

COS 226 Final Exam Review Spring 2015

Ananda Gunawardena
(guna)
guna@cs.princeton.edu
guna@princeton.edu

COS 226 – Spring 2015 - Princeton University

Material covered

- The exam will *stress* material covered since the midterm, including the following components.
 - Lectures 13–23.
 - Algorithms in Java, 4th edition*, Chapters 4–6.
 - Exercises 12–22.
 - Programming assignments 6–8
 - Wordnet, seam-carving, burrows-wheeler

Logistics

- The final exam time and location
 - The final exam is from 9am to 12noon on Saturday, May 16 in **McCosh 28** or **McCosh 50**.
 - McCosh 28: Last name begins with A–F.
 - McCosh 50: Last name begins with G–Z.
 - The exam will start and end promptly, so please do arrive on time.
 - Alternate time and place
 - Monday May 18th at 1:30PM in Friend 008
- Exam Format
 - Closed book, closed note.
 - You may bring one 8.5-by-11 sheet (both sides) with notes in your own handwriting to the exam.
 - No electronic devices (e.g., calculators, laptops, and cell phones).

COS 226 – Spring 2015 - Princeton University

What to focus on

- focus on understanding basic issues, not memorizing details
- For each algorithm
 - understand how it works on typical input
 - Why do we care about this algorithm?
 - How is it different from other algorithms for the same problem?
 - When is it effective?
- For each data structure
 - invariants
 - Operations and complexity
 - applications
 - When is it effective to use a specific data structure?

Areas/Topics covered

Data Compression LZW, Huffman, run Length Encoding	String Search KMP, Boyer-Moore, <i>rabin-karp</i>
String Sorts MSD, LSD, 3-way radix quicksort	Graphs - Shortest Path BFS, Dijkstra's, Bellman-Ford, DAGs
Graphs - Traversal order BFS, DFS, Topological sort DFS - preorder, postorder	Graphs - MST Kruskals, Prim's

Areas/Topics covered

Maxflow / Mincut Augmenting paths, Ford-Fulkerson	Reductions X linear time reduces to Y
DFA / NFA Regular Expressions	Tries R-way, TST
Algorithm Analysis Big O, order of growth, Tide	Memory Analysis primitive types, objects, arrays, nested classes

Challenge Questions

- Consider each statement and state TRUE, FALSE, UNKNOWN

- An algorithm for sorting n comparable keys in linear time or less has not been invented yet *false*
- There exist an algorithm where duplicity of elements in a set can be determined in sub-linear time *false*
- The convex hull problem (i.e. finding a set of points that encloses a given set of n points) can be solved in linearithmic time *true*
- It is possible to insert n comparable keys into a BST in time proportional to n *false*

cannot sort in $< n \log n$

Algorithm Analysis

Experimental to Predictive

$$T = \frac{16}{(40000)^2} \times (200000)^2 = 400 \quad a = \frac{16}{(40000)^2}$$

Sketches that you observe the following running times for a program with an input of size N .

N	time
5,000	0.2 seconds
10,000	1.2 seconds
20,000	3.9 seconds
40,000	16.0 seconds
80,000	63.9 seconds

$$T = aN^b$$

$$a = ?, b = ?$$

Estimate the running time of the program (in seconds) on an input of size $N = 200,000$.

$$\frac{63.9}{16} = 2^b \quad b = \log_2\left(\frac{63.9}{16}\right) \quad \frac{16}{63.9} = \frac{a(40000)^b}{a(20000)^b} = \left(\frac{1}{2}\right)^b$$

Order of growth

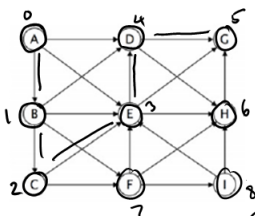
```
public static int f3(int N) {
    if (N == 0) return 1;
    int x = 0;
    for (int i = 0; i < N; i++)
        x += f3(N-1);
    return x;
}
```

```
public static int f4(int N, int R) {
    int x = 0;
    for (int i = 0; i < N; i++)
        for (int j = 1; j <= R; j += j)
            x++;
    return x;
}
```

$$f(n) = 1 + n \cdot f(n-1) \Rightarrow f(n) \sim n!$$

$$= 1 + n(1 + (n-1)f(n-2))$$

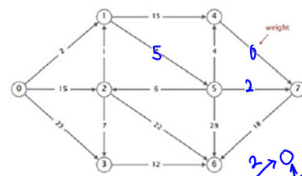
3. Graph Search



- Identify one situation where you would need to use BFS instead of DFS. *Shortest Path*
- Identify one situation where you would need to use DFS instead of BFS. *Reachability*
- Find a topological sort of the vertices (if possible)

G H D E I F C B A - Post order

8. Dijkstra's algorithm



v	distTo[]	edgeTo[]
0	0	-
1	5	0
2	15	0
3	22	0
4	10	1
5	14	4
6	36	5
7	18	5

- Give an example where Dijkstra's fail when there is a negative edge.

