6.01 Final Exam Spring 2018

Name:

Kerberos (Athena) name:

Please WAIT until we tell you to begin.

During the exam, you may refer to any written or printed paper material.
You may NOT use any electronic devices (including calculators, phones, etc).

If you have questions, please **come to us at the front** to ask them.

Enter all answers in the boxes provided.

Extra work may be taken into account when assigning partial credit,
but **only work shown on pages with QR codes will be considered.**

Question 1:	32 Points
Question 2:	20 Points
Question 3:	24 Points
Question 4:	16 Points
Question 5:	32 Points
Question 6:	30 Points
Question 7:	25 Points
Total:	179 Points

1 Soul Searching (32 Points)

1.1 Part 1

Consider the problem of planning paths for a robot on a 24x24 grid. As in our design labs, we will represent a state in this search space as a tuple (r, c), where r is a row index $0 \leqslant r < 24$ and c is a column index $0 \leqslant c < 24$. There are no obstacles, so each cell is passable. Each state has children as defined by the function `robot_successors` shown below.

1.1.1 First Attempt

Ben Bitdiddle writes the following code in an attempt to solve this problem:

```
def robot_successors(s):
    #return a list of adjacent cells within the 24x24 grid
    r,c = s
    return [(r+i,c+j) for (i,j) in [(0,1),(0,-1),(1,0),(-1,0)] if 0<=r+i<24 and 0<=c+j<24]

def ben_search(start, goal_test, successors):
    agenda = [[start]]
    while len(agenda) > 0:
        current = agenda.pop(0)
        for child in successors(current[-1]):
            new = current + [child]
            if goal_test(child):
                return new
            agenda = agenda + [new]
    return None
```

Which of the following best describes Ben's search?

A. Depth-First-Search,

B. Breadth-First-Search, or

C. something else.

A, B, or C:

Which of the following best describes the elements in the agenda in Ben's code?

A. a boolean representing whether the robot is at the goal state,

B. a single state in the search space,

C. a list containing a single state in the search space, or

D. a list containing a path from the start state to some other state.

A, B, C, or D:

Ben also comes up with the following test cases for his code:

A. `result = ben_search((0,0), lambda x:  x==(0,10), robot_successors)`

B. `result = ben_search((0,0), lambda x:  x==(7,7), robot_successors)`

C. `result = ben_search((0,0), lambda x:  x==(0,0), robot_successors)`

D. `result = ben_search((0,0), lambda x:  x==(24,24), robot_successors)`

For which of the test cases does Ben's code return a valid path? Enter one or more letters, or `None` if Ben's code doesn't return a valid path in any of these cases.

For which of the test cases does Ben's code return the shortest possible path?

1.1.2 Dynamic Programming

Having heard about the merits of the dynamic programming approach to search and noticing that Ben's code takes a **long** time to run in many cases, Ben's friend Lem E. Tweakit suggests than Ben can improve his code by keeping track of which states have been visited, so as to avoid visiting the same state twice. Lem's code is shown below, where Ben's original code is shown as comments on the right, and where changes from Ben's code are highlighted in bold.

```
def lem_search(start, goal_test, successors):         #  def ben_search(start, goal_test, successors):
    agenda = [[start]]                                 #      agenda = [[start]]
    visited = [[start]]                                #
    while len(agenda) > 0:                             #      while len(agenda) > 0:
        current = agenda.pop(0)                        #          current = agenda.pop(0)
        for child in successors(current[-1]):          #          for child in successors(current[-1]):
            new = current + [child]                    #              new = current + [child]
            if goal_test(child):                       #              if goal_test(child):
                return new                             #                  return new
            if new not in visited:                     #
                agenda = agenda + [new]                #              agenda = agenda + [new]
                visited.append(new)                    #
    return None                                        #      return None
```

Will Lem's code work better, worse, or the same as Ben's code if the start state is $(0, 0)$ and the goal state is $(20, 0)$?

Better, Worse, or Same:

Briefly explain your reasoning:

1.2 Part 2

We will now consider expanding our search in this domain. We have some number of robots on an 24x24 grid, and we want to plan their motion using graph search. As in the previous section, we will make the assumption that each robot can move north, east, south, or west, but cannot move outside the boundaries of the 24x24 grid.

