CSCI-UA.0380-001
Programming Challenges

Sean McIntyre
Class 02: Data Structures
Today's agenda

- Discussion and review of last class
- Lecture
- Break (~2:30-2:45pm)
- Practice problems (~2:45pm-4:15pm)
- Discussion of problems
Discussion
Rules of thumb

- $2^{10} =$
- $2^{20} =$
Rules of thumb

- Max 32-bit signed integer
- Max 64-bit signed integer
- When to use integer and long and BigInteger
Rules of thumb

- Nested for loops runtime
- Recursion runtime
Rules of thumb

• For a set of $n$ elements:
  – There are $n!$ permutations
  – There are $2^n$ subsets (combinations)
Exercise

Given a set $S$ of $N$ points randomly scattered on 2D plane, $N \leq 1000$.

Find two points $\in S$ that has the greatest Euclidean distance.
Exercise

Test if factorial of $n$, i.e. $n!$, $(1 \leq n \leq 10000)$ is divisible by an integer $m$. 
Exercise

Find all the occurrences of a substring $P$ of length $m$ in a string $T$ of length $n$, if any.

$1 \leq n, m \leq 1,000,000$
Exercise

Given a set $S$ of $N$ points randomly scattered on 2D plane, $N \leq 1,000,000$.

Find two points $\in S$ that has the greatest Euclidean distance.
Knowing your language exercise

Given a string that represents a base $X$ number, convert it to equivalent string in base $Y$, $2 \leq X, Y \leq 36$.

For example: “FF” in base $X = 16$ (Hexadecimal) is “255” in decimal ($Y = 10$) and “11111111” in binary ($Y = 2$).
Knowing your language exercise

Given a list of integers \( L \) of size up to 1M items, determine whether a value \( v \) exists in \( L \) by not using more than 20 comparisons?
Knowing your language exercise

Write the shortest possible code to read in a double and print it again, but now with minimum field width 7 and 3 digits after decimal point

e.g. ss1.473, s15.325 (‘s’ denotes a space)
Knowing your language exercise

Generate all possible permutations of \{0, 1, 2, \ldots, N-1\}, for \( N = 10 \).
Knowing your language exercise

Generate all possible subsets of \{0, 1, 2, \ldots, N-1\}, for N = 20.
Testing exercise

- You receive a WA response for a very easy problem. What should you do?
  - Abandon the problem
  - Improve the performance of your solution
  - Create tricky test cases and find the bug
How judging works

- Demo
Testing exercise

- You receive a TLE response for an your $O(N^3)$ solution. ($1 \leq N \leq 100$)
  - Abandon the problem
  - Improve the performance of your solution
  - Create tricky test cases and find the bug
Testing exercise

• You receive a TLE response for an your $O(N^3)$ solution. ($1 \leq N \leq 1,000,000$)
  – Abandon the problem
  – Improve the performance of your solution
  – Create tricky test cases and find the bug
Testing exercise

- You receive an RTE response. Your code runs OK in your machine. What should you do?
Data Structures Lecture
Linear data structures

1) Static arrays
   - int myArray[] = new int[10];

2) Resizable (array) lists
   - ArrayList myList = new ArrayList();

• Common operations on both of these:
  - Sorting and searching
Sorting lists and arrays

- Sorting algorithms:
  - $O(n^2)$ – bubble sort, insertion sort, selection sort
  - $O(n \log n)$ – merge sort, quick sort

- But all that matters is:
  - `Arrays.sort(myArray)`
  - `Collections.sort(myList)`
  - These are $O(n \log n)$
Searching lists and arrays

- O(n) linear scan if not sorted
- O(log n) binary search if sorted
- To binary search an array or list:
  - Arrays.binarySearch(myArray, x)
  - Collections.binarySearch(myList, x)
  - Returns the index, or -1 if doesn't exist
Exercise

Given a set of $A$ integers, how many integers in a second set $B$ are found within $A$?

$1 \leq |A|, |B| \leq 1,000,000$
Linear data structures

3) Bitmask
   - Set of booleans, or bitstring
   - Just an int (or a byte or a long)

• Exercise: Print all subsets of the Teenage Mutant Ninja Turtles
Linear data structures

4) LinkedList
   - Not really used

5) Stack
   - FIFO: Push, pop
   - Postfix notation, bracket matching problems

6) Queue
   - LIFO: Push, pop
   - Breadth-first search, topological sort
Exercise

Determine whether or not a string of brackets is balanced.

e.g.,

()  
(((0)(00)(0(0))))  
(([[{(0)}]]))
Non-linear data structures

1) Binary Search Tree
   - e.g., Red-Black tree, AVL tree
   - Contests: TreeMap, TreeSet

2) Hash table
   - Contests: HashMap, HashSet

3) Linked hash table
   - Contests: LinkedHashMap, LinkedHashSet
Exercise

- UVA 10226
- UVA 11340
Non-linear data structures

4) Heap

- Each parent node has a larger value than its children
- Contests: PriorityQueue
- Pop and push functions
- Useful for Prim's and Dijkstra's algorithms
Graphs

- A set of nodes connected by edges
  - Directed, undirected
  - Cyclic, acyclic
- Represented by:
  1) Adjacency Matrix
  2) Adjacency List
  3) Edge List
1) Adjacency Matrix

- 2D array of the connections between nodes
- If (matrix[i][j] == 1) then a connection of weight 1 exists between the $i$th and $j$th nodes
- $O(V)$ cost to enumerate neighbors
- Not good for large, sparse graphs
2) Adjacency List

- Each vertex has a list of vertices it is adjacent to.
- `ArrayList< ArrayList<Integer> >` for unweighted graphs.
  - The \( i \)th vertex is connected to the \( j \)th vertex if \( j \) exists in the \( i \)th list.
- `ArrayList< ArrayList<IntegerPair> >` for weighted graphs; to keep track of weights.
- First choice! Best choice!
3) Edge list

- A set of edges \( \{ (i, j) \} \)
- Useful if you want to sort all edges in the graph by weight (uncommon)
Implicit graphs

• For example:
  • 2D grid: The vertices are the cells in the 2D grid. Two neighboring cells in the grid have an edge between them.
  - A graph contains $N$ vertices numbered from $[0..N-1]$. There is an edge between two vertices $i$ and $j$ if $(i + j)$ is a prime.
Union-find disjoint sets

- A data structure for efficiently:
  - Finding which set an item belongs to
    - $O(\log n)$ for the first call, then $O(1)$ for subsequent calls
  - Union-ing two disjoint sets
    - $O(1)$, constant time
- Sometimes there are problems that rely on this
Practice
For next class

- **Readings:**
  - Programming Challenges: Chapter 2
    - We skipped Segment Trees and Fenwick Trees
  - Next class discussion of bitmasks

- **Exercises:**
  - Check website on Friday at 5pm