- What algorithm can be applied to find the shortest path when there is a negative edge?

- Is it always possible to find the shortest path when there are negative edges in the graph?

Complex
Bellman-Ford
No if there is a negative cycle

5. MST

- How does Kruskal's Differ from Prim's?
- What data structure is useful when running kruskals on a graph?
- What data structure is useful when running Prim's algorithm on a graph?
- Can minimum spanning tree algorithm be used to find the maximum spanning tree of a graph?
- How many edges does a MST contains (in terms of number of vertices)

Challenge problems

- Answer each question as possible, impossible or unknown
 - Find the strong components in a digraph in linear time *yes - Tarjan or Kosaraju*
 - Construct a binary heap in linear time *yes*
 - Find the maximum spanning tree in time proportional to $E+V$ *yes*

6. MST Algorithm Design

Suppose you know the MST of a weighted graph G . Now, a new edge $v-w$ of weight c is inserted into G to form a weighted graph G' . Design an $O(V)$ time algorithm to determine if the MST in G is also an MST in G' . You may assume all edge weights are distinct. Your answer will be graded for correctness, clarity, and conciseness.

- State the algorithm
- Explain why your algorithm takes $O(V)$ time

Find open from $v \rightarrow w$ if any edge has weight $> c$ swap with new edge

Since MST is a tree it has $V-1$ edges

13. KMP Table

X reduces to Y

	0	1	2	3	4	5	6	7
a	0	1	1	4	1	6	-	-
b	0	0	0	0	5	-	-	-
c	0	0	3	0	0	7	-	-

7. Match Algorithms

--- T9 texting in a cell phone	A. Trie
--- 1D range search	B. Hashing
--- 2D range search	C. 3-way radix quicksort
--- Document similarity	D. Binary search tree
--- Traveling salesperson problem	E. Kd tree
--- Web crawler	F. Depth-first search
--- Google maps	G. Breadth-first search
--- PERT/CPM (Program Evaluation and Review Technique / Critical Path Method)	H. Dijkstra's algorithm
	I. Topological sort
	J. Bellman-Ford
	K. Enumerate permutations

$a_0 + a_1b + a_2b^2 + \dots + a_nb^n$
 $a_0 + b(a_1 + a_2b + \dots + a_nb^{n-1})$

14. LZW compression

1. Compressing

ABABABABAB (A=41, B=42, next code= 81)

Handwritten: 41 42 81 41 83 84 81 42

2. Expanding

What is the result of expanding the following LZW-encoded sequence of 11 hexadecimal integers?

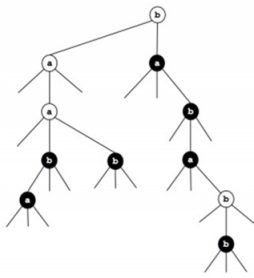
Handwritten: 43 41 42 42 82 43 81 41 87 82 80

Assume the original encoding table consists of all 7-bit ASCII characters and uses 8-bit codewords. Recall that codeword 80 is reserved to signify end of file.

