Recursive Backtracking

Hitesh Boinpally
Summer 2023

Agenda

- Review
- Gambling pt. 2
- Scrabble
- P2 Preview

Agenda

- Review 
- Gambling pt. 2
- Scrabble
- P2 Preview

Exhaustive Search

- Brute force algorithm!
  - Not **smart**
- Simply trying all possibilities, find the ones that work
- Often solved recursively

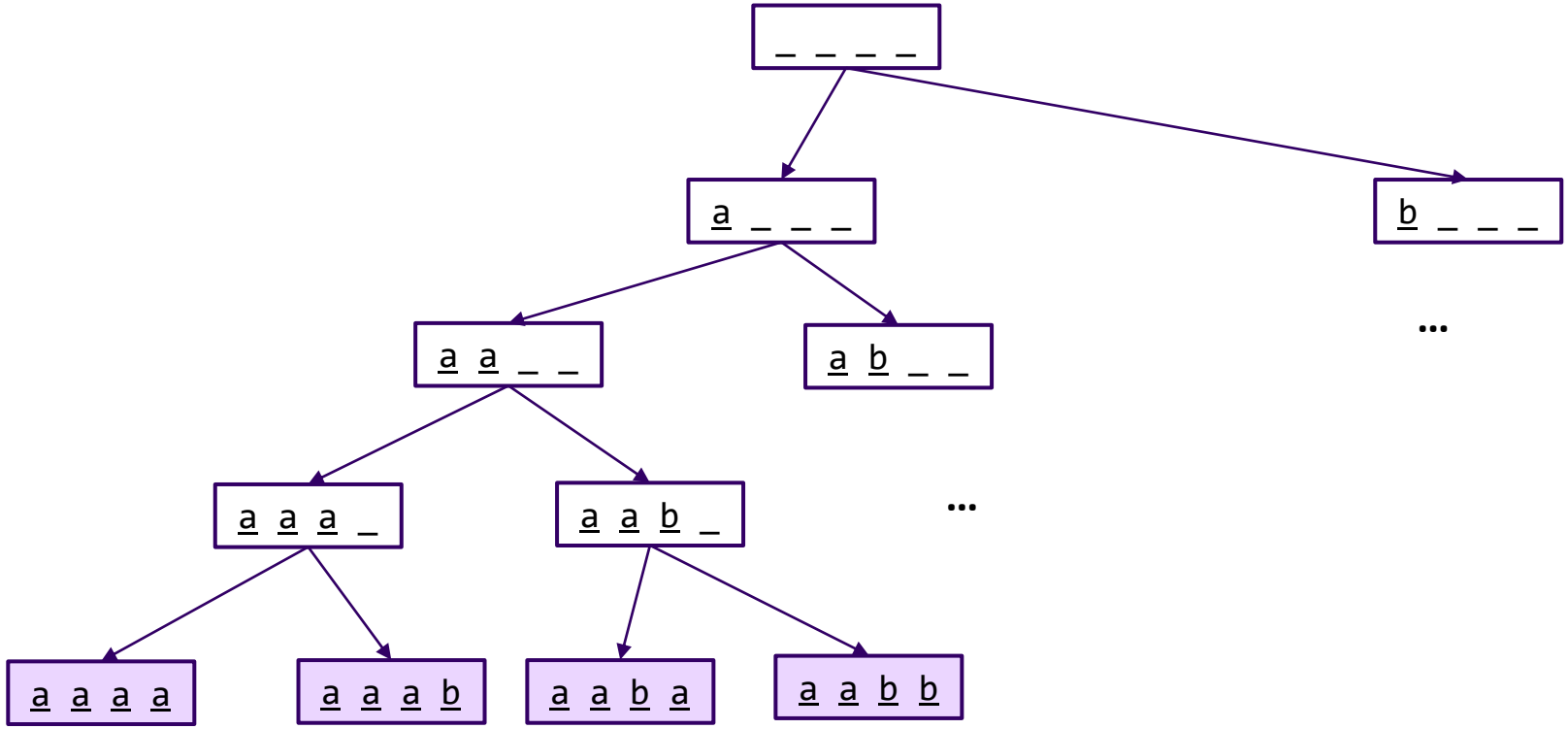
Exhaustive Search

- Brute force algorithm!
  - Not **smart**
- Simply trying all possibilities, find the ones that work
- Often solved recursively
- Two main cases
  - **Found Solution:** base case
  - **Continue Exploring**: recursive case
- Multiple recursive calls
  - One per option
- Public/Private pairs are essential

