CS302 Final Exam, December 11, 2018 - James S. Plank

There are five questions to this exam. Please answer all five questions.
Hand in your work on the answer sheets provided.

My advocated timings on these questions, and their anticipated point values, are below.

- Question 1: 15 minutes, 15 points.
- Question 2: 15 minutes, 12 points.
- Question 3: 20 minutes. 20 points.
- Question 4: 40 minutes. 33 points.
- Question 5: 30 minutes. 20 points.

Question 1

You are implementing `quick2_sort.cpp` as you did in lab, using the following prototype:

```
void recursive_sort(vector <double> &v, int start, int size, int print)
```

Suppose you call **recursive_sort(v, 0, 13, 0)**, and **v** equals the following:

```
0.12  6.25  5.56  3.76  2.38  0.45  2.88  0.20  0.86  7.98  6.12  8.32  8.96
```

Part A: What wil be the value of your pivot?

Now, suppose you call **recursive_sort(v, 0, 13, 0)**, and **v** equals the following:

```
6.67  0.66  9.91  0.95  8.26  5.15  9.81  2.72  5.69  8.17  0.88  0.46  6.05
```

Part B: Assuming the value of the pivot is 6.67, you will make two recursive calls. Please tell me what `start` and `size` will be for each of these recursive calls.

Now, suppose you call **recursive_sort(v, 0, 13, 0)**, and **v** equals the following:

```
2.10  3.39  1.19  1.19  2.10  3.39  3.39  3.39  1.19  2.10  2.10  2.10  2.10
```

Part C: Assuming the value of the pivot is 2.10, you will make two recursive calls. Please tell me what `start` and `size` will be for each of these recursive calls.

CS302 Final Exam, December 11, 2018 - James S. Plank

Question 1, Continued

You are implementing `merge1_sort.cpp` as you did in lab, with the following prototype:

```
void recursive_sort(vector <double> &v, vector <double> &temp, int start, int size, int print);
```

Suppose you call **recursive_sort(v, temp, 6, 6, 0)**, and **v** equals the following:

```
3.71  6.35  6.51  5.44  1.43  7.42  2.97  5.28  0.47  7.03  0.42  4.42
```

Part D: You will make two recursive calls. Please tell me what `start` and `size` will be for each of these calls.

Part E: From the answers below, choose the one that represents what **v** is immediately after you have made the recursive calls from **Part D**.

Part F: From the answers below, choose the one that represents what **v** is when you return from **recursive_sort()**.

Answers from which to choose. Note that these are in lexicographic order, so that you should be able to find your answer quickly.

a.	0.42	0.47	1.43	2.97	3.71	4.42	5.28	5.44	6.35	6.51	7.03	7.42
b.	0.42	0.47	1.43	2.97	3.71	4.42	5.44	6.35	5.28	7.42	7.03	6.51
c.	0.42	0.47	1.43	2.97	3.71	4.42	6.35	5.28	7.42	7.03	6.51	5.44
d.	0.42	0.47	1.43	2.97	3.71	4.42	6.35	7.42	5.28	7.03	6.51	5.44
e.	0.42	0.47	1.43	2.97	3.71	4.42	6.51	7.42	6.35	7.03	5.44	5.28
f.	1.43	3.71	5.44	6.35	6.51	7.42	0.42	0.47	2.97	4.42	5.28	7.03
g.	1.43	3.71	5.44	6.35	6.51	7.42	0.47	2.97	5.28	0.42	4.42	7.03
h.	1.43	3.71	5.44	6.35	6.51	7.42	0.47	2.97	5.28	7.03	0.42	4.42
i.	1.43	3.71	5.44	6.35	6.51	7.42	2.97	5.28	0.47	0.42	4.42	7.03
j.	1.43	3.71	5.44	6.35	6.51	7.42	2.97	5.28	0.47	7.03	0.42	4.42
k.	3.71	6.35	6.51	1.43	5.44	7.42	0.42	0.47	2.97	4.42	5.28	7.03
l.	3.71	6.35	6.51	1.43	5.44	7.42	0.47	2.97	5.28	0.42	4.42	7.03
m.	3.71	6.35	6.51	1.43	5.44	7.42	0.47	2.97	5.28	7.03	0.42	4.42
n.	3.71	6.35	6.51	1.43	5.44	7.42	2.97	5.28	0.47	0.42	4.42	7.03
o.	3.71	6.35	6.51	1.43	5.44	7.42	2.97	5.28	0.47	7.03	0.42	4.42
p.	3.71	6.35	6.51	5.44	1.43	7.42	0.42	0.47	2.97	4.42	5.28	7.03
q.	3.71	6.35	6.51	5.44	1.43	7.42	0.47	2.97	5.28	0.42	4.42	7.03
r.	3.71	6.35	6.51	5.44	1.43	7.42	0.47	2.97	5.28	7.03	0.42	4.42
s.	3.71	6.35	6.51	5.44	1.43	7.42	2.97	5.28	0.47	0.42	4.42	7.03
t.	3.71	6.35	6.51	5.44	1.43	7.42	2.97	5.28	0.47	7.03	0.42	4.42