One important question in formulating problems as graph search problems is the choice of state. Remember the following about our search infrastructure:

- We used a successor function to determine the children of a state, using only the information stored in the state
- We also used a "goal test" function to determine whether a goal condition had been satisfied

We will say that a state representation is *correct* if, in conjunction with proper successor and goal test functions, it could be used to solve the search problem in question. Consider the following scenarios, and answer the questions about each.

1.2.1 Lonely Robot

In this scenario, there is only one robot in the grid, we want to move that robot to some goal location.

Consider the following representations for `state` in this scenario:

A. A number d representing the robot's distance from the goal

B. A tuple (x, y) representing the robot's position

C. A tuple (x, y, d) containing both the robot's position and the robot's distance from the goal

D. A list of tuples $[(x_0, y_0), \dots, (x_n, y_n)]$ representing robot's path from the starting point to its position

Which of the above, if any, are correct?

1.2.2 The Scenic Route

In this scenario, there is still only one robot, and we still want it to arrive at some goal location, but we want to make sure it visits some other location before arriving at the goal.

Consider the following representations for `state` in this scenario:

A. A tuple (x, y) representing the robot's position

B. A list of tuples $[(x_0, y_0), \dots, (x_n, y_n)]$ representing robot's path from its starting point to its position

C. A 2-D array of booleans, representing whether the robot has visited each location

D. A 2-D array of booleans representing whether the robot has visited each location, plus a tuple (x, y) representing the robot's current location

Which of the above, if any, are correct?

1.2.3 The *Really* Scenic Route

In this scenario, there is still one robot, but we want to plan a path such that it visits **every** location in the grid, without visiting the same location twice.

Consider the following representations for `state` in this scenario:

A. A tuple (x, y) representing the robot's position

B. A list of tuples $[(x_0, y_0), \dots, (x_n, y_n)]$ representing robot's path from its starting point to its position

C. A 2-D array of booleans, representing whether the robot has visited each location

D. A 2-D array of booleans representing whether the robot has visited each location, plus a tuple (x, y) representing the robot's current location

Which of the above, if any, are correct?

1.2.4 New Friend Request

In this scenario, there are two robots in the grid, and we want to plan actions such that they end up adjacent to one another.

Consider the following representations for `state` in this scenario:

A. Two tuples (x_0, y_0) and (x_1, y_1) representing each of the robots' positions

B. A boolean representing whether or not the robots are adjacent to one another

C. A 2-D array of booleans, representing whether each location is occupied by a robot

D. A number d representing the distance between the two robots

Which of the above, if any, are correct?

1.2.5 Flash Mob

In this scenario, there are N robots in the grid. Each of the N robots has its own goal location, and we want to plan actions for all the robots such that they all arrive at their respective goals.

Consider the following representations for `state` in this scenario:

A. N tuples $(x_0, y_0), \dots, (x_{N-1}, y_{N-1})$ representing each of the robots' positions

B. N booleans, each representing whether a certain robot is at its goal

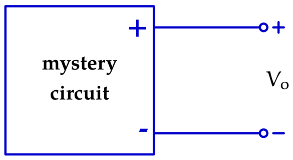
C. A 2-D array of integers, representing how many robots are in each location

D. N numbers $d_0, \dots, d_{N-1}$ representing each robot's distance from its goal

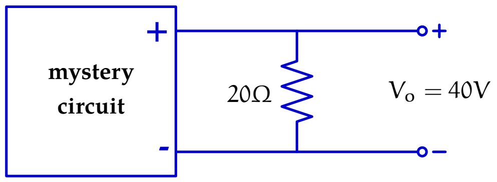
Which of the above, if any, are correct?

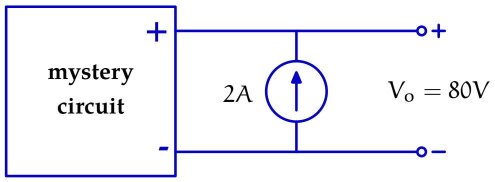
2 The Gift That Keeps on Giving (20 Points)

For your birthday, your friend gives you a mystery circuit made up of only linear components (just what you've always wanted!). The circuit is wrapped up so that you cannot see what is inside, but you can see and make use of two terminals.

