { 50, 80, 30, 20, 20, 20, 90, 50, 20 };
```

50 - ~~1~~ ~~2~~ 1

80 - 1

30 - ~~1~~ 0

20 - ~~4~~ 3

90 - 1

30

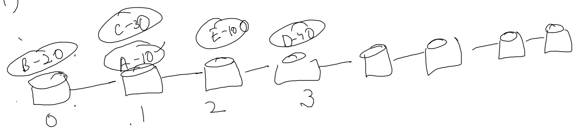
20

50

f^n (Key)

f^n (A) ⇒ 1

Avg Bucket size factor



a f^n(B) → 0

map.get(C)

2->

AL

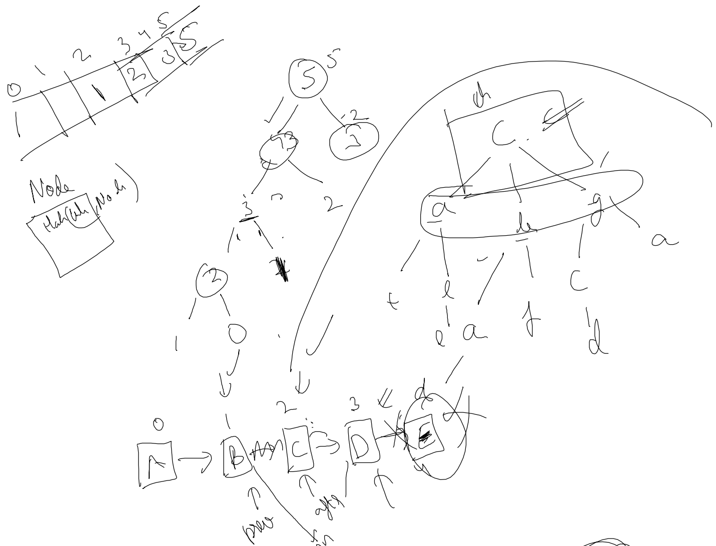
f^n(a - C)

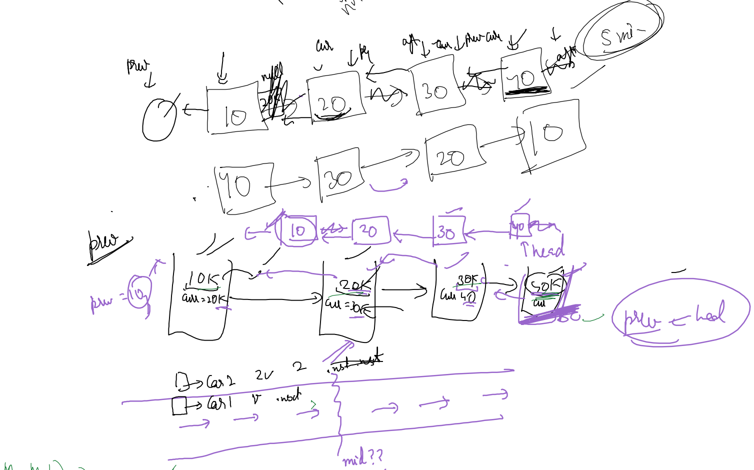
value

$$fib(n) = fib(n-1) + fib(n-2)$$

$$T(n) = T(n-1) + T(n-2) + 1$$

$$2^n$$

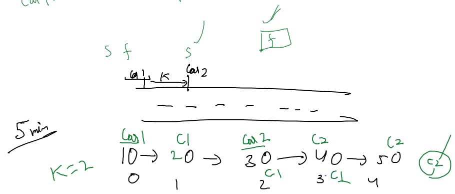




M1) → LL → copy new] O(n)

M2 → car 2 stack car 1] O(n)

M-3) → O(n) ~~X~~ no extra space



K=1 → 50

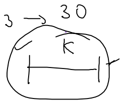
K=2 → 40

K=3 → 30

K=4 → 20

K=5 → 10

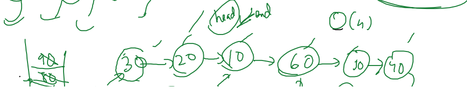
size

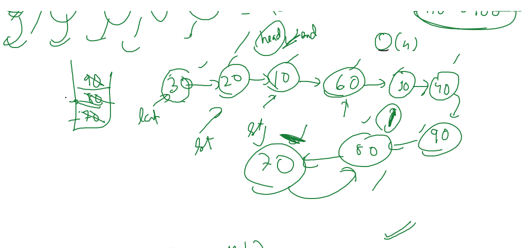


k=3

10 -> 20 -> 30 -> 40 -> 50 -> 60 -> 70 -> 80 -> 90 -> 100 -> 110

30 -> 20 -> 10 -> 60 -> 50 -> 40 -> 90 -> 80 -> 70 110 -> 100

