Lecture 24 - Dec. 3

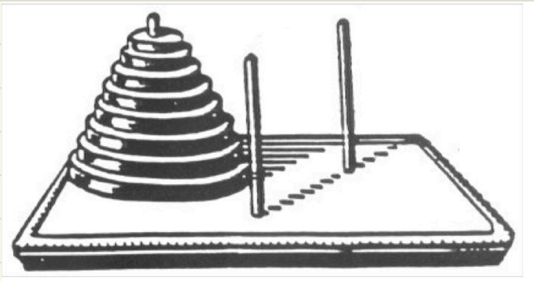
Recursion

Tower of Hanoi: Specification, Legend
Tower of Hanoi: Java, Tracing
Tower of Hanoi: Running Time

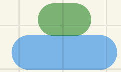
Announcements/Reminders

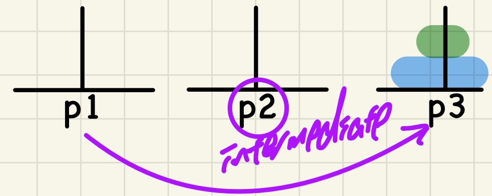
- **Lab5** due **midnight** today
  + Required study: **Abstract Classes** & **Interfaces**
- **ProgTest3** results released
- Extra office hours: 3pm to 5pm on Thursday
- **Exam** Review Session (Zoom): 3pm on Friday
- Materials for tutorial session on **recursion**

