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.
6 Flood Fill (18 Points)

In computer drawing programs, a "paint bucket" tool is often provided, which re-colors all the cells of a particular color in an enclosed region. When a user clicks on a pixel, that pixel and all surrounding pixels that share the original color are replaced with a different color. Typically, this is implemented with a "flood fill" algorithm that closely resembles the graph search algorithms we have seen in 6.01.

For now, we will assume that an image is represented as a 2-dimensional array (a list of lists) called \texttt{image}, where \texttt{image[r][c]} returns the color of the pixel in row \texttt{r} and column \texttt{c}, \texttt{image.height} is the number of rows in the image, and \texttt{image.width} is the number of columns in the image. Also, assume that the color that was originally clicked is stored as \texttt{clicked_color}, that the desired color is stored as \texttt{fill_color}, and that a copy of the original image (before the flood fill) is stored as \texttt{original_image}.

Consider three possible successor functions for a search in this space:

```python
01 def color_successor_1(point):
02 row, col = point
03 neighbors = []
04 for (dr, dc) in [(0,1), (1,0), (0,-1), (-1,0)]:
05     new_r = row+dr
06     new_c = col+dc
07     if 0 <= new_r < image.height and 0 <= new_c < image.width:
08         image[new_r][new_c] = fill_color
09         if original_image[new_r][new_c] == clicked_color:
10             neighbors.append((new_r, new_c))
11     return neighbors
```

In \texttt{color_successor}_2, lines 8-10 instead read:

```python
08     if original_image[new_r][new_c] == clicked_color:
09         image[new_r][new_c] = fill_color
10     neighbors.append((new_r, new_c))
```

In \texttt{color_successor}_3, lines 8-10 instead read:

```python
08     if original_image[new_r][new_c] == clicked_color:
09         image[new_r][new_c] = fill_color
10     neighbors.append((new_r, new_c))
```

Three possible goal tests:

```python
def goal_test_a(state):
    return True
def goal_test_b(state):
    return False
def goal_test_c(state):
    return original_image[state[0]][state[1]] != clicked_color
```

And two possible search functions, each implemented as in 11b601 (note that the source code of the search function is available on the last page of this exam, which you may remove):

i. BFS with dynamic programming

ii. DFS with dynamic programming
Consider all possible combinations of the successor functions, goal test, and search. Each combination is represented by:

- a number (1-3) representing the successor function,
- a letter (a-c) representing the goal test function, and
- a roman numeral (i or ii) representing the search function.

For each combination, enter all three values above in one of the boxes below the image that would result from running the search with that combination of inputs to completion, starting from the location labeled "S". For example, if an image would result from running with successor function 1, goal test a, and search i, enter (1, a, i) in one of the boxes below that image.

If there are more boxes than you need, leave the remaining boxes blank. If there are too few boxes, enter any subset of the valid answers in the boxes.
1 Civilization (9 Points)

Imagine that you are the ruler of a very simple civilization, and you want to build a vast pyramid as a monument to yourself. The state of the civilization can be represented as a tuple \((p, f, h)\) where \(p\) is the number of people in the society, \(f\) the number of units of food they have, and \(h\) is the height of their pyramid.

As ruler, on any given day, you can give your people one of three possible commands: build, farm, or wait. The new state that results from executing one of these commands is completely determined, as described by the successor function below:

```python
def civ_successors(s):
    return [farm_successors(s), wait_successors(s), build_successors(s)]
```

```python
def farm_successors(s):
    (people, food, height) = s
    new_people = min(people, food)
    new_food = food + people
    new_height = height
    return (new_people, new_food, new_height)
```

```python
def wait_successors(s):
    (people, food, height) = s
    new_people = min(people, food) * 2
    new_food = max(food - people, 0)
    new_height = height
    return (new_people, new_food, new_height)
```

```python
def build_successors(s):
    (people, food, height) = s
    new_people = min(people, food)
    new_food = max(food - 2 * people, 0)
    new_height = height + int(people / 10)
    return (new_people, new_food, new_height)
```

1. What are the successors of state \((22, 10, 0)\), using the `civ_successors` function?
2. Assuming that each action has a cost of 1, which of the following heuristics is admissible for the goal of reaching a pyramid with a height of exactly 3?
Circle those heuristics that are admissible.