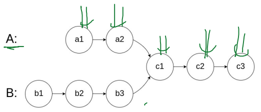




min stack

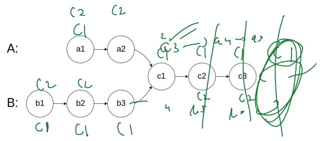
M1) n x m

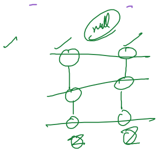
M2) → Treat visited
n+m
Node

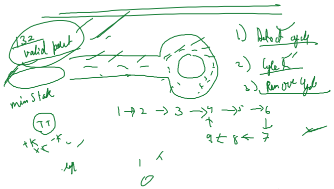


M3) → CountA lenA, CountB lenB
n+m
(0, lenB-lenA)

M-4
n+m

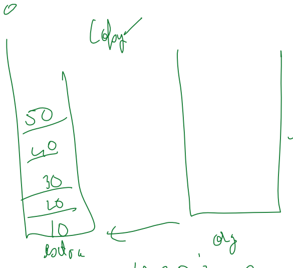






1) Detect cycle
2) Cycle B
3) Remove cycle

1 → 2 → 3 → 4 → 5 → 6
9 ← 8 ← 7



10, 20, 30, 40, 50

M-1) Time O(n), Space O(n)

10, 20, 30, 40, 50

A → B

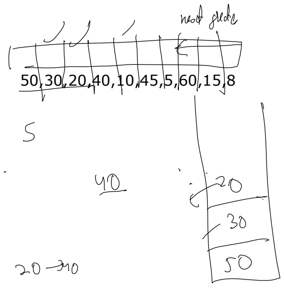
50,30,20,40,10,45,5,60,15,8

50 → 60
30 → 40
20 → 40
40 → 45
10 → 45
45 → 60
5 → 60

60 → -1
15 → -1
8 → -1

next greater

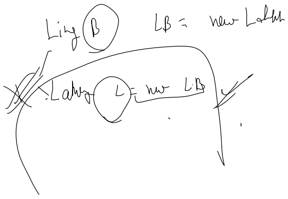
50,30,20,40,10,45,5,60,15,8

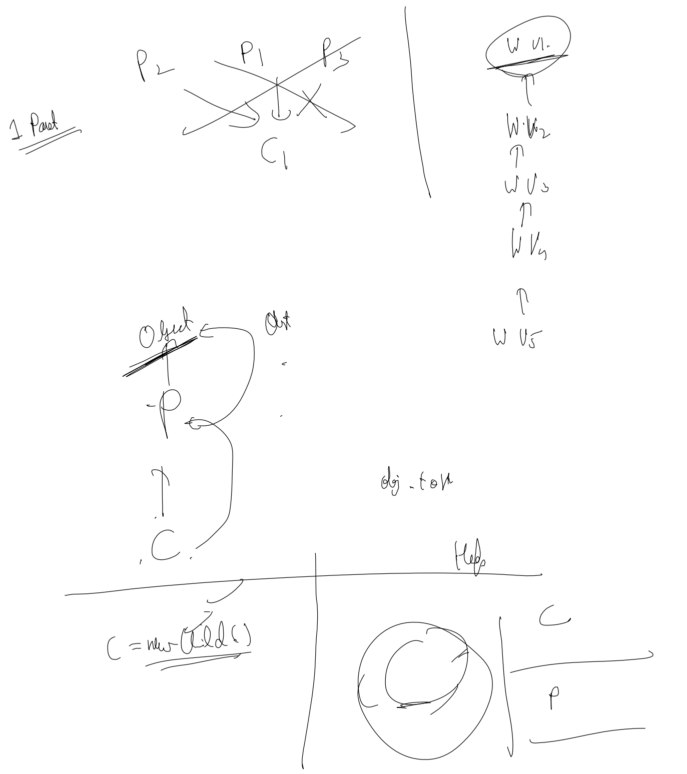


20 → 40
30 → 40

Ling B   LB = new Linkedlist ( )

~~Laby~~ L = new LB





DP

f^n parse <=> unique sub problem

T.D.