Tower of Hanoi: Strategy

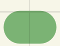


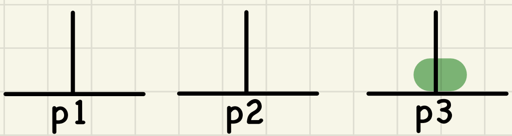
Consider 2 disks: A < B

move  from p1 to p3

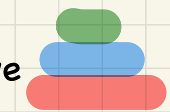


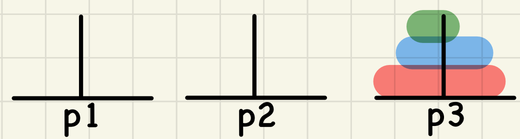
Consider 1 disk: A

move  from p1 to p3

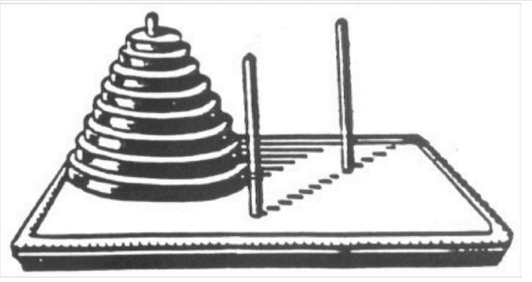


Consider 3 disks: A < B < C

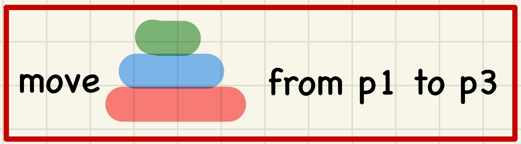
move  from p1 to p3

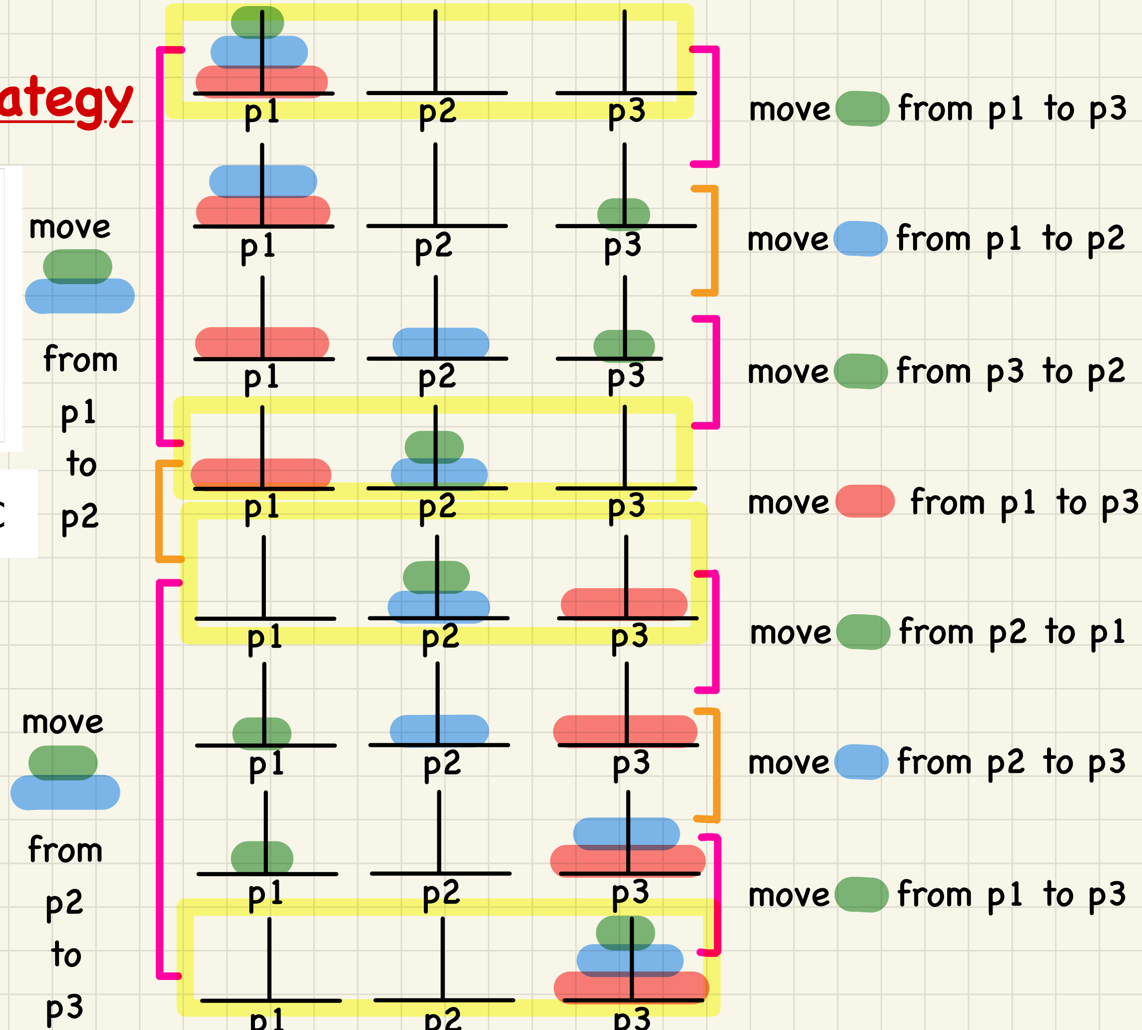


Tower of Hanoi: Strategy

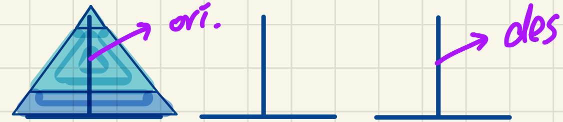


Consider 3 disks A < B < C





Tower of Honoi in Java




```
void towerOfHanoi(String[] disks) {
  tohHelper (disks, 0, disks.length - 1, 1, 3);
}
void tohHelper(String[] disks, int from, int to, int ori, int des){
  if(from > to) { }
  else if(from == to) {
    print("move " + disks[to] + " from " + ori + " to " + des);
  }
  else {
    int intermediate = 6 - ori - des;
    tohHelper (disks, from, to - 1, ori, intermediate);
    print("move " + disks[to] + " from " + ori + " to " + des);
    tohHelper (disks, from, to - 1, intermediate, des);
  }
}
```