A. def heuristic_a(s):
   return 0
B. def heuristic_b(s):
   return 1
C. def heuristic_c(s):
   return s[2]
D. def heuristic_d(s):
   return max(0, 3 - s[2])
E. def heuristic_e(s):
   return max(0, s[2] - 3)
F. def heuristic_f(s):
   return s[0] / 10
G. def heuristic_g(s):
   return s[0] / 10 + max(0, 3 - s[2])
H. def heuristic_h(s):
   return max(0, 3 - s[2] - int(s[0] / 10))

3. Of the heuristics above that are admissible, which one would make the search most efficient (make the search expand the smallest number of states)?

Enter one of A-H: [ ]
2 New Phone (13 Points)

You are working for a cell phone company and are studying your new cell phone model to compete with the iPhone 6s:

2.1 First in Customer Satisfaction

When making a phone call with this phone, the call can be in one of three states: clear, noisy, or interrupted.

Each minute, the state of a phone call can change, however.

After studying the characteristics of the cell phone and analyzing many phone calls with it, you’ve developed the following transition model to describe the state of phone calls made with this cell phone:

```python
import lib601.dist as dist

def callStatus(s):
    if s == 'clear':
        return dist.DDist({'clear':0.7, 'noisy':0.3, 'interrupted':0})
    if s == 'noisy':
        return dist.DDist({'clear':0, 'noisy':0.8, 'interrupted':0.2})
    else: # s == 'interrupted'
        return dist.DDist({'clear':0, 'noisy':0.4, 'interrupted':0.6})
```

Assume you start with a known ‘clear’ phone call at minute 0.

What is the distribution over call quality at minute 3?

Pr(clear) =  
Pr(noisy) =  
Pr(interrupted) =  

Your long-lost brother calls, and you spent a long time talking (approaching infinity). What is the distribution over all quality at this point?

Pr(clear) =  
Pr(noisy) =  
Pr(interrupted) =  
2.2 Highest 4G LTE Call Quality

As you are wandering around testing the call quality of your new phone¹, you get lost in a small region of rural Illinois. You know you are in one of 5 towns: Princeton, Kasbeer, Malden, Dover, or Zearing. Having driven by Timber’s Edge Alpaca Farm recently, you think there is a 60% chance that you are in Kasbeer, and a 10% chance that you are in each of the other four towns.

You also happen to have a local map, which marks the advertised signal strengths for each of these towns (Princeton, which has a cell tower, has the highest strength at 4 bars):

![Map of signal strengths](image)

You know that cell carriers tend to overstate their signal strength, so you assume that, if the advertised signal strength in a city is \( n \), then your phone will actually measure:
- \( n - 1 \) bars with probability 0.5
- \( n \) bars with probability 0.3
- \( n + 1 \) bars with probability 0.2

You look down at your phone and see that it is measuring a signal strength of 3 bars. What is your updated distribution over your location?

\[
\begin{align*}
\Pr(\text{Zearing}) & \quad \Pr(\text{Dover}) & \quad \Pr(\text{Malden}) \\
\Pr(\text{Kasbeer}) & \quad \Pr(\text{Princeton})
\end{align*}
\]

How many times will you have to measure 3 bars before you think it is more likely that you are in Dover than in Princeton? You may assume that you never measure anything other than 3 bars. Explain briefly.

---

¹ Can you hear me now? Good!


7 In the Mystical Moist Night-air (15 Points)

Seven astronomers are looking at the same region of the night sky through their telescopes, trying to determine how many stars there are in that region.

Each astronomer’s telescope has some problems: they could go out of focus, display elements that are not stars, fail to display a star, etc.

Each astronomer builds up their own observation model for the telescope they are using, which describes the number of “bright dots” \(D\) they expect to observe, given the number of stars \(s\) that actually exist in the region. These models are shown in the table below:

<table>
<thead>
<tr>
<th>Astronomer</th>
<th>(\Pr(D = s + 2))</th>
<th>(\Pr(D = s + 1))</th>
<th>(\Pr(D = s))</th>
<th>(\Pr(D = s - 1))</th>
<th>(\Pr(D = s - 2))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Andy</td>
<td>0</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td>Becky</td>
<td>0.001</td>
<td>0.001</td>
<td>0.996</td>
<td>0.001</td>
<td>0.001</td>
</tr>
<tr>
<td>Chad</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Diane</td>
<td>0.15</td>
<td>0.15</td>
<td>0.4</td>
<td>0.15</td>
<td>0.15</td>
</tr>
<tr>
<td>Erick</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Francine</td>
<td>0.1</td>
<td>0.1</td>
<td>0.7</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>Gerald</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.6</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Each of these astronomers takes a sequence of 20 pictures and observes the same sequence of bright pixels per image: 5, 5, 7, 6, 6, 7, 6, 5, 7, 7, 5, 7, 6, 5, 5, 5, 7, 7, 7, 6