Hash < [ ], soln >

(,)

2^n

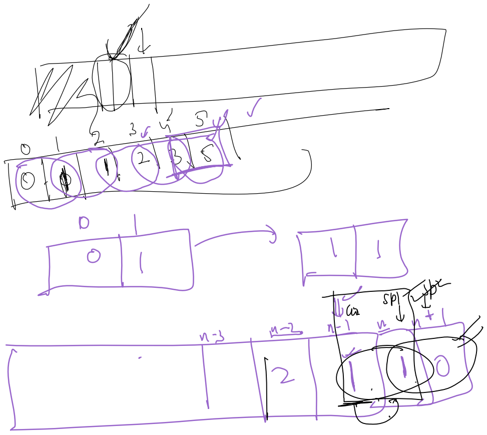
K^n

no. of unique state x self work

10,K

BU

Bottom Up



A=5

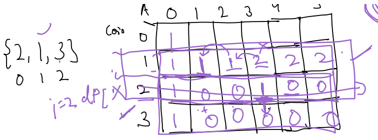
{2, 1, 3}
0 1 2

Coin \ A	0	1	2	3	4	5
0	1					
1	1	1	1	2	2	2
2	1	0	0	1	0	0

dp[

i, A

```
for (int i = coins.length - 1; i >= 0; i--) {
        for (int A = 0; A <= Amount; A++) {
                if (A == 0) {
                        dp[i][A] = 1;
                        continue;
                }
//                          i, A
```



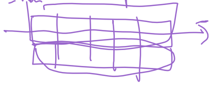
```
for (int i = coins.length - 1; i >= 0; i--) {
          for (int A = 0; A <= Amount; A++) {
                    if (A == 0) {
                              dp[i][A] = 1;
                              continue;
                    }
//                            i, A
                    int sp1 = 0;
                    if (A >= coins[i]) {
                              sp1 = dp[i][A - coins[i]];
                    }
                    int sp2 = dp[i + 1][A];
                    dp[i][A] = sp1 + sp2;
          }
}
```

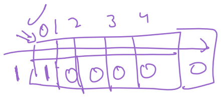
i = 1, A = 1   O(n)

dP[1] = dP[0]

dP[0] = ma

sp1 = DP[1][0] = 1

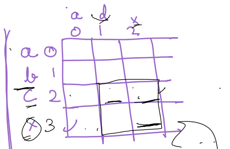




```
publicstaticintminDistance_BU(Stringword1, Stringword2) {
int[][] dp= newint[word1.length() + 1][word2.length() + 1];
for(inti1= word1.length(); i1>= 0; i1--) {
for(inti2= word2.length(); i2>= 0; i2--) {
// i1, i2
if(i1== word1.length() || i2== word2.length()) {