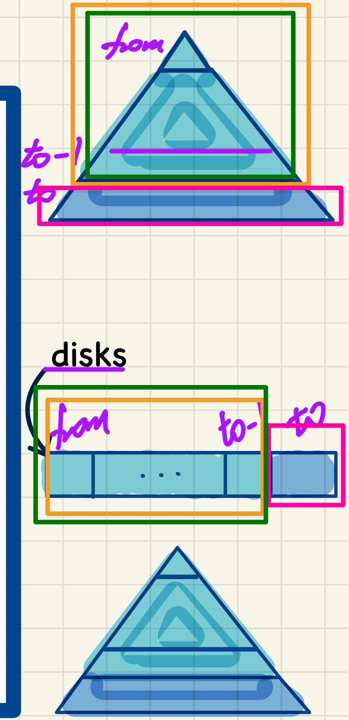
peg 1 peg 3

range of array/tower

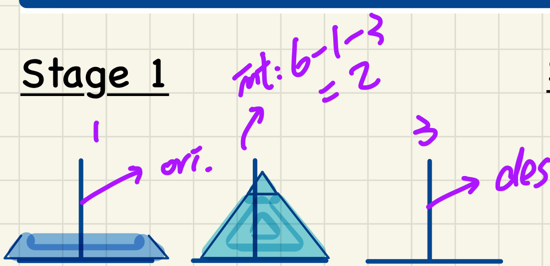
only one disk in the tower

p2 to p3 → p(6-2-3) → p1

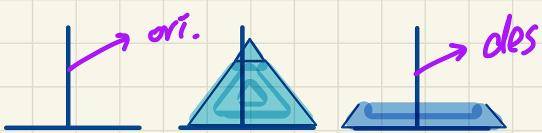
① ② ③



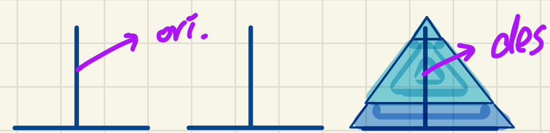
Stage 1



Stage 2



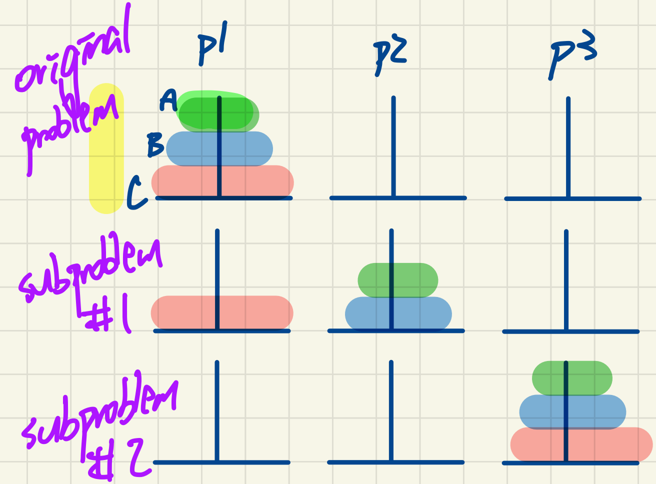
Stage 3



Tower of Hanoi: Tracing

ds →

0	1	2
A	B	C



tohH(ds, 0, 2, p1, p3)

subproblem #1

tohH(ds, 0, 1, p1, p2)

intermediate: p3

④ move C: p1 to p3

tohH(ds, 0, 1, p2, p3)

subproblem #2

① tohH(ds, 0, 0, p1, p3) → move A: p1 to p3

base case

② move B: p1 to p2

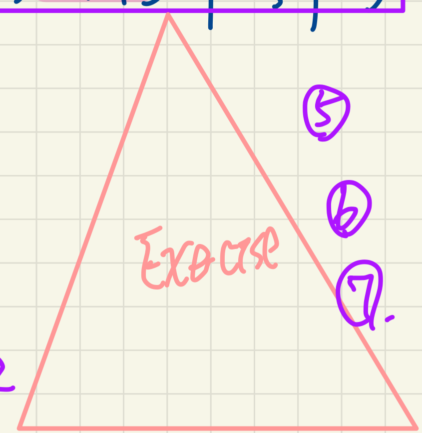
③ tohH(ds, 0, 0, p3, p2) → move A: p3 to p2

base case

⑤

⑥

⑦.