C	A	B	B							
C	A	B	B							

Handwritten calculations: CA=81, CB=82, CBA=83, CBA=84, CBA=85, CBA=86, CBA=87, CBA=88, CBA=89, CBA=90, CBA=91, CBA=92, CBA=93, CBA=94, CBA=95, CBA=96, CBA=97, CBA=98, CBA=99, CBA=100, CBA=101, CBA=102, CBA=103, CBA=104, CBA=105, CBA=106, CBA=107, CBA=108, CBA=109, CBA=110, CBA=111, CBA=112, CBA=113, CBA=114, CBA=115, CBA=116, CBA=117, CBA=118, CBA=119, CBA=120, CBA=121, CBA=122, CBA=123, CBA=124, CBA=125, CBA=126, CBA=127, CBA=128, CBA=129, CBA=130, CBA=131, CBA=132, CBA=133, CBA=134, CBA=135, CBA=136, CBA=137, CBA=138, CBA=139, CBA=140, CBA=141, CBA=142, CBA=143, CBA=144, CBA=145, CBA=146, CBA=147, CBA=148, CBA=149, CBA=150, CBA=151, CBA=152, CBA=153, CBA=154, CBA=155, CBA=156, CBA=157, CBA=158, CBA=159, CBA=160, CBA=161, CBA=162, CBA=163, CBA=164, CBA=165, CBA=166, CBA=167, CBA=168, CBA=169, CBA=170, CBA=171, CBA=172, CBA=173, CBA=174, CBA=175, CBA=176, CBA=177, CBA=178, CBA=179, CBA=180, CBA=181, CBA=182, CBA=183, CBA=184, CBA=185, CBA=186, CBA=187, CBA=188, CBA=189, CBA=190, CBA=191, CBA=192, CBA=193, CBA=194, CBA=195, CBA=196, CBA=197, CBA=198, CBA=199, CBA=200, CBA=201, CBA=202, CBA=203, CBA=204, CBA=205, CBA=206, CBA=207, CBA=208, CBA=209, CBA=210, CBA=211, CBA=212, CBA=213, CBA=214, CBA=215, CBA=216, CBA=217, CBA=218, CBA=219, CBA=220, CBA=221, CBA=222, CBA=223, CBA=224, CBA=225, CBA=226, CBA=227, CBA=228, CBA=229, CBA=230, CBA=231, CBA=232, CBA=233, CBA=234, CBA=235, CBA=236, CBA=237, CBA=238, CBA=239, CBA=240, CBA=241, CBA=242, CBA=243, CBA=244, CBA=245, CBA=246, CBA=247, CBA=248, CBA=249, CBA=250, CBA=251, CBA=252, CBA=253, CBA=254, CBA=255, CBA=256, CBA=257, CBA=258, CBA=259, CBA=260, CBA=261, CBA=262, CBA=263, CBA=264, CBA=265, CBA=266, CBA=267, CBA=268, CBA=269, CBA=270, CBA=271, CBA=272, CBA=273, CBA=274, CBA=275, CBA=276, CBA=277, CBA=278, CBA=279, CBA=280, CBA=281, CBA=282, CBA=283, CBA=284, CBA=285, CBA=286, CBA=287, CBA=288, CBA=289, CBA=290, CBA=291, CBA=292, CBA=293, CBA=294, CBA=295, CBA=296, CBA=297, CBA=298, CBA=299, CBA=300, CBA=301, CBA=302, CBA=303, CBA=304, CBA=305, CBA=306, CBA=307, CBA=308, CBA=309, CBA=310, CBA=311, CBA=312, CBA=313, CBA=314, CBA=315, CBA=316, CBA=317, CBA=318, CBA=319, CBA=320, CBA=321, CBA=322, CBA=323, CBA=324, CBA=325, CBA=326, CBA=327, CBA=328, CBA=329, CBA=330, CBA=331, CBA=332, CBA=333, CBA=334, CBA=335, CBA=336, CBA=337, CBA=338, CBA=339, CBA=340, CBA=341, CBA=342, CBA=343, CBA=344, CBA=345, CBA=346, CBA=347, CBA=348, CBA=349, CBA=350, CBA=351, CBA=352, CBA=353, CBA=354, CBA=355, CBA=356, CBA=357, CBA=358, CBA=359, CBA=360, CBA=361, CBA=362, CBA=363, CBA=364, CBA=365, CBA=366, CBA=367, CBA=