intL1= word1.length() - i1;
intL2= word2.length() - i2;
dp[i1][i2] = Math.max(L1, L2);
} elseif(word1.charAt(i1) == word2.charAt(i2)) {
dp[i1][i2] = dp[i1+ 1][i2+ 1];
} else{
intIns= dp[i1][i2+ 1];
intDel= dp[i1+ 1][i2];
intRep= dp[i1+ 1][i2+ 1];
dp[i1][i2] = Math.min(Del, Math.min(Ins, Rep)) + 1;
}
}
}
returndp[0][0];
}
```

From <https://github.com/Lakshya-CB/BML_Java22/blob/main/Lec_60_DP/Edit_distance.java>

```
public static int minDistance_BUSE(String word1, String word2) {
          int[][] dp = new int[word1.length() + 1][word2.length() + 1];
          for (int i1 = word1.length(); i1 >= 0; i1--) {
                    for (int i2 = word2.length(); i2 >= 0; i2--) {
//                            i1, i2
                              if (i1 == word1.length() || i2 == word2.length()) {
                                        int L1 = word1.length() - i1;
                                        int L2 = word2.length() - i2;

                                        dp[i1][i2] = Math.max(L1, L2);
                              } else if (word1.charAt(i1) == word2.charAt(i2)) {
                                        dp[i1][i2] = dp[i1 + 1][i2 + 1];
                              } else {
                                        int Ins = dp[i1][i2 + 1];
                                        int Del = dp[i1 + 1][i2];
                                        int Rep = dp[i1 + 1][i2 + 1];
                                        dp[i1][i2] = Math.min(Del, Math.min(Ins, Rep)) + 1;