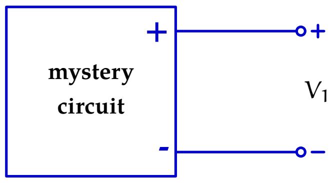


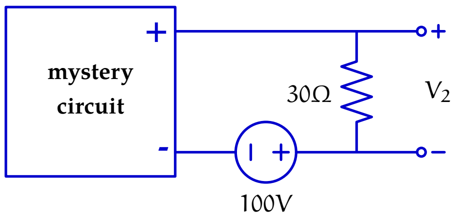
Curious, you connect the circuit to a 20Ω resistor and measure the voltage between the terminals as 40V. You also try connecting a 2A current source (as shown below), and you measure the voltage between the terminals as 80V.





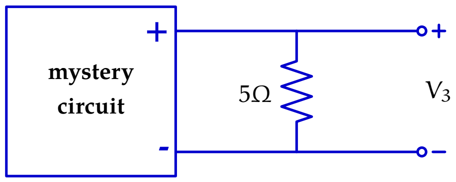
For each of the circuits below, solve for the indicated voltage. Make the ideal op-amp assumption, and ignore power supply limitations.

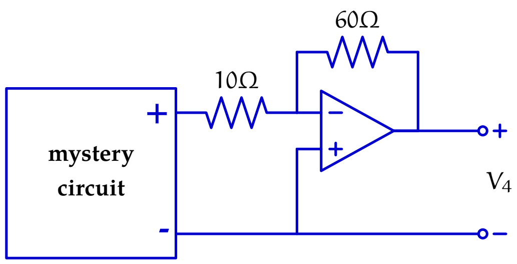




V_1 =

V_2 =



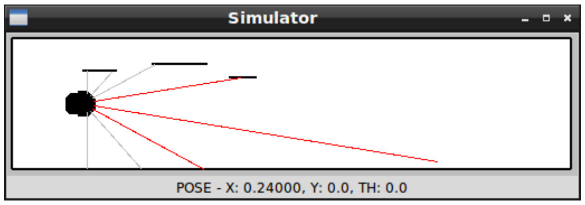


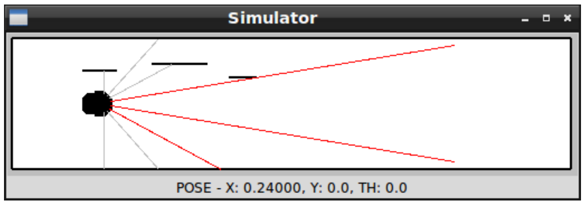
V_3 =

V_4 =

3 Onward and Rightward (24 Points)

In this problem, we consider state estimation as in Design Lab 11. A robot moves through a world with three walls, located at different distances. It starts at the location shown below (left), and ends slightly to the right of the starting location (right), as shown below.

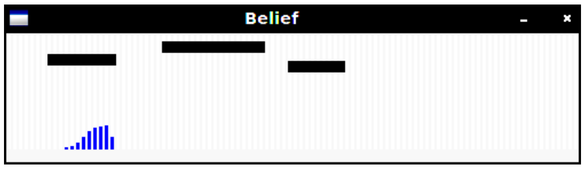




Although "we" know where the robot is located, the robot does not know where it is, and it will try to determine its horizontal postion by using the leftmost sonar sensor to determine the distance to the nearest wall.

Assume that the horizontal position of the robot is discretized as 100 states, and that the sonar readings are discretized into 30 values ranging from 0 to 29. The ideal readings are: 7 on the left-most wall, 9 on the center wall, and 5 on the right-most wall. The ideal reading in open space is 17. The robot moves forward at a constant velocity on every timestep, and the robot moves a total of twelve times.

We will represent the robot's belief as shown below, where the solid lines near the top of the figure represent the locations of walls in the world, and the bars in the lower portion of the figure represent the belief state, i.e., the height of each bar represents the probability that the robot is at the corresponding horizontal location.



3.1 Models

Consider the following transition models:

1. $P_1(S_{t+1} = s'|S_t = s) = \begin{cases} 0.9 & \text{if } s' = s+1 \\ 0.1 & \text{if } s' = s \end{cases}$

2. $P_2(S_{t+1} = s'|S_t = s) = \begin{cases} 0.9 & \text{if } s' = s-1 \\ 0.1 & \text{if } s' = s \end{cases}$

3. $P_3(S_{t+1} = s'|S_t = s) = \begin{cases} 0.5 & \text{if } s' = s-1 \\ 0.5 & \text{if } s' = s+1 \end{cases}$

Also consider the following observation models:

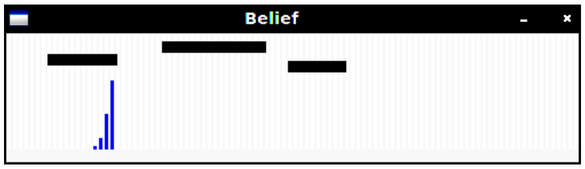
A. this model returns the ideal reading with probability 1

B. this model returns a triangular distribution with a *half-width* of 16, centered on the ideal reading

C. this model returns a uniform distribution of width 14, centered on the ideal reading

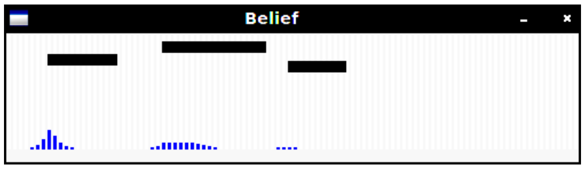
Determine the transition and observations models that would lead to each of the belief states shown below.

Belief 1



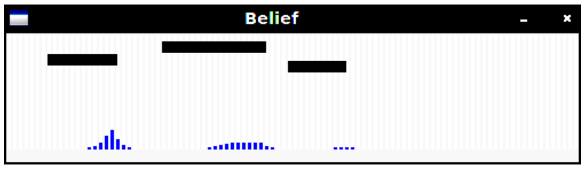
Transition Model (1, 2, or 3):		Observation Model (A, B, or C):	

Belief 2



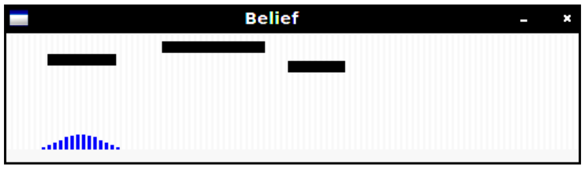
Transition Model (1, 2, or 3):		Observation Model (A, B, or C):	

Belief 3



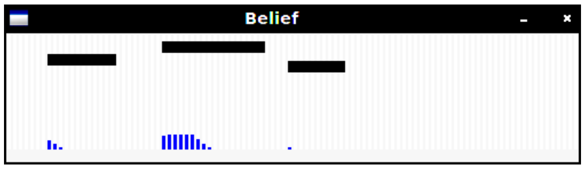
Transition Model (1, 2, or 3):		Observation Model (A, B, or C):	

Belief 4



Transition Model (1, 2, or 3):		Observation Model (A, B, or C):	

Belief 5

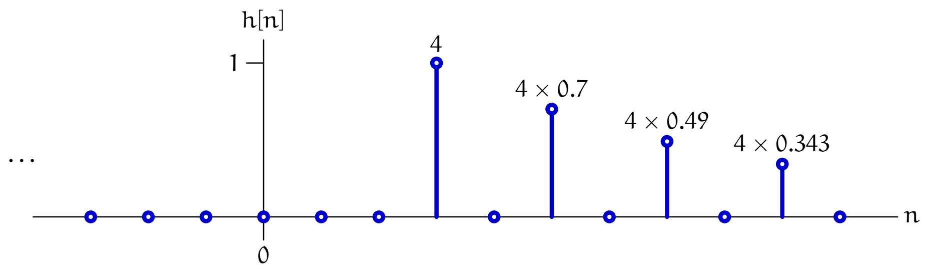


Transition Model (1, 2, or 3):		Observation Model (A, B, or C):	

4 Mystery System (16 points)

Consider a linear, time-invariant system whose unit-sample response $h[n]$ is shown below.

$$h[n] = \begin{cases} 4\left(0.7^{((n-3)/2)}\right) & \text{if } n \in \{3, 5, 7, 9, 11, 13, 15, \ldots\} \\ 0 & \text{otherwise} \end{cases}$$



4.1 Poles

Is it possible to represent this system with a finite number of poles?

Yes or No:

If yes, enter the number of poles and list the poles below. If a pole is repeated k times, then enter that pole k times. If there are more than 5 poles, enter any 5 poles. If there are fewer than 5 poles, write `None` in the remaining boxes.

Number of Poles:

Poles:

If no, explain briefly:

4.2 Diagram

Is it possible to represent this system using finitely-many gains, delays and adders (and no other components)?