368, CBA=369, CBA=370, CBA=371, CBA=372, CBA=373, CBA=374, CBA=375, CBA=376, CBA=377, CBA=378, CBA=379, CBA=380, CBA=381, CBA=382, CBA=383, CBA=384, CBA=385, CBA=386, CBA=387, CBA=388, CBA=389, CBA=390, CBA=391, CBA=392, CBA=393, CBA=394, CBA=395, CBA=396, CBA=397, CBA=398, CBA=399, CBA=400, CBA=401, CBA=402, CBA=403, CBA=404, CBA=405, CBA=406, CBA=407, CBA=408, CBA=409, CBA=410, CBA=411, CBA=412, CBA=413, CBA=414, CBA=415, CBA=416, CBA=417, CBA=418, CBA=419, CBA=420, CBA=421, CBA=422, CBA=423, CBA=424, CBA=425, CBA=426, CBA=427, CBA=428, CBA=429, CBA=430, CBA=431, CBA=432, CBA=433, CBA=434, CBA=435, CBA=436, CBA=437, CBA=438, CBA=439, CBA=440, CBA=441, CBA=442, CBA=443, CBA=444, CBA=445, CBA=446, CBA=447, CBA=448, CBA=449, CBA=450, CBA=451, CBA=452, CBA=453, CBA=454, CBA=455, CBA=456, CBA=457, CBA=458, CBA=459, CBA=460, CBA=461, CBA=462, CBA=463, CBA=464, CBA=465, CBA=466, CBA=467, CBA=468, CBA=469, CBA=470, CBA=471, CBA=472, CBA=473, CBA=474, CBA=475, CBA=476, CBA=477, CBA=478, CBA=479, CBA=480, CBA=481, CBA=482, CBA=483, CBA=484, CBA=485, CBA=486, CBA=487, CBA=488, CBA=489, CBA=490, CBA=491, CBA=492, CBA=493, CBA=494, CBA=495, CBA=496, CBA=497, CBA=498, CBA=499, CBA=500, CBA=501, CBA=502, CBA=503, CBA=504, CBA=505, CBA=506, CBA=507, CBA=508, CBA=509, CBA=510, CBA=511, CBA=512, CBA=513, CBA=514, CBA=515, CBA=516, CBA=517, CBA=518, CBA=519, CBA=520, CBA=521, CBA=522, CBA=523, CBA=524, CBA=525, CBA=526, CBA=527, CBA=528, CBA=529, CBA=530, CBA=531, CBA=532, CBA=533, CBA=534, CBA=535, CBA=536, CBA=537, CBA=538, CBA=539, CBA=540, CBA=541, CBA=542, CBA=543, CBA=544, CBA=545, CBA=546, CBA=547, CBA=548, CBA=549, CBA=550, CBA=551, CBA=552, CBA=553, CBA=554, CBA=555, CBA=556, CBA=557, CBA=558, CBA=559, CBA=560, CBA=561, CBA=562, CBA=563, CBA=564, CBA=565, CBA=566, CBA=567, CBA=568, CBA=569, CBA=570, CBA=571, CBA=572, CBA=573, CBA=574, CBA=575, CBA=576, CBA=577, CBA=578, CBA=579, CBA=580, CBA=581, CBA=582, CBA=583, CBA=584, CBA=585, CBA=586, CBA=587, CBA=588, CBA=589, CBA=590, CBA=591, CBA=592, CBA=593, CBA=594, CBA=595, CBA=596, CBA=597, CBA=598, CBA=599, CBA=600, CBA=601, CBA=602, CBA=603, CBA=604, CBA=605, CBA=606, CBA=607, CBA=608, CBA=609, CBA=610, CBA=611, CBA=612, CBA=613, CBA=614, CBA=615, CBA=616, CBA=617, CBA=618, CBA=619, CBA=620, CBA=621, CBA=622, CBA=623, CBA=624, CBA=625, CBA=626, CBA=627, CBA=628, CBA=629, CBA=630, CBA=631, CBA=632, CBA=633, CBA=634, CBA=635, CBA=636, CBA=637, CBA=638, CBA=639, CBA=640, CBA=641, CBA=642, CBA=643, CBA=644, CBA=645, CBA=646, CBA=647, CBA=648, CBA=649, CBA=650, CBA=651, CBA=652, CBA=653, CBA=654, CBA=655, CBA=656, CBA=657, CBA=658, CBA=659, CBA=660, CBA=661, CBA=662, CBA=663, CBA=664, CBA=665, CBA=666, CBA=667, CBA=668, CBA=669, CBA=670, CBA=671, CBA=672, CBA=673, CBA=674, CBA=675, CBA=676, CBA=677, CBA=678, CBA=679, CBA=680, CBA=681, CBA=682, CBA=683, CBA=684, CBA=685, CBA=686, CBA=687, CBA=688, CBA=689, CBA=690, CBA=691, CBA=692, CBA=693, CBA=694, CBA=695, CBA=696, CBA=697, CBA=698, CBA=699, CBA=700, CBA=701, CBA=702, CBA=703, CBA=704, CBA=705, CBA=706, CBA=707, CBA=708, CBA=709, CBA=710, CBA=711, CBA=712, CBA=713, CBA=714, CBA=715, CBA=716, CBA=717, CBA=718, CBA=719, CBA=720, CBA=721, CBA=722, CBA=723, CBA=724, CBA=725, CBA=726, CBA=727, CBA=728, CBA=729, CBA=730, CBA=731, CBA=732, CBA=733, CBA=734, CBA=735, CBA=736, CBA=737, CBA=738, CBA=739, CBA=740, CBA=741, CBA=742, CBA=743, CBA=744, CBA=745, CBA=746, CBA=747, CBA=748, CBA=749, CBA=750, CBA=751, CBA=752, CBA=753, CBA=754, CBA=755, CBA=756, CBA=757, CBA=758, CBA=759, CBA=760, CBA=761, CBA=762, CBA=763, CBA=764, CBA=765, CBA=766, CBA=767, CBA=768, CBA=769, CBA=770, CBA=771, CBA=772, CBA=773, CBA=774, CBA=775, CBA=776, CBA=777, CBA=778, CBA=779, CBA=780, CBA=781, CBA=782, CBA=783, CBA=784, CBA=785, CBA=786, CBA=787, CBA=788, CBA=789, CBA=790, CBA=791, CBA=792, CBA=793, CBA=794, CBA=795, CBA=796, CBA=797, CBA=798, CBA=799, CBA=800, CBA=801, CBA=802, CBA=803, CBA=804, CBA=805, CBA=806, CBA=807, CBA=808, CBA=809, CBA=810, CBA=811, CBA=812, CBA=813, CBA=814, CBA=815, CBA=816, CBA=817, CBA=818, CBA=819, CBA=820, CBA=821, CBA=822, CBA=823, CBA=824, CBA=825, CBA=826, CBA=827, CBA=828, CBA=829, CBA=830, CBA=831, CBA=832, CBA=833, CBA=834, CBA=835, CBA=836, CBA=837, CBA=838, CBA=839, CBA=840, CBA=841, CBA=842, CBA=843, CBA=844, CBA=845, CBA=846, CBA=847, CBA=848, CBA=849, CBA=850, CBA=851, CBA=852, CBA=853, CBA=854, CBA=855, CBA=856, CBA=857, CBA=858, CBA=859, CBA=860, CBA=861, CBA=862, CBA=863, CBA=864, CBA=865, CBA=866, CBA=867, CBA=868, CBA=869, CBA=870, CBA=871, CBA=872, CBA=873, CBA=874, CBA=875, CBA=876, CBA=877, CBA=878, CBA=879, CBA=880, CBA=881, CBA=882, CBA=883, CBA=884, CBA=885, CBA=886, CBA=887, CBA=888, CBA=889, CBA=890, CBA=891, CBA=892, CBA=893, CBA=894, CBA=895, CBA=896, CBA=897, CBA=898, CBA=899, CBA=900, CBA=901, CBA=902, CBA=903, CBA=904, CBA=905, CBA=906, CBA=907, CBA=908, CBA=909, CBA=910, CBA=911, CBA=912, CBA=913, CBA=914, CBA=915, CBA=916, CBA=917, CBA=918, CBA=919, CBA=920, CBA=921, CBA=922, CBA=923, CBA=924, CBA=925, CBA=926, CBA=927, CBA=928, CBA=929, CBA=930, CBA=931, CBA=932, CBA=933, CBA=934, CBA=935, CBA=936, CBA=937, CBA=938, CBA=939, CBA=940, CBA=941, CBA=942, CBA=943, CBA=944, CBA=945, CBA=946, CBA=947, CBA=948, CBA=949, CBA=950, CBA=