Exhaustive Search

- Brute force algorithm!
  - Not **smart**
- Simply trying all possibilities, find the ones that work
- Often solved recursively
- Two main cases
  - **Found Solution:** base case
  - **Continue Exploring**: recursive case
- Multiple recursive calls
  - One per option
- Public/Private pairs are essential

fourAB Visualized



Agenda

- Review
- Gambling pt. 2 
- Scrabble
- P2 Preview

Dice Rolls Review

- Write a method that prints out all possible outcomes when rolling some n 6-sided dice.

Example: `diceRoll(2)`

[1, 1]	[3, 1]	[5, 1]
[1, 2]	[3, 2]	[5, 2]
[1, 3]	[3, 3]	[5, 3]
[1, 4]	[3, 4]	[5, 4]
[1, 5]	[3, 5]	[5, 5]
[1, 6]	[3, 6]	[5, 6]
[2, 1]	[4, 1]	[6, 1]
[2, 2]	[4, 2]	[6, 2]
[2, 3]	[4, 3]	[6, 3]
[2, 4]	[4, 4]	[6, 4]
[2, 5]	[4, 5]	[6, 5]
[2, 6]	[4, 6]	[6, 6]

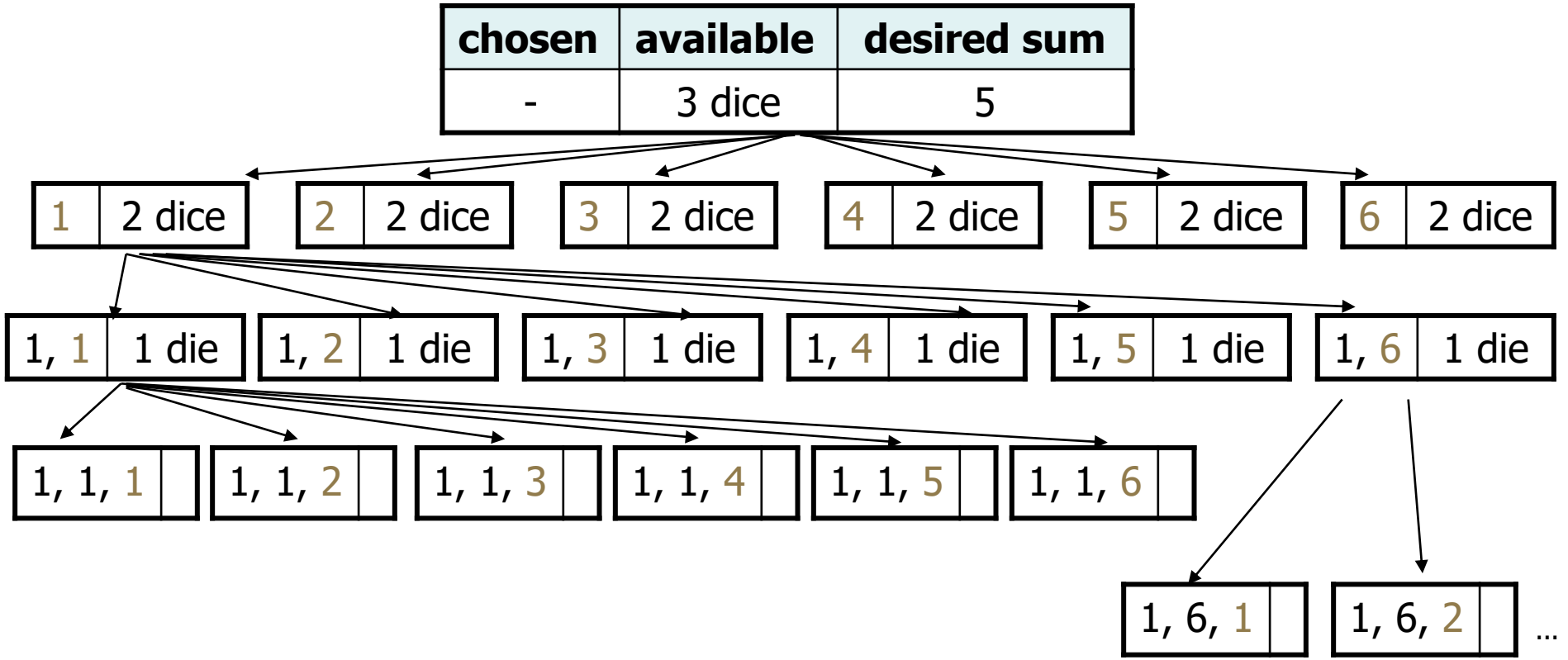
Dice Sums

- Write a method that prints out all outcomes that sum to a particular value when rolling some n 6-sided dice.

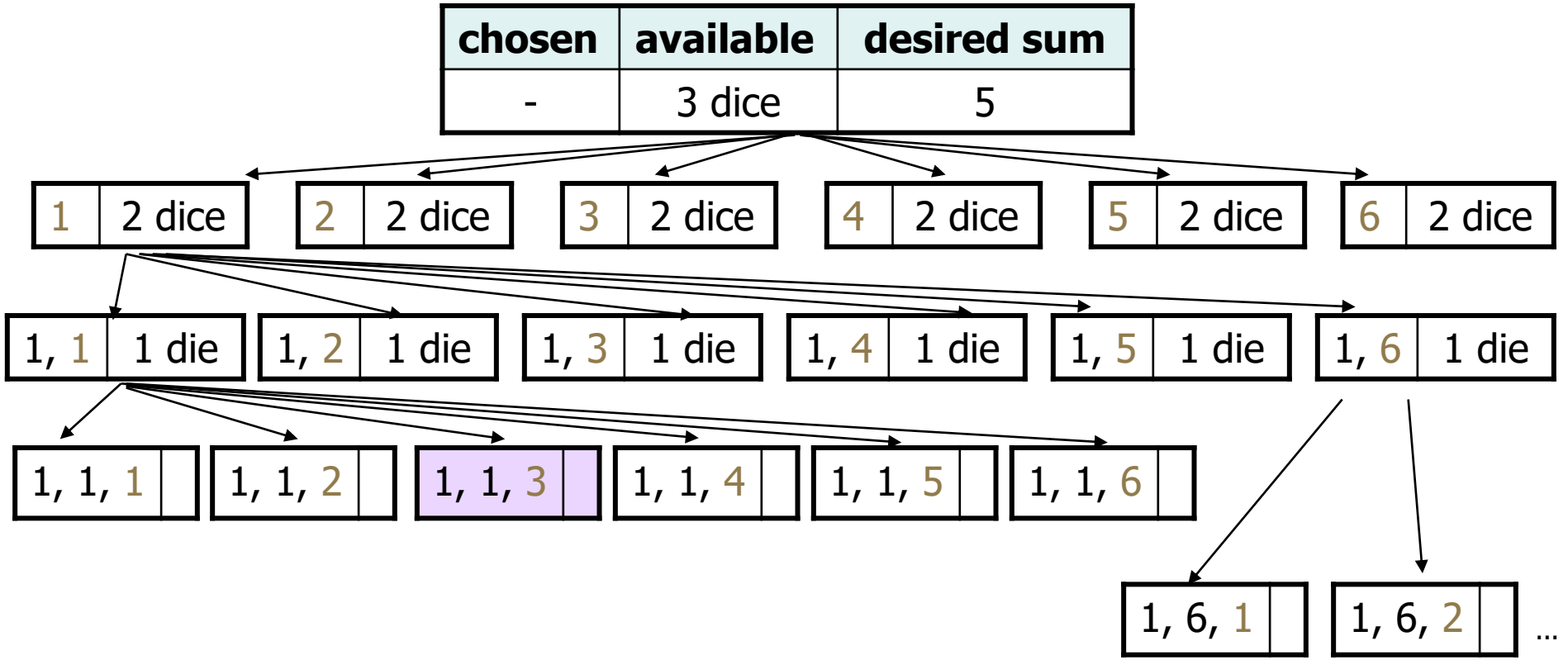
Example: `diceRoll(2, 7)`

[1, 6]
[2, 5]
[3, 4]
[4, 3]
[5, 2]
[6, 1]