                              }
                    }
          }
          return dp[0][0];
```



i1 → 3 to 0

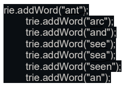
i2 → 2 to 0

We

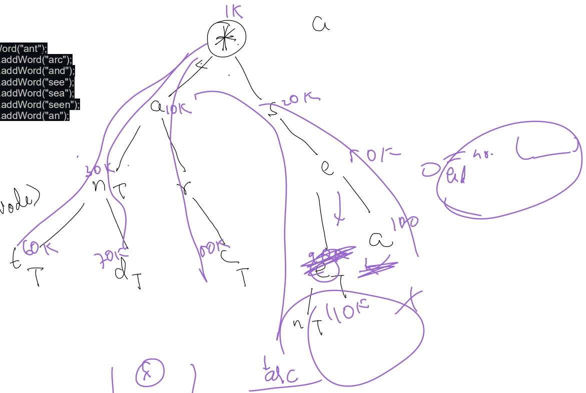
O 1

w1 "abc"

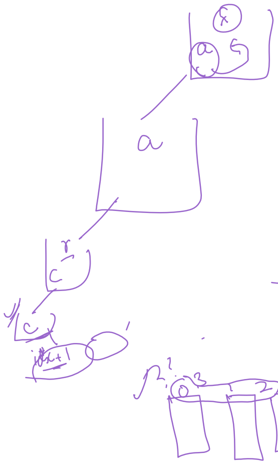
w2 "ad"



```
rie.addWord("ant");
          trie.addWord("arc");
          trie.addWord("and");
          trie.addWord("see");
          trie.addWord("sea");
          trie.addWord("seen");
          trie.addWord("an");
```



Node {
char ch;
HashMap<ch, Node>



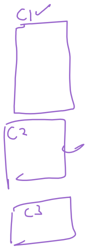
C1, 1

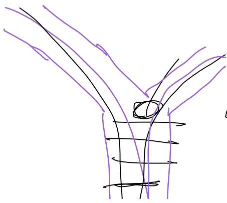
C1, 2

C1, 1C

C2, 10

→ C3, 11



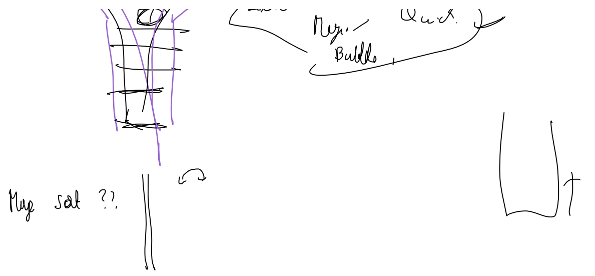


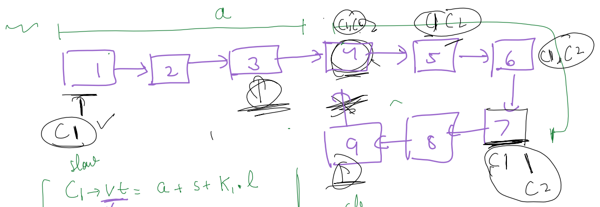
Select - In

Merge

Bubble

Quick





$C_1 \to \underline{vt} = a + s + K_1 \cdot l$ 2X

$C_2 \Rightarrow 2vt = a + s + K_2 \cdot l$

~~cle~~
~~rpt~~

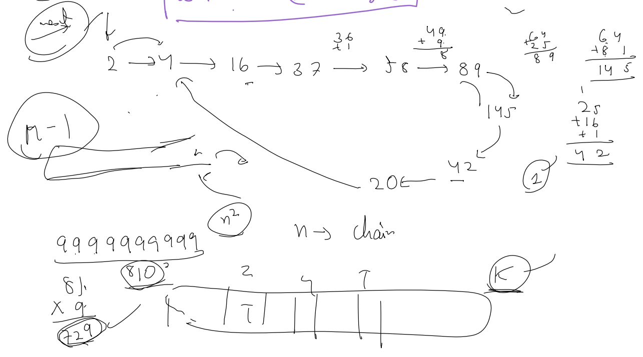
$\boxed{a+s} = c \cdot l$

$2(a+s+K_1 l) = (a+s+K_2 l)$

~~fast = next(next(fast))~~ ~~slow = next(slow)~~

$\boxed{a+s = (K_2 - 2K_1) l}$ ✓

L.L.



T/F

BS

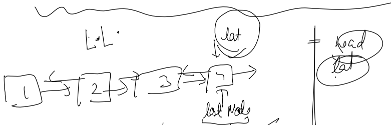
ith

L.L.

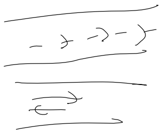
.next

cycle ??

L.L.

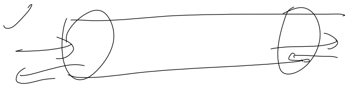


O(n)	Add last	~~O(n)~~	O(1) ✓
	Remove last	O(n)	O(n) → O(1)



L.L
doubly
head
last
prev . next

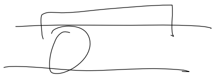
catch

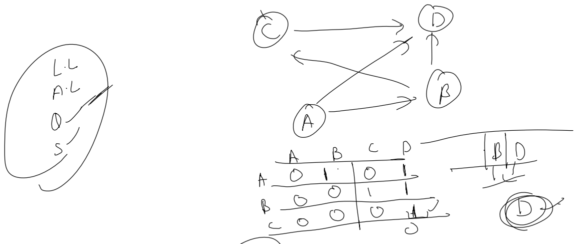


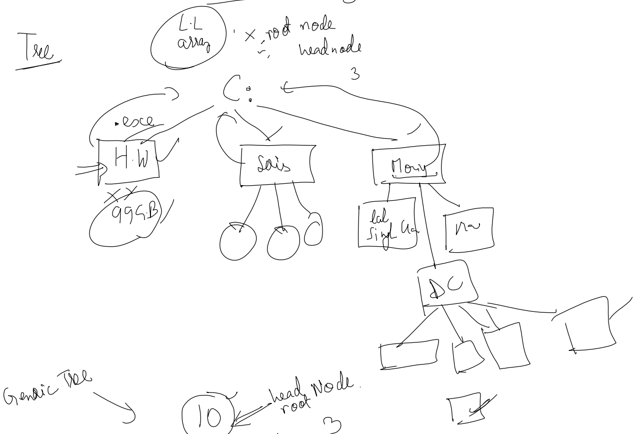
L.L.