951, CBA=952, CBA=953, CBA=954, CBA=955, CBA=956, CBA=957, CBA=958, CBA=959, CBA=960, CBA=961, CBA=962, CBA=963, CBA=964, CBA=965, CBA=966, CBA=967, CBA=968, CBA=969, CBA=970, CBA=971, CBA=972, CBA=973, CBA=974, CBA=975, CBA=976, CBA=977, CBA=978, CBA=979, CBA=980, CBA=981, CBA=982, CBA=983, CBA=984, CBA=985, CBA=986, CBA=987, CBA=988, CBA=989, CBA=990, CBA=991, CBA=992, CBA=993, CBA=994, CBA=995, CBA=996, CBA=997, CBA=998, CBA=999, CBA=1000, CBA=1001, CBA=1002, CBA=1003, CBA=1004, CBA=1005, CBA=1006, CBA=1007, CBA=1008, CBA=1009, CBA=1010, CBA=1011, CBA=1012, CBA=1013, CBA=1014, CBA=1015, CBA=1016, CBA=1017, CBA=1018, CBA=1019, CBA=1020, CBA=1021, CBA=1022, CBA=1023, CBA=1024, CBA=1025, CBA=1026, CBA=1027, CBA=1028, CBA=1029, CBA=1030, CBA=1031, CBA=1032, CBA=1033, CBA=1034, CBA=1035, CBA=1036, CBA=1037, CBA=1038, CBA=1039, CBA=1040, CBA=1041, CBA=1042, CBA=1043, CBA=1044, CBA=1045, CBA=1046, CBA=1047, CBA=1048, CBA=1049, CBA=1050, CBA=1051, CBA=1052, CBA=1053, CBA=1054, CBA=1055, CBA=1056, CBA=1057, CBA=1058, CBA=1059, CBA=1060, CBA=1061, CBA=1062, CBA=1063, CBA=1064, CBA=1065, CBA=1066, CBA=1067, CBA=1068, CBA=1069, CBA=1070, CBA=1071, CBA=1072, CBA=1073, CBA=1074, CBA=1075, CBA=1076, CBA=1077, CBA=1078, CBA=1079, CBA=1080, CBA=1081, CBA=1082, CBA=1083, CBA=1084, CBA=1085, CBA=1086, CBA=1087, CBA=1088, CBA=1089, CBA=1090, CBA=1091, CBA=1092, CBA=1093, CBA=1094, CBA=1095, CBA=1096, CBA=1097, CBA=1098, CBA=1099, CBA=1100, CBA=1101, CBA=1102, CBA=1103, CBA=1104, CBA=1105, CBA=1106, CBA=1107, CBA=1108, CBA=1109, CBA=1110, CBA=1111, CBA=1112, CBA=1113, CBA=1114, CBA=1115, CBA=1116, CBA=1117, CBA=1118, CBA=1119, CBA=1120, CBA=1121, CBA=1122, CBA=1123, CBA=1124, CBA=1125, CBA=1126, CBA=1127, CBA=1128, CBA=1129, CBA=1130, CBA=1131, CBA=1132, CBA=1133, CBA=1134, CBA=1135, CBA=1136, CBA=1137, CBA=1138, CBA=1139, CBA=1140, CBA=1141, CBA=1142, CBA=1143, CBA=1144, CBA=1145, CBA=1146, CBA=1147, CBA=1148, CBA=1149, CBA=1150, CBA=1151, CBA=1152, CBA=1153, CBA=1154, CBA=1155, CBA=1156, CBA=1157, CBA=1158, CBA=1159, CBA=1160, CBA=1161, CBA=1162, CBA=1163, CBA=1164, CBA=1165, CBA=1166, CBA=1167, CBA=1168, CBA=1169, CBA=1170, CBA=1171, CBA=1172, CBA=1173, CBA=1174, CBA=1175, CBA=1176, CBA=1177, CBA=1178, CBA=1179, CBA=1180, CBA=1181, CBA=1182, CBA=1183, CBA=1184, CBA=1185, CBA=1186, CBA=1187, CBA=1188, CBA=1189, CBA=1190, CBA=1191, CBA=1192, CBA=1193, CBA=1194, CBA=1195, CBA=1196, CBA=1197, CBA=1198, CBA=1199, CBA=1200, CBA=1201, CBA=1202, CBA=1203, CBA=1204, CBA=1205, CBA=1206, CBA=1207, CBA=1208, CBA=1209, CBA=1210, CBA=1211, CBA=1212, CBA=1213, CBA=1214, CBA=1215, CBA=1216, CBA=1217, CBA=1218