CS302 Final Exam, December 11, 2018 - James S. Plank

Question 2

Suppose that I state the halting problem as in the box:

There are three important details about the halting problem that I have either omitted, or gotten incorrect. Please tell me what they are.

Let **P** be a procedure with the following prototype:

```
string P(string program, string inputfile);
```

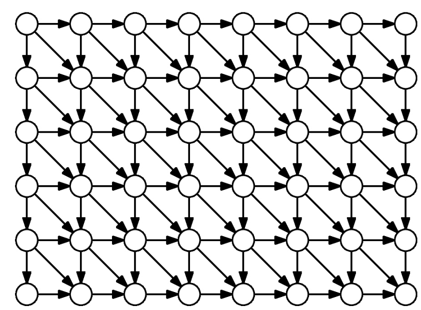
P takes as arguments a filename of a program written in C++, and a file which will be used as standard input to the program. **P** runs the program on the input and returns "halt" if the program halts on the input.

We don't know whether it is possible to write the procedure **P**.

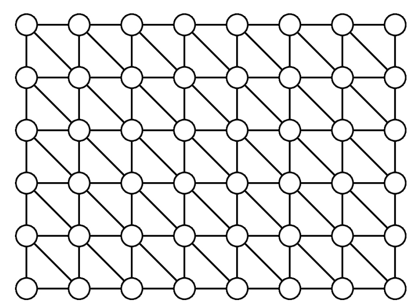
Question 3

Let's define the following graphs:

- *G* is an undirected, weighted, connected graph with *V* nodes and *E* edges.
- *H* is an directed, weighted, acyclic graph with *V* nodes and *E* edges.
- *J* is a weighted, directed graph with *V* nodes, defined as follows:
  - Let *r* and *c* be two integers such that $V = rc$.
  - The nodes are defined having labels (i,j) where *i* and *j* are integers, $0 \le i < r$, and $0 \le j < c$.
  - There is an edge from node (i,j) to node (k,l) if and only if:
    - $(k-i) \in \{0,1\}$
    - $(l-j) \in \{0,1\}$
    - $(i,j) \ne (k,l)$.
- *K* is an undirected, weighted graph with the same definition as *J*, except each edge is now undirected.



J, where *r=6* and *c=8*



K, where *r=6* and *c=8*

Please tell me the running times of the following algorithms, and use the choices on the answer sheet. Give me the most precise/clean answer. For example, if $O(E)$ is $O(V)$, then $O(V+E)$ is redundant. You can instead say $O(V)$.

- **A**: Finding the minimum spanning tree of *K*.
- **B**: Determining whether two nodes in *G* are connected.
- **C**: Finding the path with the smallest total weight, between two nodes in *H*.
- **D**: Finding the path with the smallest total weight, between two nodes in *J*.
- **E**: Finding the path with maximum flow between two nodes in *G*.
- **F**: Finding the path with the smallest total weight, between two nodes in *K*.
- **G**: Determining whether two nodes in *H* are connected.
- **H**: Finding the path with the smallest number of edges between two nodes in *G*.
- **I**: Finding the path with maximum flow between two nodes in *J*.
- **J**: Finding the minimum spanning tree of *G*.

CS302 Final Exam, December 11, 2018 - James S. Plank

Question 4

```
A: { D, F }
B: { E }
C: { B }
D: { C, E }
E: { C }
F: { C, D }
```

You are given a directed graph, defined by the adjacency lists to the right.