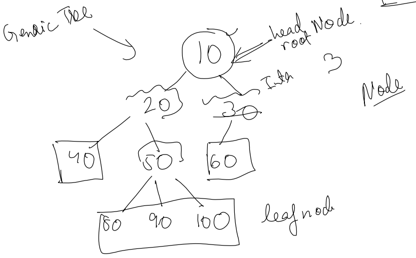
dabl

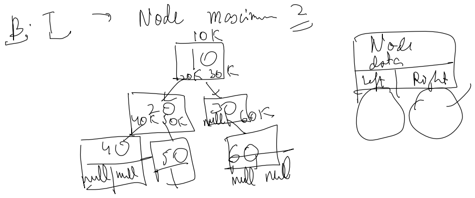
Tree
8:50 pm

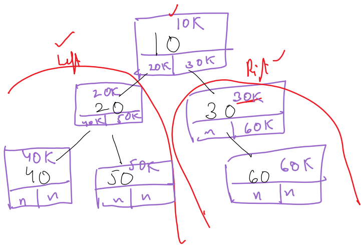






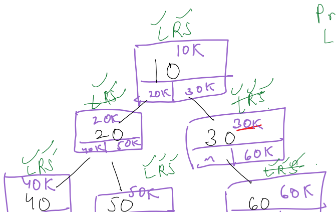






B : Print (10K)

SP : Prin (20K)
Print (30K)
Syso(10K)



Print (n.n)
L, R ⟸ S.L.R Pre order
BT

S L R

10, 20, 40, 50, 30, 60 ← Pre (S.L.R)

40, 20, 50, 10, 30, 60 ← In (L.S.R)

40, 50, 20, 60, 30, 10 ← Post (L.R.S)

3

H Ru

S.R.L

R S L

R L S

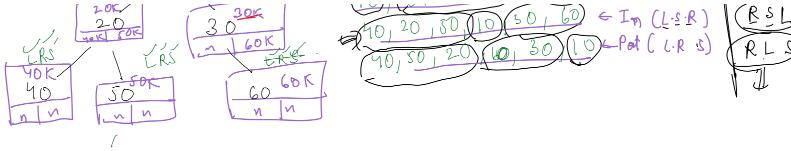
Traversals

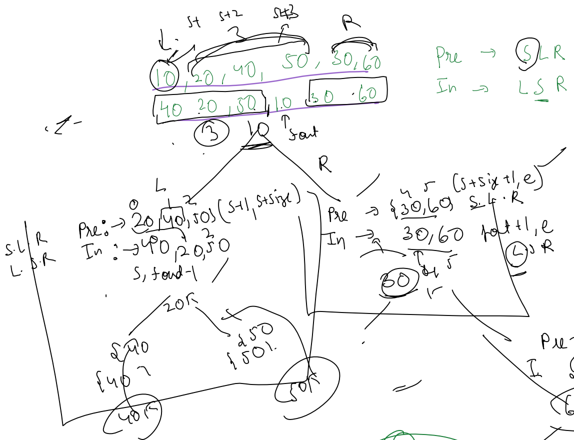
rder

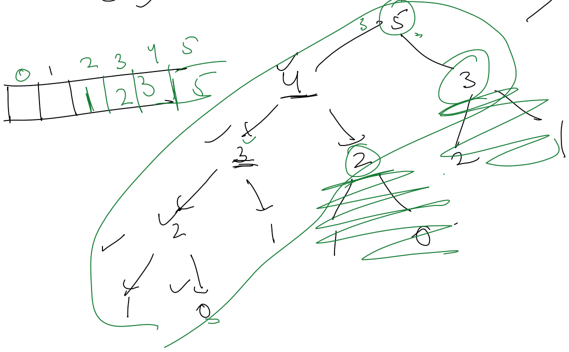
→ Post

→ In

→ Pre ✓







Fibo (n) → soln

0

n

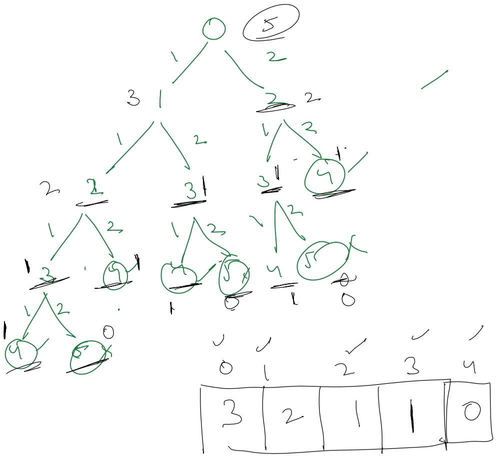
dP[i]

Fibo(i)

BP(n)

sp sp1 (n-1)

sp2 (n-2)



n = 3

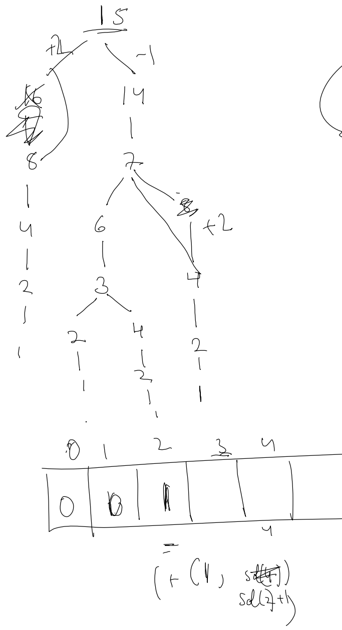
int
dp
for
// cl

}
ret

→ In

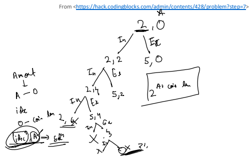
→ Prev

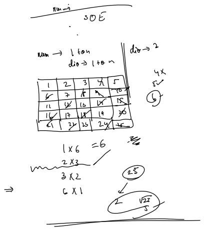
```
[] dp = new int[n+2];
[n]=1;
 (int curr = n-1; curr >= 0; curr--) {
imb(curr)
   int sp1 = dp[curr + 1];
   int sp2 = dp[curr + 2];
   dp[curr] = sp1 + sp2;