, CBA=1219, CBA=1220, CBA=1221, CBA=1222, CBA=1223, CBA=1224, CBA=1225, CBA=1226, CBA=1227, CBA=1228, CBA=1229, CBA=1230, CBA=1231, CBA=1232, CBA=1233, CBA=1234, CBA=1235, CBA=1236, CBA=1237, CBA=1238, CBA=1239, CBA=1240, CBA=1241, CBA=1242, CBA=1243, CBA=1244, CBA=1245, CBA=1246, CBA=1247, CBA=1248, CBA=1249, CBA=1250, CBA=1251, CBA=1252, CBA=1253, CBA=1254, CBA=1255, CBA=1256, CBA=1257, CBA=1258, CBA=1259, CBA=1260, CBA=1261, CBA=1262, CBA=1263, CBA=1264, CBA=1265, CBA=1266, CBA=1267, CBA=1268, CBA=1269, CBA=1270, CBA=1271, CBA=1272, CBA=1273, CBA=1274, CBA=1275, CBA=1276, CBA=1277, CBA=1278, CBA=1279, CBA=1280, CBA=1281, CBA=1282, CBA=1283, CBA=1284, CBA=1285, CBA=1286, CBA=1287, CBA=1288, CBA=1289, CBA=1290, CBA=1291, CBA=1292, CBA=1293, CBA=1294, CBA=1295, CBA=1296, CBA=1297, CBA=1298, CBA=1299, CBA=1300, CBA=1301, CBA=1302, CBA=1303, CBA=1304, CBA=1305, CBA=1306, CBA=1307, CBA=1308, CBA=1309, CBA=1310, CBA=1311, CBA=1312, CBA=1313, CBA=1314, CBA=1315, CBA=1316, CBA=1317, CBA=1318, CBA=1319, CBA=1320, CBA=1321, CBA=1322, CBA=1323, CBA=1324, CBA=1325, CBA=1326, CBA=1327, CBA=1328, CBA=1329, CBA=1330, CBA=1331, CBA=1332, CBA=1333, CBA=1334, CBA=1335, CBA=1336, CBA=1337, CBA=1338, CBA=1339, CBA=1340, CBA=1341, CBA=1342, CBA=1343, CBA=1344, CBA=1345, CBA=1346, CBA=1347, CBA=1348, CBA=1349, CBA=1350, CBA=1351, CBA=1352, CBA=1353, CBA=1354, CBA=1355, CBA=1356, CBA=1357, CBA=1358, CBA=1359, CBA=1360, CBA=1361, CBA=1362, CBA=1363, CBA=1364, CBA=1365, CBA=1366, CBA=1367, CBA=1368, CBA=1369, CBA=1370, CBA=1371, CBA=1372, CBA=1373, CBA=1374, CBA=1375, CBA=1376, CBA=1377, CBA=1378, CBA=1379, CBA=1380, CBA=1381, CBA=1382, CBA=1383, CBA=1384, CBA=1385, CBA=1386, CBA=1387, CBA=1388, CBA=1389, CBA=1390, CBA=1391, CBA=1392, CBA=1393, CBA=1394, CBA=1395, CBA=1396, CBA=1397, CBA=1398, CBA=1399, CBA=1400, CBA=1401, CBA=1402, CBA=1403, CBA=1404, CBA=1405, CBA=1406, CBA=1407, CBA=1408, CBA=1409, CBA=1410, CBA=1411, CBA=1412, CBA=1413, CBA=1414, CBA=1415, CBA=1416, CBA=1417, CBA=1418, CBA=1419, CBA=1420, CBA=1421, CBA=1422, CBA=1423, CBA=1424, CBA=1425, CBA=1426, CBA=1427, CBA=1428, CBA=1429, CBA=1430, CBA=1431, CBA=1432, CBA=1433, CBA=1434, CBA=1435, CBA=1436, CBA=1437, CBA=1438, CBA=1439, CBA=1440, CBA=1441, CBA=1442, CBA=1443, CBA=1444, CBA=1445, CBA=1446, CBA=1447, CBA=1448, CBA=1449, CBA=1450, CBA=1451, CBA=1452, CBA=1453, CBA=1454, CBA=1455, CBA=1456, CBA=1457, CBA=1458, CBA=1459, CBA=1460, CBA=1461, CBA=1462, CBA=1463, CBA=1464, CBA=1465, CBA=1466, CBA=1467, CBA=1468, CBA=1469, CBA=1470, CBA=1471, CBA=1472, CBA=1473, CBA=1474, CBA=1475, CBA=1476, CBA=1477, CBA=1478, CBA=1479, CBA=148

