

Arrays as Collections

The Big Picture

Arrays are often used for holding a collection of things –

- can write loops, exploit integer indexes to avoid repeating code when performing the same task for each thing
- can accommodate the case where the number of things isn't known until runtime

Patterns of usage –

- number of things doesn't change, and is known when the array is created
- number of things can change, but the maximum number is known when the array is created
 - not all of the slots will be used all the time (*partially-full array*)
- number of things can change, but the maximum isn't known and/or the maximum is much bigger than the minimum
 - number of slots in the array may change (*dynamic array*)

2

For each of the following applications, would it be most appropriate to use a partially-full array, a dynamic array, or a regular always-full array?

the players in a game of Monopoly, where players are eliminated when they go bankrupt [Choose] partially full

the collection of cards in your hand during a game of War [Choose] dynamic

how many times each value on a 6-sided die has been rolled [Choose] regular

the tiles in the tile bag in Scrabble [Choose] partially full

the state of the board in a game of Tic-Tac-Toe [Choose] regular

the players playing at a Blackjack table at a casino, where players can join the table or leave between games [Choose] dynamic

Partially-Full Arrays

If not all of the slots are used, how do we know which ones have values and which have junk?

- keep all the used slots together (at the beginning is convenient)
- maintain an additional variable to store the number of slots in use

Distinguish *capacity* (the number of slots) from *size* (the number of slots in use at the moment).