The graphs on the facing page show the each astronomer’s belief that there are actually \(S = 6\) stars in the region being photographed. Assume that each astronomer initially thought it was equally likely that there were 5, 6, or 7 stars in the image, and that there was no chance that there was any other number of stars. Match each graph with the associated astronomer by putting the first letter of their name next to the matching graph.

Notice also that one of the astronomers is missing. This astronomer was unable to appropriately update their belief based on the evidence presented in the photographs. Which astronomer was this? Briefly explain why they were unable to update their belief.

Missing Astronomer:

Explain briefly (1-2 sentences):
2 Delay Removal (17 Points)

Your company is interested in ensuring that the system shown below converges to its final value as soon as possible. It has enough money to engineer one of the delays out of the system. Your job is to determine which delay, if removed, would result in the fastest convergence.

Here is the original system.

2.1 Analysis

Let $\mathcal{H}_1$ represent the system that results when $R_1$ is a wire, $R_2$ is a delay, and $R_3$ is a delay. Let $\mathcal{H}_2$ represent the system that results when $R_2$ is a wire, $R_1$ is a delay, and $R_3$ is a delay. Let $\mathcal{H}_3$ represent the system that results when $R_3$ is a wire, $R_1$ is a delay, and $R_2$ is a delay.

Fill in the following table with properties of $\mathcal{H}_1$, $\mathcal{H}_2$, and $\mathcal{H}_3$.

<table>
<thead>
<tr>
<th>system functional</th>
<th>pole(s)</th>
<th>converge?</th>
<th>oscillate?</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}_1 =$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathcal{H}_2 =$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathcal{H}_3 =$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2.2 Recommendation

Which delay should your company remove?

1 or 2 or 3 or none: 

Briefly explain (1-2 sentences):
4 Circuits (20 points)

A. Dana, Emerson, and Flann have a supply of +10 V and want to make buffered supplies of 2.5 V, 5.0 V and 7.5 V for use in their circuit, but they are having a disagreement about how to do it. Here is the circuit Dana suggests:

1. In Dana’s circuit, what are the actual values of $V_1$, $V_2$, and $V_3$?

$$V_1 = \underline{\hspace{2cm}} \quad V_2 = \underline{\hspace{2cm}} \quad V_3 = \underline{\hspace{2cm}}$$

If they are incorrect, can you change the resistor values to make it work? If so, how?
Here is the circuit that Emerson suggests:

2. In Emerson’s circuit, what are the actual values of $V_1$, $V_2$, and $V_3$?

$V_1 = \underline{\phantom{0000}}$  $V_2 = \underline{\phantom{0000}}$  $V_3 = \underline{\phantom{0000}}$

If they are incorrect, can you change the resistor values to make it work? If so, how?
Here is the circuit that Flann suggests:

3. In Flann’s circuit, what are the actual values of $V_1$, $V_2$, and $V_3$?

$V_1 =$ \hspace{2cm} $V_2 =$ \hspace{2cm} $V_3 =$

If they are incorrect, can you change the resistor values to make it work? If so, how?
B. Consider the following circuit

1. Write a formula for $V_+$ (the voltage on the positive input to the op-amp) in terms of $V_{\text{in}}$ and the resistor values for this circuit:

$V_+ =$
2. Write a formula for $V_{\text{out}}$ in terms of $V_+$ and the resistor values for this circuit:

$V_{\text{out}} =$

3. For each of these relationships, state whether it is possible to choose resistor values that make it hold in the circuit above. Write Yes or No; it is not necessary to provide the resistor values.

   a. $V_{\text{out}} = 2.5 - \frac{3}{16} V_\text{in}$
   b. $V_{\text{out}} = 2.5 + \frac{3}{16} V_\text{in}$
   c. $V_{\text{out}} = -2.5 + \frac{3}{16} V_\text{in}$