urn dp[0];
```



~~10 4~~ 5
2 5 3

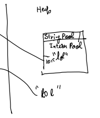
From <https://hack.codingblocks.com/admin/contents/428/problem?step=7>

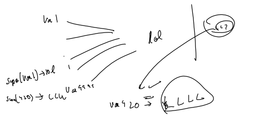




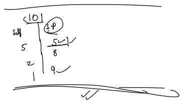
Heap

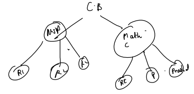
```
String str = "lol";
String str1 = "lol";
String str2 = "lol";

System.out.println(str==str2);

String str00 = new String("lol");
System.out.println(str00);
System.out.println(str==str00);
```

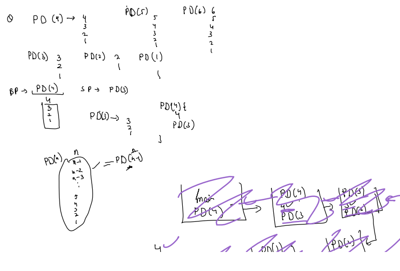




1) Bigger Problem.
2) Smaller Proble
3) ASsume SP
4) Bigger Solⁿ create

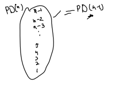


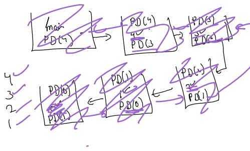




17

16





```
public static void PD(int n) {
//                    BP : PD(n)
//                    SP : PD(n-1)

        If(n===0){
        Return;
        }
                System.out.println(n);
                PD(n-1);
        }
```

Q PI(4)

1
2
3
4

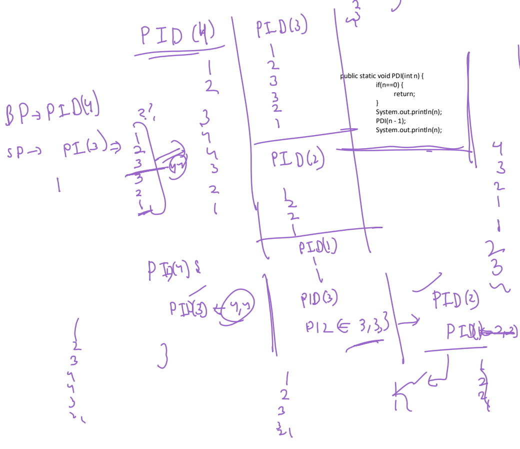
PDI (4)

4
3
2
1
1
2
3
4

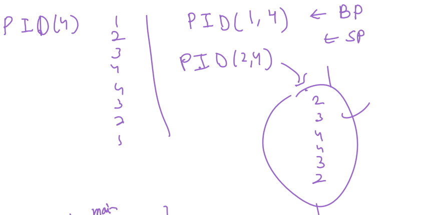
BP → PDI(4) →

