Allen Scholars Bridge Quiz #3 (Sample)

1. (1 pt) Matching

Fill in the blanks in the story with the appropriate words/phrases we learned. Some possible choices are listed to the left. (.25 pts each)

graphic design
participatory design
low fidelity prototype
contextual inquiry
wireframe
human computer interaction
hi fidelity design
use cases
web programming

Cameron is thinking of building a website for their cafe and selling some of the home made chocolates they make there. They draw out a rough **wireframe** sketch of each of the web pages of the site on paper, and think through the navigation of the links. Cameron then takes the **low fidelity prototype** to their mother, their younger sibling, and their best friend to use **participatory design** to observe their interactions and hear feedback. Based on the feedback, Cameron implements the site, then returns to watch the same three people use the site in their natural environment using a technique known as **contextual inquiry**.

2. (1 pt) Connecting JS to HTML. You are trying to log to the console `"I was clicked!"` every time a button on a page is clicked. Here is the HTML for the button:

```
<button class="btn-outline" id="clicker">Click me!</button>
```

Which of the following pieces of code accomplishes this task correctly?
You may assume that it is in the `appropriate window.onload` boilerplate.

A.
```
let buttonElement = document.getElementById("clicker");
buttonElement.onClick = function() {
  console.log("I was clicked!");
}
```

B.
```
let buttonElement = document.getElementById("btn-outline");
buttonElement.onClick = function() {
  console.log("I was clicked!");
}
```

C.
```
let buttonElement = document.getElementById("clicker");
buttonElement.onclick = function() {
  console.log("I was clicked!");
}
```

D.
```
let buttonElement = document.getElementById("btn-outline");
buttonElement.onclick = function() {
  console.log("I was clicked!");
}
```

3. (1.5 pts) Code Tracing. Consider the following function:

```
function mystery(n) {
  let num = n;
  while (num > 1) {
    console.log(num);
    num = num / 2;
    console.log(num);
    num = num - 1;
  }
  console.log(num);
}
```

Part A (0.5 pts)

What output would display in the console if the following function call was made?

```
mystery(0);
```

```
0
```

Part B (0.5 pts)

What output would display in the console if the following function call was made?

```
mystery(4);
```

```
4
2
1
```

Part 2 (0.5 pts)

What output would display in the console if the following function call was made?

```
mystery(14);
```

```
14
7
6
3
2
1
0
```

4. (1.5 pts) Debugging. Lauren and Matt are making their own app for Project 3: a bird feeding app! Depending on the time of day, the app is supposed to tell them whether or not they should feed the bird, and how much. The following table explains their logic:

Time of day	Action
12am - 7am	No bird feeding (Lauren, Matt, and the bird are all asleep!)
7am - 9am	Feed the bird one snack (bird breakfast)
9am - 3pm	No bird feeding (it's bridge!)
3pm - 4pm	Feed the bird one snack (bird late lunch)
4pm - 6pm	No bird feeding (office hours)
6pm - 8pm	Feed the bird two snacks (bird dinner)
8pm - 12am	No bird feeding (bird is sleeping, Lauren and Matt are grading)

Here is their JavaScript function that tries to represent the table: it takes in the hour (as a whole number, from 1-12) and whether or not it is AM (as a boolean), and returns the number of snacks Matt and Lauren should feed the bird at that time.

```
1   function snacksToGiveBird(hour, isAM) {
2     if (isAM) {
3       if (hour < 7) {
4         return 0;
5       } else {
6         return 1;
7       }
8       if (hour >= 9) {
9         return 0;
10      }
11    } else if (hour === 3) {
12      return 1;
13    } else if (hour === 6) {
14      return 2;
15    } else {
16      return 0;
17    }
18  }
```

But, Matt and Lauren were super tired (and also hungry) and wrote many bugs :(
Help them debug!

Part A: Matt and Lauren first tested their code with this function call:

```
snacksToGiveBird(10, true);
```

This *should* return `0` (since Matt & Lauren are teaching), but instead returns `1`.

What line(s) of code are causing this bug? Describe why this is happening, then suggest a fix.

Line: 5

Bug description: hour is 10, so it's not less than 7. That means it will take this else branch and return 1. But, according the above table, we should be returning 0.

Fix: change Line 5 to be `} if (hour <= 9) {`

Note: there are many different "valid" lines and fixes for this question; the rubric would enumerate all of them, but the important thing is that your line would have to match your bug description.

Part B: There is one other bug in the function. What function call would demonstrate this bug (e.g. similar to the one shown in Part A)? How would you then fix the bug?

Line: 15

Bug description: if hour is 7, we should be feeding the bird 2 snacks - but we aren't!

Fix: add another else-if statement for hour === 7 (after the last else if, but before the else), which should then return 2

Note: there are many different "valid" lines and fixes for this question; the rubric would enumerate all of them, but the important thing is that your line would have to match your bug description.