Assignment 2: Static Web: HTML/CSS

Due Sunday, January 9 11:59pm PST

Setup

Accept the Github Classroom assignment and clone this repo that contains stencil code for Assignment 2.

Introduction

This is a multi-part assignment with the objective of making you comfortable working with HTML and CSS. By the end of this assignment, you will have styled some rectangular blocks and created a simple version of Twitter's home page.

If this assignment seems overwhelming to you, please come see a TA at TA hours to talk through some strategies for tackling it. We expect this assignment to be a time-consuming assignment as we cover a lot of fundamental techiniques. But with a good strategy, it can be finished in a reasonable amount of time.

Note: Only CSS and HTML will be used for this assignment. If you want to use JavaScript (or libraries such as jQuery) then feel free to, but we will only be grading correctness on your CSS and HTML.

If you can, **Start Early!**

Part One

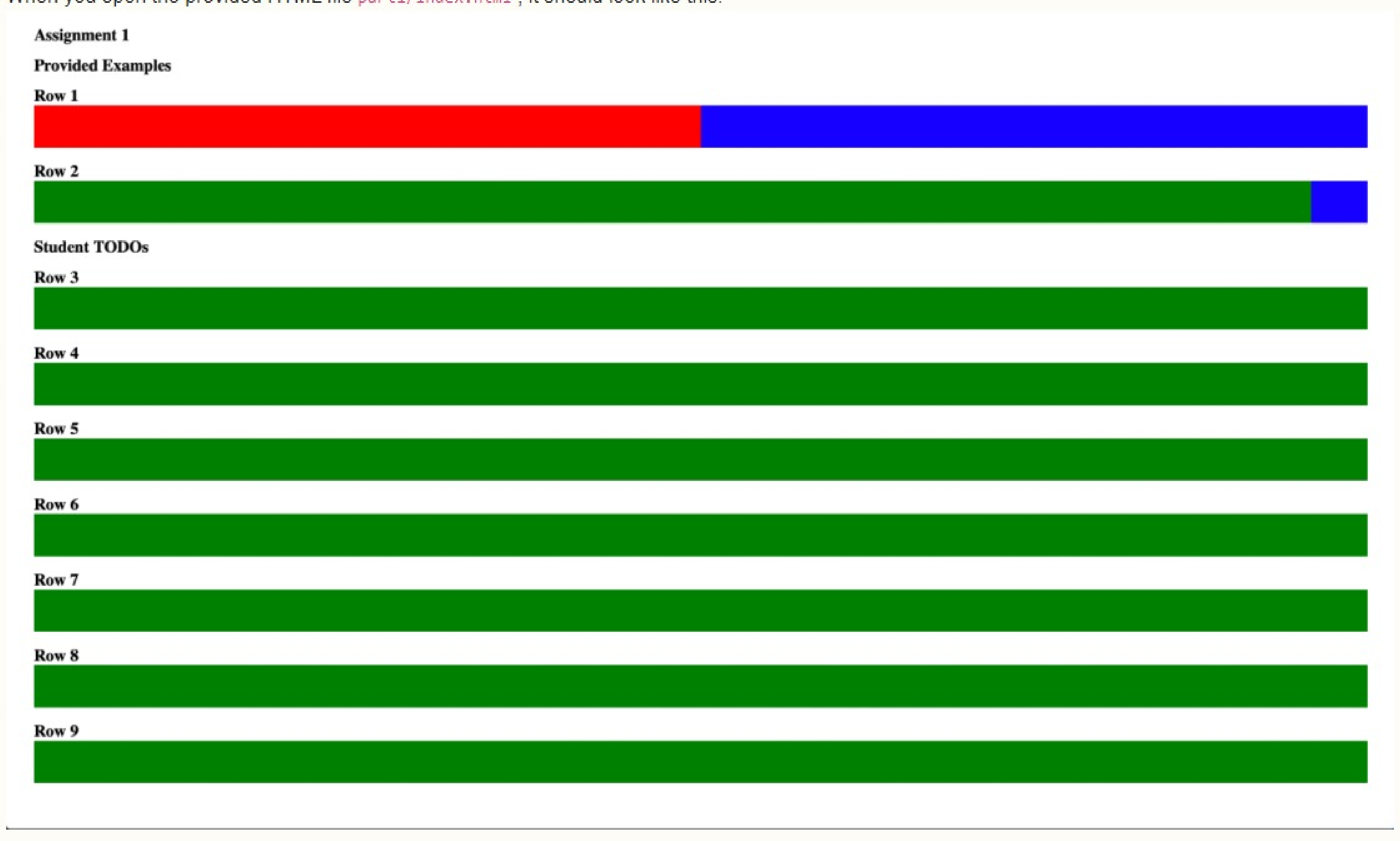
Specifications

Now that you understand some of the basics of HTML and CSS, let's take a look at how to align HTML elements. There are multiple ways to align HTML elements, but in this part, we recommend using flexboxes as they are widely used in modern web development (for example BootstrapV4 is built on top of flexboxes).

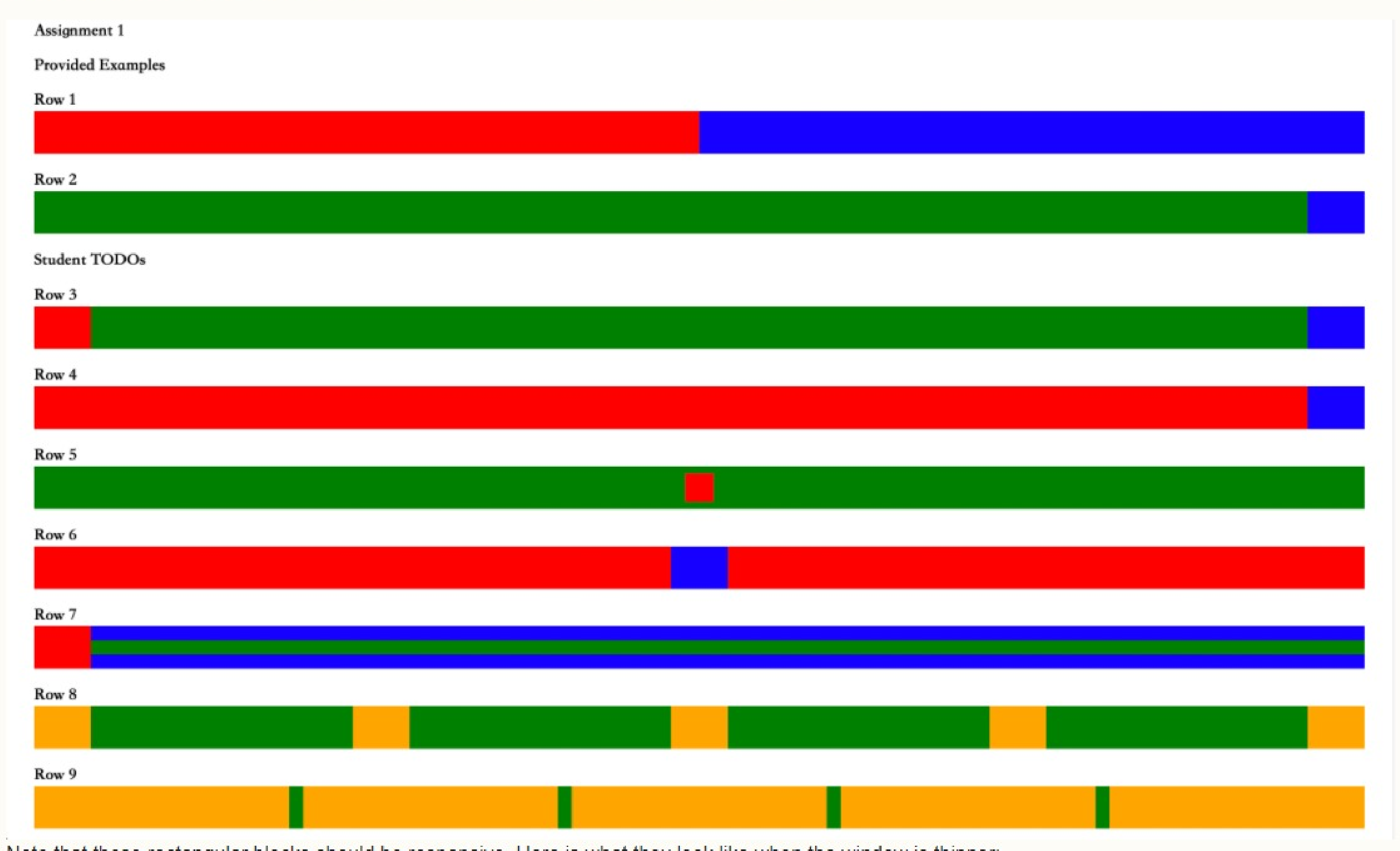
Refer to this great webpage on how to use flexboxes: CSS Flexbox Guide .

