2D Arrays & Sorting

2D Arrays

2D arrays are represented as an array in an array

In C++, they are declared as `type name [a][b]`

In Java, they can be declared as `type[][] name = new type[a][b]`

They are used to represent grids, boards, etc.

Practice

https://dmoj.ca/problem/ccc08j3

https://dmoj.ca/problem/ccc03s1

https://dmoj.ca/problem/ccc08s2

Sorting Algorithms

Insertion Sort

- Best case: O(n)
- Average case: O(n^2)
- Worst case: O(n^2)

Merge Sort

- Best case: O(nlogn)
- Average case: O(nlogn)
- Worst case: O(nlogn)

Quick Sort

- Best case: O(nlogn)
- Average case: O(nlogn), better than merge sort
- Worst case: O(n^2)

Built-in Sorting

C++ sort a vector:

```
sort(myvector.begin(), myvector.end());
```

Java sort an array:

```
Arrays.sort(myArr);
```

Both of these use some implementation of quick sort

Insertion Sort

Sort an array in the way you would compare playing cards in your hand

Iterates through an array, removes one element from the input, places it in the sorted list

Procedure

- Loop through an array
- At each position, checks the value against the largest value in the sorted list, if larger, leave it
- If smaller, go through the sorted list and insert it in (shifting over all larger ones)
- Sorts an array of size N in N – 1 iterations

See https://en.wikipedia.org/wiki/Insertion_sort for implementation

Merge Sort

Sort an array by breaking it down into smaller pieces, and merging it back together

The “merge” is where the sorting occurs

Procedure

- Repeatedly break an array in half
- Once the halves only contain single elements
- Merge them back together in a zipper-like fashion

See https://en.wikipedia.org/wiki/Merge_sort for implmenetation

Quick Sort

Recursively sorts arrays

Procedure

- Pick a pivot
- Order the array such that all elements greater than the pivot comes after, all elements less come before
- Break the array at the pivot
- Recursively sort the smaller arrays

See https://en.wikipedia.org/wiki/Quicksort for implementation

Project - Connect Four

See 3_compcont_homework3.pdf

Test

Will be held next class

90 minutes

- Conditionals
- Loops
- 1D and 2D arrays
- Sorting

You will write code on your computer to be submitted via USB stick