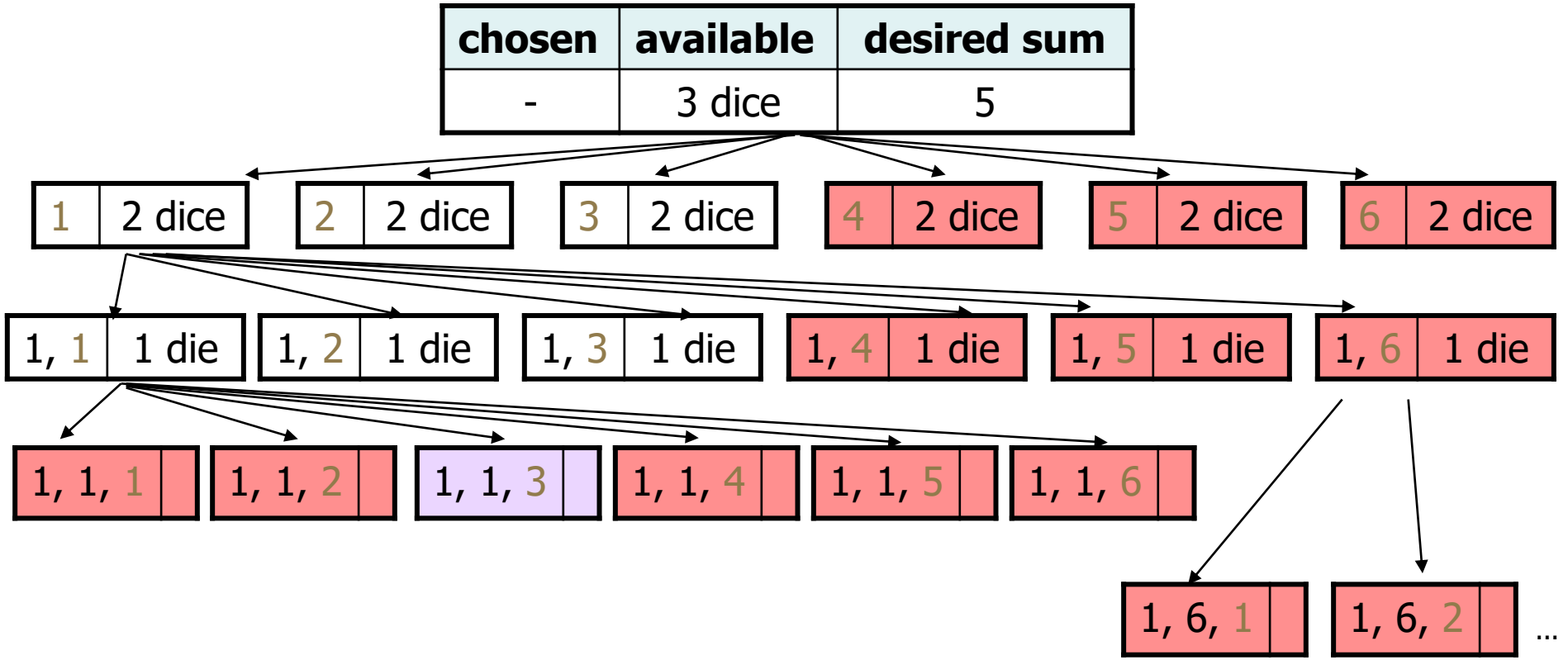
Should we consider all paths?



Should we consider all paths?