Also feel free to use online resources such as Stack Overflow, MDN, W3, and Google for reference.

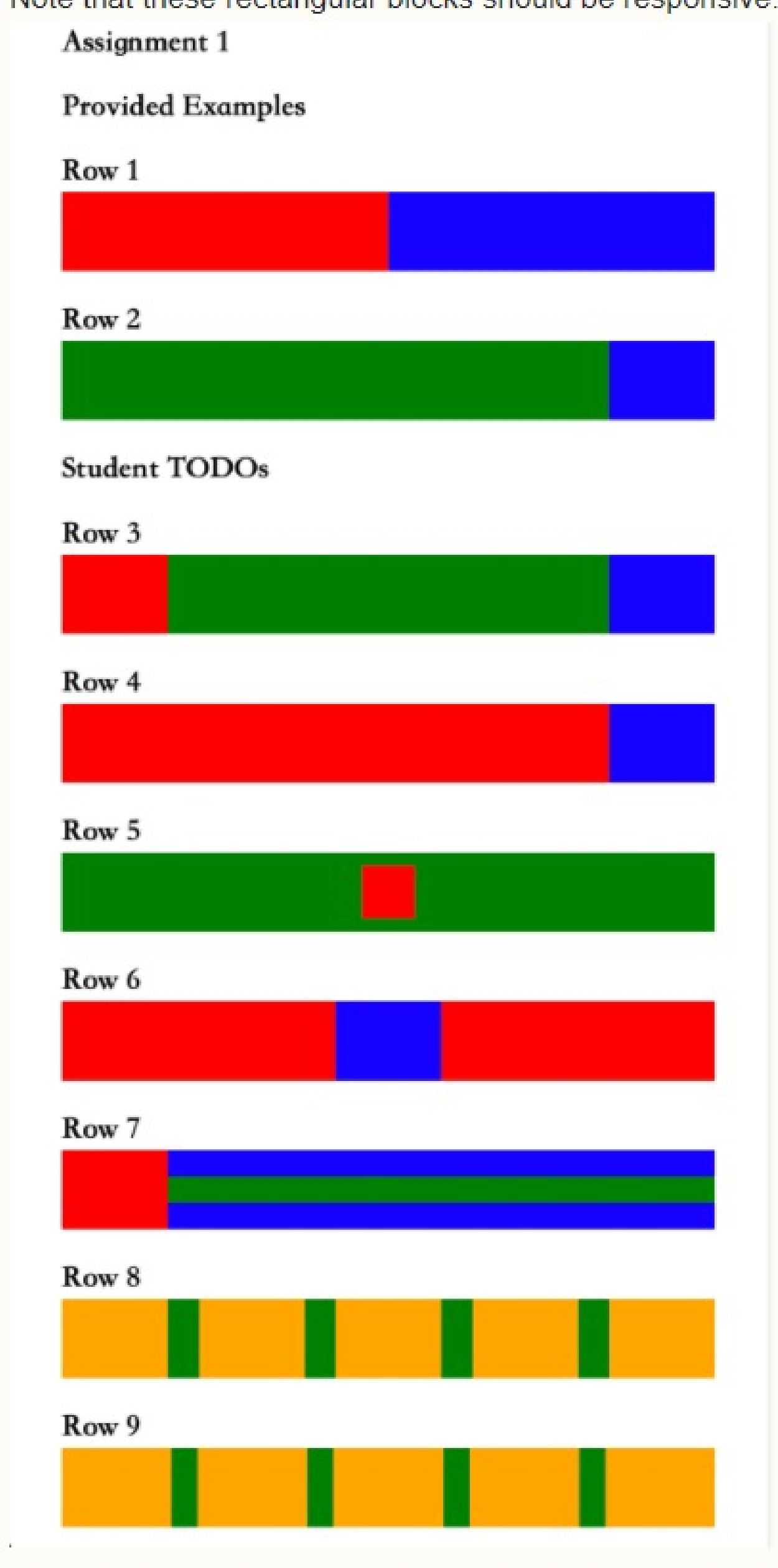
When you open the provided HTML file `part1/index.html` , it should look like this:



As you can see, there are 9 rectangles. The styling and makeup of the first two rectangles are already built for you. Your task is to apply stylings and add div elements inside of the next 7 green rectangular blocks to create a webpage that looks like this:



Note that these rectangular blocks should be responsive. Here is what they look like when the window is thinner:



We will describe how each row will behave dynamically and provide some hints for a possible approach:

3. For the third row, the red and blue end rectangles should remain the same width, and the green space should shrink.

   Possible Approach: Have a div with a red background and a div with a blue background, both with fixed width. Use an appropriate value for Justify Content.

4. For the fourth row, the blue end rectangle should remain the same width, and the red rectangle should shrink.

   Possible Approach: Have a div with a red background and a div with a blue background. Have a fixed width on the blue div. Use Flex Grow.

5. For the fifth row, the red square should remain the same size, but always remain in the center of the green rectangle.

   Hint: Think about how to keep a div fixed size and how to align something in the absolute center of the parent element.

6. For the sixth row, the blue rectangle should remain the same size, while the red rectangles should shrink. The blue rectangle should remain in the center of the row.

   Hint: Use two red divs.

7. For the seventh row, the red rectangle should remain the same width.

   Hint: Nest divs and use `background-color: transparent`

8. For the eighth row, the orange rectangles should remain the same size while the green space between them shrinks.

9. For the ninth row, the green space between the orange rectangles should remain the same width while the orange rectangles narrow.

The examples we provided with the first two rectangular blocks use flexboxes. You are not required to use flexboxes for the next 7 rows, but we recommend it as it will also be useful in part 2 of this assignment.

You should only have to use the `div` html element to complete this assignment. Also, **none of the divs you create inside of the provided wrapper divs should have** `background-color: green;` . But it is valid to specify non-green background colors for any divs, including the wrapper.

Try to style the boxes as closely to the solution image as possible don't worry about getting exact dimensions or rgb values. We care about what structure, CSS styles you used, and the dynamic behavior of the page. However just for reference,

- The color of the boxes we used are `background-color:` `red` , `blue` , and `orange`
- Some width/height values we used are `20px, 40px, 80px`

You are not required to use Bootstrap in this part. You can use if you want, but we actually recommend writing plain CSS. Just for this part, inline CSS is acceptable, but you should generally avoid using inline CSS in the future.

General Notes

As a reminder, it's a good idea to run your HTML and CSS syntax through validators. You should also consider using an accessibility checker such as WAVE.

Troubleshooting

There are hundreds of HTML and CSS tags, properties, and values, and we do not expect students to learn each one by heart. However, this assignment and the first lab are intended for you to intuitively understand the languages, and to be proficient at knowing how to tackle a design by the end of the semester.

If you're having problems, there are many guides on HTML and CSS online (CSSTricks and MDN are your friends), as well as on our resources page.

As always, if you are stuck on a particular part, you can always talk to the friendly TAs or ask questions on course piazza (check your email for a signup link).

As a general rule of thumb, do not expect TAs to be able to solve every web problem you have. Even the most adept web developer can struggle a lot with specific CSS rules to use.