First Exam
CS 2254 Web App Development

KEY

Wednesday March 1, 2017

Instructor Muller
Boston College
Spring 2017

Before reading further, please arrange to have an empty seat on either side of you. Now that you are seated, please write your name **on the back** of this exam.

This is a closed-notes and closed-book exam. Computers, calculators, and books are prohibited.

- Partial credit will be given so be sure to show your work.
- Feel free to write helper functions if you need them.
- **Please write neatly.**

Problem	Points	Out Of
1		6
2		6
3		8
Total		20

1. (6 Points): HTML

1. (3 Points) Draw the document rendered by the following HTML.

```
<html>
  <head></head>
  <body>
    <div>A</div>
    <span>B</span>
    <span>C</span>
    <div>D</div>
  </body>
</html>
```

Answer:

A
B C
D

2. (3 Points) Write HTML that would render the following document. Don't worry about the font size or the exact colors of the backgrounds, any font size and a rough approximation of the background colors will do. (The gray line immediately below is not part of the document.)

Answer:

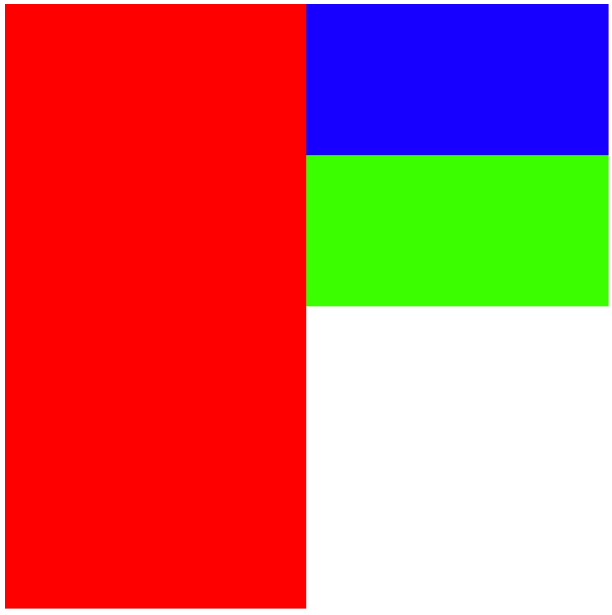
```
<html>
  <head>
  </head>
  <body>
    <span style="background:#0A0A0A; color:#FFFFFF">Boston</span>
    <span style="background:#A0A0A0">College</span>
  </body>
</html>
```

2. (6 Points): CSS

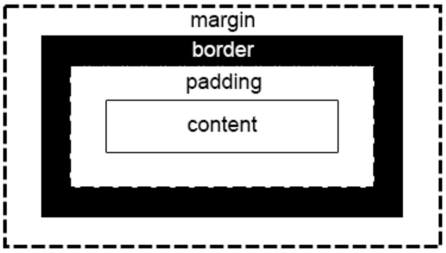
1. (3 Points) Draw the web page rendered by the following.

```
<html>
  <head>
    <style>
      .container { display: flex }
      .a { width: 200px; height: 400px; background: red }
      .b { width: 200px; height: 100px}
      .c { background: blue}
      .d { background: chartreuse}
    </style>
  </head>
  <body>
    <div class="container">
      <div class = "a"></div>
      <div>
        <div class = "b c"></div>
        <div class = "b d"></div>
      </div>
    </div>
  </body>
</html>
```