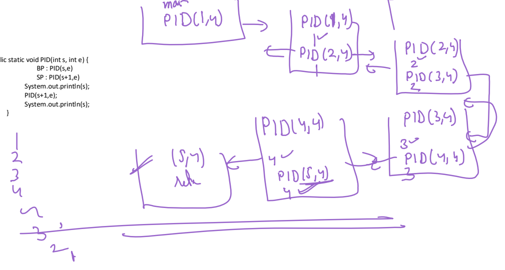
SP → PDI(3)

4
3
2
1
1
2
3
4



```
public static void PDI(int n) {
                if(n==0) {
                        return;
                }
                System.out.println(n);
                PDI(n - 1);
                System.out.println(n);
```





```
public static void PID(int s, int e) {
//                    BP : PID(s,e)
//                    SP : PID(s+1,e)
                System.out.println(s);
                PID(s+1,e);
                System.out.println(s);
        }
```

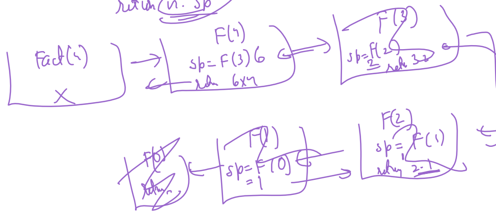
4x3 x 2 x 1

Fac(4) =

BP → Fact(n)

SP → Fact(n-1) = (n-1)! = sp

return n. sp



X10
x20
x30
x40
50

x1 x100

??

Q a^b

a^b / a^{b-1}

0 1 2 3 4 5 6 7 8

0 1 1 2 3 5 8 , 13 ,

PDI (3)

3
2
1
1
2
3

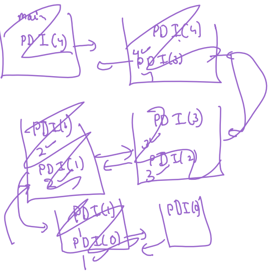
PDI(2)

2
1
1
2

PDI(1)

1
1

DI(4)
4
PDI(3)



~~a^b / a^{b-1}~~

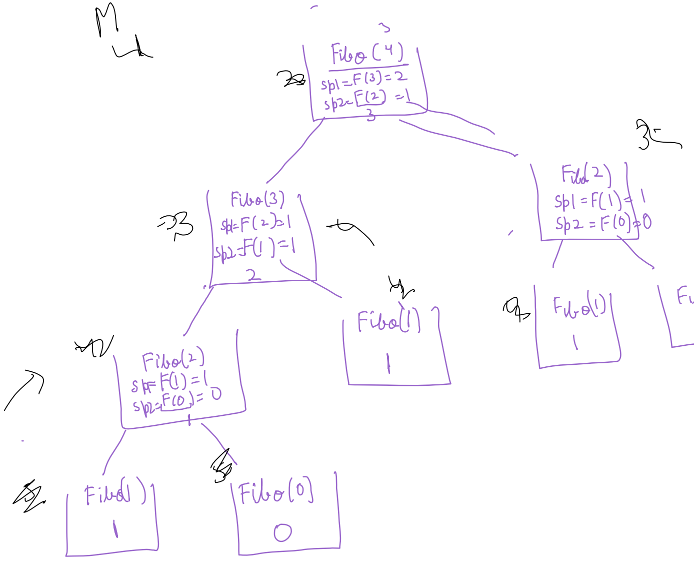
0 1 2 3 4 5 6 7 8

0, 1, 1, 2, 3, 5, 8, 13,

DP → Fibo(n)

sp → Fibo(n-1) = sp1

Fibo(n-2) = sp2



Quick Notes Page 21

$b_0(0)]$

0