Tower of Hanoi: Tracing

Say ds (disks) is $\{A, B, C\}$, where $A < B < C$.

$$
tohH(ds, \underbrace{0,2}_{\{A,B,C\}}, p1, p3) = \begin{cases}
tohH(ds, \underbrace{0,1}_{\{A,B\}}, p1, p2) = \begin{cases}
tohH(ds, \underbrace{0,0}_{\{A\}}, p1, p3) = \begin{cases} \texttt{Move } \mathbf{A}\texttt{: p1 to p3} \end{cases} \\
\texttt{Move } \mathbf{B}\texttt{: p1 to p2} \\
tohH(ds, \underbrace{0,0}_{\{A\}}, p3, p2) = \begin{cases} \texttt{Move } \mathbf{A}\texttt{: p3 to p2} \end{cases}
\end{cases} \\
\texttt{Move } \mathbf{C}\texttt{: p1 to p3} \\
tohH(ds, \underbrace{0,1}_{\{A,B\}}, p2, p3) = \begin{cases}
tohH(ds, \underbrace{0,0}_{\{A\}}, p2, p1) = \begin{cases} \texttt{Move } \mathbf{A}\texttt{: p2 to p1} \end{cases} \\
\texttt{Move } \mathbf{B}\texttt{: p2 to p3} \\
tohH(ds, \underbrace{0,0}_{\{A\}}, p1, p3) = \begin{cases} \texttt{Move } \mathbf{A}\texttt{: p1 to p3} \end{cases}
\end{cases}
\end{cases}
$$

Tower of Hanoi: Running Time

$T(n) = ?$ $\quad T(n-\boxed{?}) \rightarrow n-1$

$= T(1)$

Running Time as a Recurrence Relation

```
void towerOfHanoi(String[] disks) {
  tohHelper(disks, 0, disks.length - 1, 1, 3);
}
void tohHelper(String[] disks, int from, int to, int ori, int des) {
  if(from > to) { }
  else if(from == to) {
    print("move " + disks[to] + " from " + ori + " to " + des);
  }
  else {
    int intermediate = 6 - ori - des;
    tohHelper(disks, from, to - 1, ori, intermediate);
    print("move " + disks[to] + " from " + ori + " to " + des);
    tohHelper(disks, from, to - 1, intermediate, des);
  }
}
```

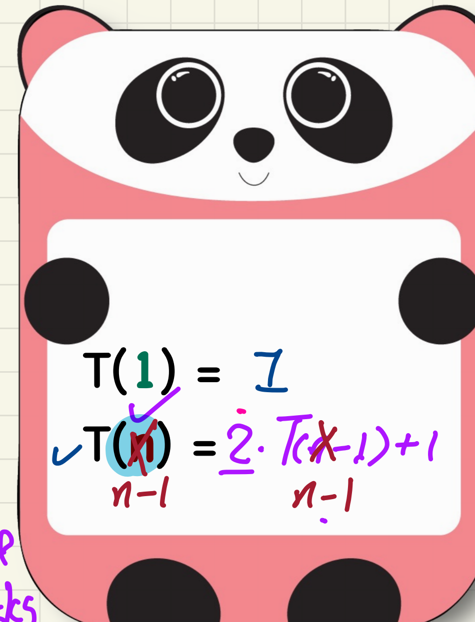
$T(n)$

n disks

$T(n-1)$

1.

$T(n-1)$



$$T(n) = 2\cdot \boxed{T(n-1)} + 1$$

$$= 2\cdot(2\cdot \boxed{T(n-2)} + 1) + 1$$

$$= \underbrace{2\cdot(2\cdot(2\cdot}_{2^3} T(n-\boxed{3}) \underbrace{+1) + 1) + 1}_{1*3}$$

$$= \overbrace{2\cdot(2\cdot \ldots}^{2^{n-1}} \cdot T(1)) \underbrace{+1+1+1}_{n-1}$$

$T(1) = 1$

$2^{n-1} + (n-1)$

" seconds to complete n disks

$n-1$

n