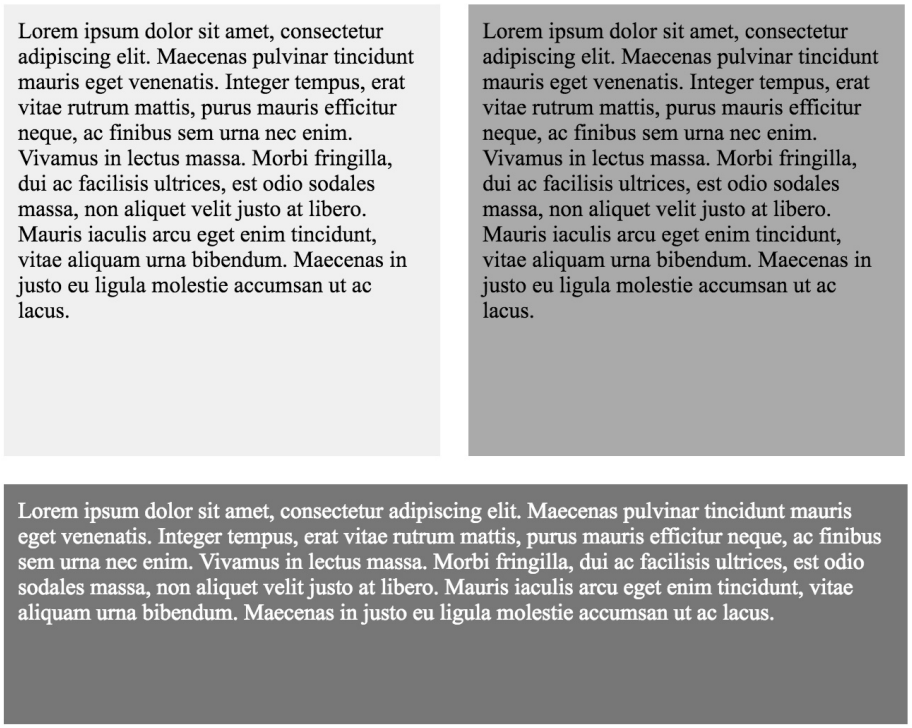
Answer:



2. (3 Points) Remember that the CSS *box model* is



Show CSS + HTML that would generate the following.



The boxes on top are 300px square. The box on the bottom is half that height. Any margins or padding used would be 10px. Don't worry too much about the shades of gray, something approximate will do. And for the text, writing "Lorem ..." will do.

Answer:

```
<html>
<head>
  <style media="screen">
    .container {
      display: flex
    }
    .a {
      width: 300px;
      height: 300px;
      margin: 10px;
      padding: 10px;
    }
    .b { background: #F0F0F0 }
    .c { background: #AAAAAA }
    .d { background: #777777;
        color:white;
        margin: 10px;
        padding: 10px;
        width: 620px;
        height: 150px;
      }
  </style>
</head>
<body>
  <div class = "container">
    <div class = "a b">
      Lorem ...
    </div>
    <div class = "a c">
      Lorem ...
    </div>
  </div>
  <div class = "d">
    Lorem ...
  </div>
</body>
</html>
```

3. (8 Points): JavaScript

1. (2 Points) Consider a JS record with `a` and `b` fields: `{a: "Alice", b: "Tom"}`. Write a JS function `swapEvery : ab-record list -> ab-record list` such that the values of the `a` and `b` fields are swapped. For example, the call

   ```
   swapEvery([{a: "Alice", b: "Tom"}, {a: "Tom", b: "Zhangyang"}])
   ```

   should return the list `[{a: "Tom", b: "Alice"}, {a: "Zhangyang", b: "Tom"}]`.

   Answer:

   ```
   1. let swap = ( ab => {a: ab.b, b: ab.a} )
      function swapEvery(abs) { return abs.map(swap); }

   2. function swapEvery(abs) {
        for (let i = 0; i < abs.length; i++) {
          let tmp = abs[i].a;
          abs[i].a = abs[i].b;
          abs[i].b = tmp;
          }
        return abs;
      }
   ```

2. (1 Point) Did your solution above produce *new* records and a new list or did it rearrange the contents of the old ones?

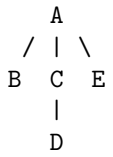
   Answer:

   ```
   1. Both the list and its elements are all brand new.
   2. Both the list and its elements are recycled.
   ```

3. (5 Points) Let's say we have a tree `{info:"A", children:[...]}` where the `children` field is a list of trees. For example, the JS record

```
let t = {info:"A", children:[{info:"B", children:[]},
                             {info:"C", children:[{info:"D", children:[]}]},
                             {info:"E", children:[]}]}
```

would represent the tree



Write a JS function `breadthFirst : tree -> string` such that a call on the above tree would return the result of a *breadth first* traversal of the tree. For example, `breadthFirst(t)` (as shown above) would return the string `"ABCED"`. Of course a breadth first traversal visits all of the nodes on one level before visiting any of the nodes on a lower level.

Answer:

```
function breadthFirst(tree) {
  let queue = [ tree ];
  let answer = "";
  while (queue.length > 0) {
    let t = queue[0];
    answer = answer + t.info;
    queue = queue.splice(1).concat(t.children);
  }
  return answer;
}
```

or

```
function breadthFirst(tree) { return traverse([ tree ]); }

function traverse(queue) {
  if (queue.length === 0)
    return "";
  else {
    let tree = queue[0];
    return tree.info + traverse(queue.splice(1).concat(tree.children));
  }
}
```