9. TST



1. List the words in alphabetical order (black nodes denote the end of a word)
2. Insert aaca to TST
3. Why and when would you use a TST instead of a R-way trie?

21. counting memory

- standard data types
- object overhead – 16 bytes
- array overhead – 24 bytes
- references – 8 bytes
- Inner class reference – 8 bytes

```
public class TwoThreeTreeKey extends Comparable<Key>, Value> {
    private Node root;

    private class Node {
        private int count; // subtree count
        private Key key1, key2; // the one or two keys
        private Value value1, value2; // the one or two values
        private Node left, middle, right; // the two or three subtrees
    }
    ...
}
```

- How much memory is needed for a 2-3 tree that holds N nodes?

10. String Sorting

Put an X in each box if the string sorting algorithm (the standard version considered in class) has the corresponding property.

	mergesort	LSD radix sort	MSD radix sort	3-way radix quicksort
stable	✓	✓	✓	✗
in-place	✗	✗	✗	✓
sublinear time (in best case)	✗	✗	✓	✗
fixed-length strings only		✓		

22. String Sorting

List key invariants for each algorithm

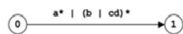
1. MSD
2. LSD
3. 3-way radix quicksort

Handwritten notes and diagrams:

- Red circles around 'LSD' and 'MSD' in the list.
- Handwritten 'MSD' and 'LSD' with arrows pointing to the list items.
- Handwritten '3 4 4 2 3 2 2 1' with arrows pointing to the list items.
- Handwritten '1 2 3' and '1 2 3' with arrows pointing to the list items.
- Handwritten '1 2 3' and '1 2 3' with arrows pointing to the list items.

12. Regular Expression to NFA

Convert the RE $a^* | (b | cd)^*$ into an equivalent NFA using the algorithm described in lecture, showing the result after applying each transformation.



3

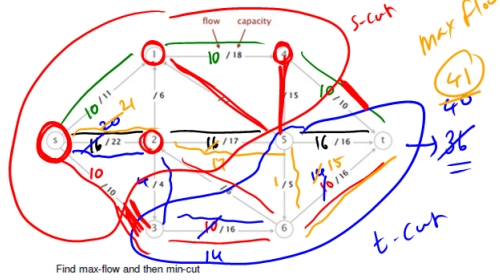
KMP Table

Identify the string using the partially completed DFA

	0	1	2	3	4	5	6	7	8	9	10
A	0	0	3	0			7			10	11
B	1	2	2	4	5		2				1
S	B	B	A	B	B	A	B	B	A	A	A

Complete the DFA

15. MaxFlow-MinCut



17. Algorithm Design

In data compression, a set of binary code words is *prefix-free* if no code word is a prefix of another. For example, $\{01, 10, 0010, 1111\}$ is prefix free, but $\{01, 10, 0010, 10100\}$ is not because 10 is a prefix of 10100.

- Design an efficient algorithm to determine if a set of binary code words is prefix-free

Insert into a tree

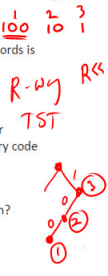
- What is the order of growth of the worst-case running time of your algorithm as a function of N and W , where N is the number of binary code words and W is the total number of bits in the input?

W

- What is the order of growth of the memory usage of your algorithm?

WR

$R=2$



17. Algorithm Design

In data compression, a set of binary code words is *prefix-free* if no code word is a prefix of another. For example, $\{01, 10, 0010, 1111\}$ is prefix free, but $\{01, 10, 0010, 10100\}$ is not because 10 is a prefix of 10100.

- Design an efficient algorithm to determine if a set of binary code words is prefix-free
- What is the order of growth of the worst-case running time of your algorithm as a function of N and W , where N is the number of binary code words and W is the total number of bits in the input?
- What is the order of growth of the memory usage of your algorithm?

19. Burrows-Wheeler

What is the Burrows-Wheeler transform of

b a b a a b a c

What is the Burrows-Wheeler inverse transform of

7
b b b b a a a a a

23. Reductions

Consider the following two problems:

- 3SUM. Given N integers x_1, x_2, \dots, x_N , are there three distinct indices i, j , and k such that $x_i + x_j + x_k = 0$?
- 3SUMPLUS. Given N integers x_1, x_2, \dots, x_N and an integer b , are there three distinct indices i, j , and k such that $x_i + x_j + x_k = b$?

- Show that 3SUM linear-time reduces to 3SUMPLUS. To demonstrate your reduction, give the 3SUMPLUS instance that you would construct to solve the following 3SUM instance: x_1, x_2, \dots, x_N .

- Show that 3SUMPLUS linear-time reduces to 3SUM. To demonstrate your reduction, give the 3SUM instance that you would construct to solve the following 3SUMPLUS instance: b, x_1, x_2, \dots, x_N .

3SUM Reduces to 3SUMPLUS $\rightarrow b=0$
3SUMPLUS Reduces to 3SUM \rightarrow

$$x_i + x_j + x_k = 0$$

$$x_i + x_j + x_k = b$$

$$(3x_i - b) + (3x_j - b) + (3x_k - b) = 0$$

$$\Rightarrow x_i + x_j + x_k = b$$