Part A: If you do a depth-first search from **A**, where the body of the depth-first search prints out the node's name, please tell me the order in which the nodes are printed.

Part B: If you do a breadth-first search from **A**, where the body of the breadth-first search prints out the node's name, please tell me the order in which the nodes are printed.

Part C: Suppose you remove the edge **BE** from the graph. If you do a topological sort from **A**, where the body of the topological sort prints out the node's name, please tell me the order in which the nodes are printed.

You are given an undirected graph, defined by the adjacency matrix to the right.

```
  |  0  1  2  3  4  5 |
--| -- -- -- -- -- -- |
0 |  - 15  9  -  - 28 |
1 | 15  -  1  7  3  - |
2 |  9  1  -  8 10  - |
3 |  -  7  8  -  -  7 |
4 |  -  3 10  -  - 12 |
5 | 28  -  -  7 12  - |
--| -- -- -- -- -- -- |
```

Part D: What is that length of the shortest path from node 0 to node 5?

Part E: You are using Dijstra's algorithm to find the shortest path from node 0 to node 5. In what order do you remove the nodes from the multimap?

Part F: Please list me the edges that compose the minimum spanning tree of this graph?

You are given a directed graph, whose adjacency matrix is to the right.

```
  |  0  1  2  3  4  5 |
--| -- -- -- -- -- -- |
0 |  - 30  -  -  -  - |
1 |  -  - 22  5  -  - |
2 | 12  -  - 13  8  - |
3 |  -  -  -  -  - 45 |
4 |  -  -  - 22  -  - |
5 |  -  -  -  - 17  - |
--| -- -- -- -- -- -- |
```

Part G: What is the maximum flow path from node 0 to node 5?

Part H: You are using the Edmonds-Karp algorithm to find the maximum flow of the graph. Show me the adjacency matrix of the residual graph after processing the first path in the algorithm. Do this on the answer sheet, and only fill in the squares of entries that change from the matrix to the right.

Part I: What is the minimum cut of the graph with respect to nodes 0 and 5?

Part J: What is the maximum flow from node 0 to node 5?

CS302 Final Exam, December 11, 2018 - James S. Plank

Question 5

You have been hired by a French company called *Brouillon Rois*, which allows a very specific kind of sports betting. For a given contest, there are **N** athletes, who each have an integer **cost** and a double **performance**. Your goal is to select **T** athletes such that their total cost is ≤ a budget **B** and their total performance is maximized.

Behold the following class definition:

```
class BR {
  public:
    int N;                                   // Total number of athletes.
    int B;                                   // Total budget.
    int T;                                   // Number of players to select.
    vector <int> C;                          // C[i] is the cost of athlete i.
    vector <double> P;                       // P[i] is the performance of athlete i.
    double BestTeam();                       // This calculates the best performance.
    double DP(int index, int b, int t);      // See below.
    map <string, double> Cache;              // A DP cache.
};
```

Assume that **N**, **B**, **T**, **C** and **P** have all been initialized for you, and that `C.size()` equals `P.size()` equals **N**. Your job at *Brouillon Rois* will be to write **BestTeam()**, which returns the total performance of the best team. To restate, the "best team" is the set of **T** players, whose total cost is ≤ **B**, and whose total performance is maximized.

This will be a dynamic program. It uses a helper method **DP()**, which returns the value of the best performing team composed of **t** players, whose total cost is ≤ **b**, and who have indices between **index** and **(N-1)**.

I'm not going to ask you to write **DP()**. However, I am going to ask the following things:

Part A: Write **BestTeam()**, assuming that **DP()** has been written correctly. You should assume that **BestTeam()** may get called multiple times, with **N**, **B**, **T**, **C** and **P** being reset each time.

Part B: Tell me what a good memoization string will be for **DP**. Write the code to create it and put into the variable **key**. You may assume that you have declared the following variables:

```
char buf[100];
stringstream ss;
string key, s;
```

Part C: What are the base case(s) that you think will apply to **DP()**. For each base case, tell me what it is, and what gets returned. Be specific here. If you want to write code for it, that's fine. You don't have to, though.

Part D: What recursive calls will you make in **DP()**? By that, I mean tell me the arguments to the recursive calls.

Part E: In terms of big-O, tell me the potential size of the memoization cache.