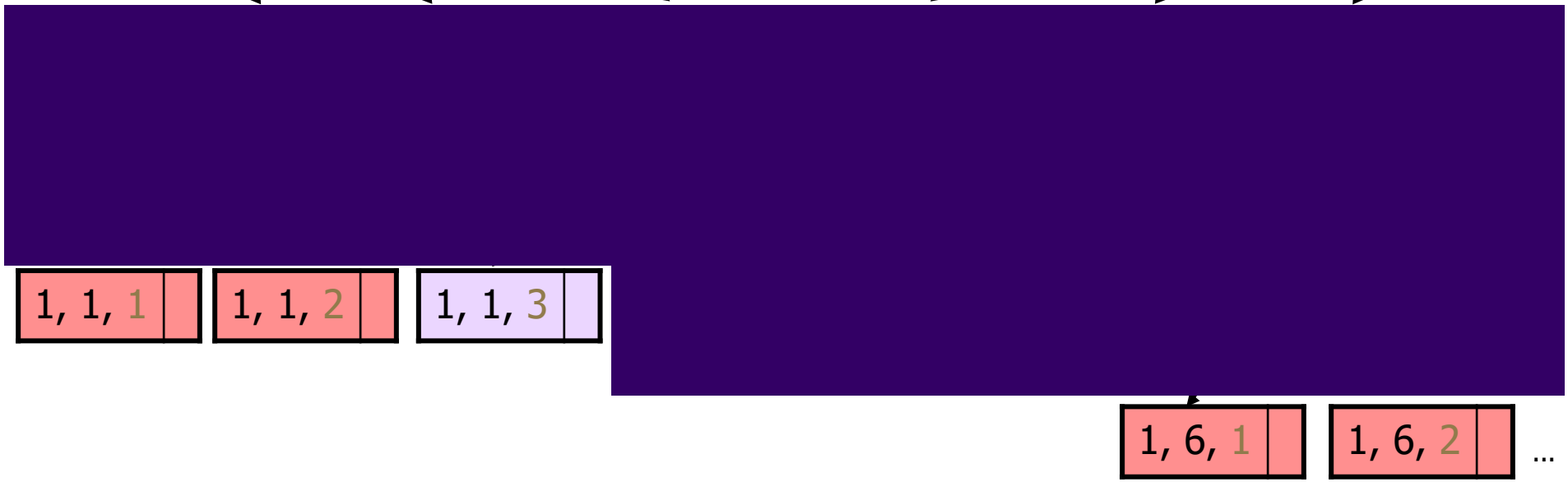


Should we consider all paths?



Should we consider all paths?

chosen	available	desired sum
-	3 dice	5



1, 1, 1

1, 1, 2

1, 1, 3

1, 6, 1

1, 6, 2

...

Exhaustive Search

- Brute force algorithm!
  - Not **smart**
- Simply trying all possibilities, find the ones that work
- Often solved recursively
- Two main cases
  - **Found Solution:** base case
  - **Continue Exploring**: recursive case
- Multiple recursive calls
  - One per option
- Public/Private pairs are essential

Recursive Backtracking

- Brute force algorithm!
  - Not **smart**
- Simply trying all possibilities, find the ones that work
- Often solved recursively
- ~~Two~~ **Three** main cases
  - **Found Solution:** base case
  - **Continue Exploring**: recursive case
  - **Dead End:** Implicit case to stop exploring
- Multiple recursive calls
  - One per option
- Public/Private pairs are essential

Agenda

- Review
- Gambling pt. 2
- Scrabble 
- P2 Preview

Agenda

- Review
- Gambling pt. 2
- Scrabble
- P2 Preview 

P2 Preview

- Utilize concepts learned today
  - Parts of `diceSum` and `Scrabble` will be useful
- Lots of helper classes
  - Make sure you understand how to use each of them!

P2 Preview

- Utilize concepts learned today
  - Parts of `diceSum` and `Scrabble` will be useful
- Lots of helper classes
  - Make sure you understand how to use each of them!
- Initial submission numbers have been low
- Remember that to be making good progress you should be submitting a (at least) **behaviorally fully functional assignment initially**
  - Then utilize the resubmission to improve your score