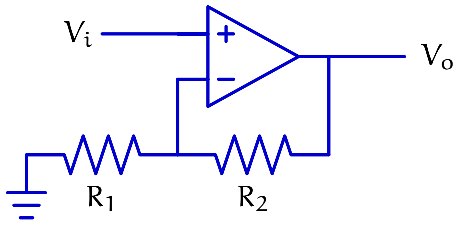
Yes or No:

If yes, draw a block diagram of the system:

If no, explain briefly:

5 Circuit Diagnosis (32 Points)

Acme Inc. manufactures circuits corresponding to the design below. The design calls for $R_1 = 2k\Omega$ and $R_2 = 1k\Omega$ resistors.



Unfortunately, Acme sometimes constructs faulty circuits.

With probability 0.06, the robot that assembles the circuit will accidentally use a $1k\Omega$ resistor instead of a $2k\Omega$ resistor for R_1. Also, with probability 0.06, the R_2 resistor is shorted (has a resistance of 0Ω instead of $1k\Omega$).

Our challenge is to help Acme diagnose their circuits, based on the measurement of an output voltage V_o given an input test voltage V_i.

Acme uses a test voltage $V_i = 1V$ to check each circuit. For each case below, the following voltages are measured:

- Good circuit, R_1 is correct and R_2 is not shorted: $V_o = 1.5V$
- R_1 is correct but R_2 is shorted: $V_o = 1V$
- R_1 is the wrong value, and R_2 is not shorted: $V_o = 2V$
- R_1 is the wrong value, and R_2 is shorted: $V_o = 1V$

5.1 Testing

Acme notices that, with the test voltage of $V_i = 1V$, it cannot distinguish unambiguously between all four of the cases above, and they ask for advice on a better test voltage so they can distinguish between all four cases. Suggest a test voltage V_i that will separate these four cases. If no better test voltage exists, explain why not.

Is there a better test voltage? Circle one: **Yes** / **No**

If yes, what is the value? If no, briefly explain why (1-2 sentences):

5.2 Two Wrongs Don't Make a Right

Based on historical data, Acme knows that 90% of their circuits work correctly, 6% of their circuits have the wrong R_1 as described above, and 6% of their circuits have a shorted R_2. What percentage of their circuits have **both** the wrong R_1 and a shorted R_2?

$\boxed{}$ percent have both failures (wrong R_1 and shorted R_2)

5.3 Results

Acme tests a circuit using a test voltage $V_i = 1V$, and measures $V_o = 1V$. Given that measurement, what are the probabilities below associated with that circuit?

Good circuit, R_1 is correct and R_2 is not shorted: $\boxed{}$

R_1 is correct but R_2 is shorted: $\boxed{}$

R_1 is the wrong value, and R_2 is not shorted: $\boxed{}$

R_1 is the wrong value, and R_2 is shorted: $\boxed{}$

What is the probability that R_1 is correct? $\boxed{}$

What is the probability that R_2 is shorted? $\boxed{}$

5.4 Faulty Tester

Acme starts using a different volt-meter to measure the output of their circuits. This meter outputs a value of 1V with probability 0.1 (regardless of the value measured), and it outputs the actual value measured with probability 0.9.

When measuring V_o on a random circuit, the volt-meter reads 1V. Given this output, what are the probabilities below? Express your answers as fractions.

Good circuit, R_1 is correct and R_2 is not shorted:

R_1 is correct but R_2 is shorted:

R_1 is the wrong value, and R_2 is not shorted:

R_1 is the wrong value, and R_2 is shorted:

What is the probability that R_1 is correct?

What is the probability that R_2 is shorted?

5.5 Testing

Using the volt-meter described in Part 4, what is a better test voltage V_i to apply to the circuit, to better distinguish between the good and bad circuit cases? If no V_i voltage can do better than $V_i = 1V$ using this volt-meter, explain why not.

Is there a better test voltage? Circle one: **Yes**/ **No**

If yes, what is the value? If no, briefly explain why (1-2 sentences):

5.6 Limited Warranty

Acme ships a circuit to its best customer, BestAmp Corporation, that does indeed have the correct R_1 resistor, and in which the R_2 resistor has a resistance of $1k\Omega$ right after assembly.

Assume that each month after assembly there is a 0.01 probability that an R_2 resistor that was okay at the start of that month will short out and have zero resistance by the end of that month.

What is the probability that the circuit fails in the first month?

What is the probability that the circuit fails during the second month?

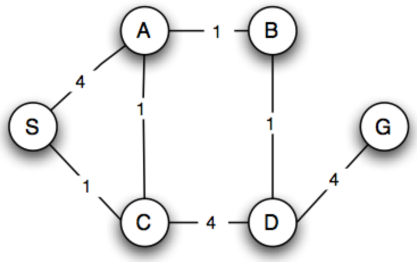
What is the probability that the circuit is still good at the end of the second month?

5.7 Returns Policy

BestAmp ships the circuit back to Acme one month after it was assembled. The circuit was known to be functioning correctly right after assembly, but is in unknown condition now. Acme checks the device on the volt-meter from part 4. What is the probability that the meter reports 1V when $V_i = 1V$ is applied to the circuit?

6 Search (30 Points)

Consider searching in the following graph:



The start node is S and the goal node is G.

Assume that states are visited (pushed on the agenda) in reverse alphabetical order. Also, use alphabetical order to settle ties in costs. (For example if two nodes have the same cost, the node whose ending state comes earliest in the alphabet is popped first.)

Give the sequence of states expanded (removed from agenda, in order of expansion) during each of the following searches.

Also, give the final path found, as a sequence of states, and the cost of that path.

1. Depth first, with dynamic programming

Expanded states (in order):

Final path: Cost of final path:

2. Breadth first, with dynamic programming

Expanded states (in order):

Final path: Cost of final path:

3. Uniform cost, with dynamic programming

Expanded states (in order):

Final path: Cost of final path:

4. Weighted A* with $\alpha = 1$ (use only heuristic) with dynamic programming, using the following heuristic estimates of the distance to `G`: {`S:1, A:1, B:1, C:8, D:1, G:0`}

Expanded states (in order):

Final path: Cost of final path:

5. A* with dynamic programming, using the following heuristic estimates of the distance to `G`: {`S:1, A:1, B:1, C:8, D:1, G:0`}

Expanded states (in order):

Final path: Cost of final path:

7 Motor Control (25 Points)

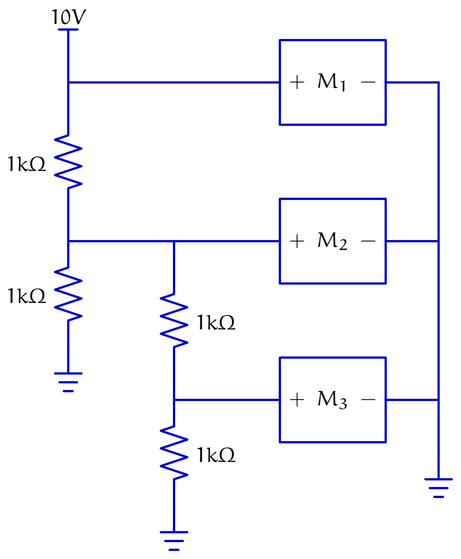
Kim, Pat, Jody, Chris, and Jamie are trying to design a controller for an animatronic display of three dancing robots using a 10V power supply and three motors. The first robot is supposed to spin as fast as possible (in one direction only), the second at half the speed of the first, and the third at half the speed of the second.

Each engineer has come up with a design, as shown below. Assume the motors have a resistance of approximately 5Ω, and that the rotational speed of a motor is proportional to the voltage drop across is.

Let M_1 be the motor associated with robot 1, M_2 be the motor associated with robot 2, and M_3 be the motor associated with robot 3. Let V_1, V_2, and V_3 be the voltage drops across these motors, respectively.

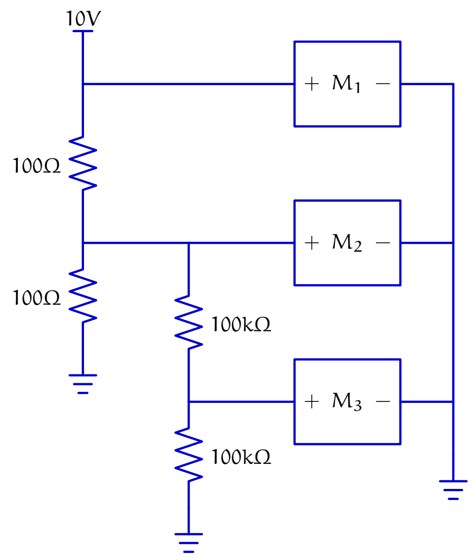
For each design, indicate the voltage drop across each of the motors. Your answers for integer values should be exact, and your other answers should be accurate to within 5% of the true value (you do not need to solve for the exact value).

7.1 Jody



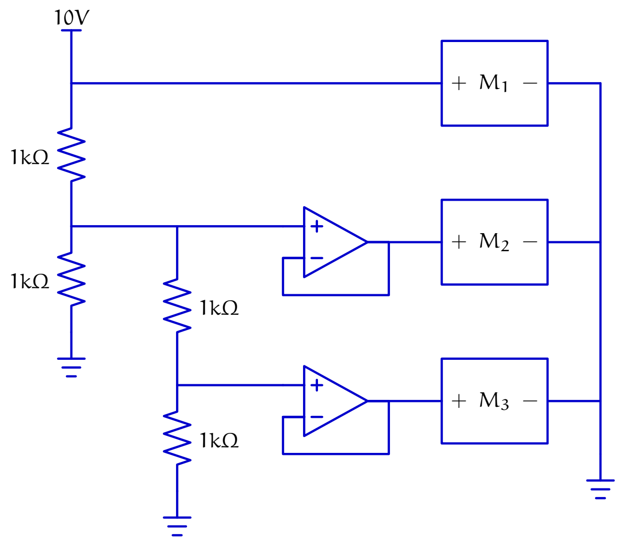
$V_1 =$ $\boxed{}$ $V_2 =$ $\boxed{}$ $V_3 =$ $\boxed{}$

7.2 Chris



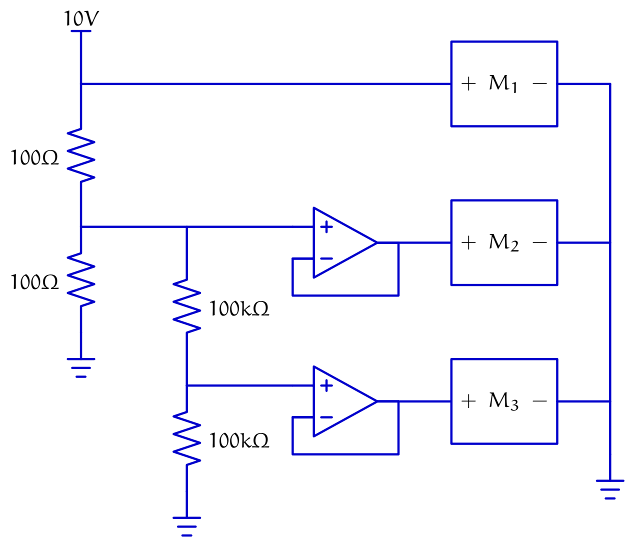
V_1 = V_2 = V_3 =

7.3 Pat



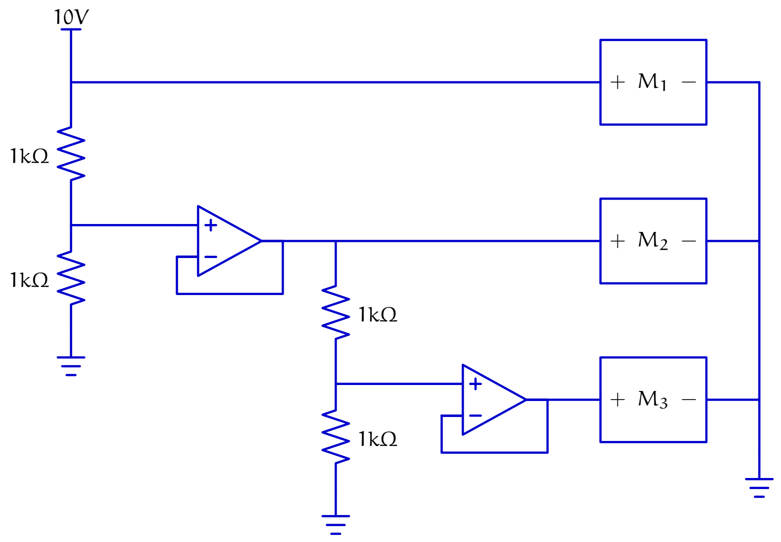
V_1 = V_2 = V_3 =

7.4 Kim



$V_1 =$ $V_2 =$ $V_3 =$

7.5 Jamie



$V_1 =$ $V_2 =$ $V_3 =$

Worksheet (intentionally blank)

Worksheet (intentionally blank)

Worksheet (intentionally blank)

Worksheet (intentionally blank)

Worksheet (intentionally blank)

Worksheet (intentionally blank)

Worksheet (intentionally blank)

Worksheet (intentionally blank)

Worksheet (intentionally blank)

Worksheet (intentionally blank)