1. What is the difference between designing your page using table and div positioning?

Using table positioning

* Ease of use: - the existence of table attributes makes for a rather flat learning curve because the developer doesn’t have to use a separate style sheet. With dives, the developer must use the style attribute or an external style sheet, because the div tag doesn’t have any attributes attached to it.

Tables don’t break when the content is too wide. Columns are not squeezed under other columns as they are in a div-based structure. This adds to the feeling that using tables is safe.

* Maintainability: - Table contains different tags: the table tag being the wrapper, tr for each row and td for each cell. The thead and tbody tags are not used for structural purposes because they add semantic meaning to the content. For readability, each tag is normally given its own line of code, with indention. With all of these tags for the table, several extra lines of code are added to content. The colspan and rowspan attributes make the code even more complex, and any developer maintaining that page in future has to go through **a lot of code to understand** its structure. Which yield hard to maintain the program.

Nested tables are [code smell](http://c2.com/xp/CodeSmell.html) that a website is stuck in table hell. The number of lines of code is endless, and the **complexity** is overwhelming. Tables are far from clean code and don’t bring anything semantic to the content unless you’re dealing with actual tabular data.

Another drawback to tables is that they make it harder to separate content from design.

* **FLEXIBILITY WITH MEDIA: -** In an ideal world, the same markup would be used for printers, mobile devices, screens and other media. Using tables for structure provides less flexibility in moving columns around and hiding entire columns.

Your user may want to put the side column at the bottom or view a printer-friendly version. With tables, this would require **a separate page for each media**, which means extra costs during development and higher maintenance costs compared to a div-based website that separates content and design.

Using DIV positioning

* Ease of use: - Div.-based structures have a much steeper learning curve than table-based structures. The developer must know CSS and understand the difference between block-level elements and inline elements, when to use floats and when to use absolute positioning and how to solve browser bugs.

The div element isn’t visual like the table element. Everyone knows what a table looks like, but divs are not as obvious. The good thing about a div, though, is that it’s only one element. It’s not wrapped in a parent element the way td tags are in tables. The container, then, is **more flexible** and doesn’t have the limitations of its parent tag.

* Maintainability: - The biggest problem with div tags is that **they are used too often**. Dives should only be used to build structure and as placeholders for design elements when no other block-level elements can describe the content. The div tag is for logical groupings of elements.

Nesting div tags deeply is a sure path to maintenance hell, and the code will make developers think twice before touching it, simply because it’s so unreadable. True, using descriptive class and structure names makes the code more understandable, but using them for nested div tags is not always easy.

* **FLEXIBILITY WITH MEDIA:-**  a div-based structure is more flexible in supporting different media than a table-based structure. Not having to maintain separate pages for each media saves maintenance and development costs. A div-based structure allows you to move columns around and even hide columns using display: none in the style sheet.

1. Which do you prefer? Why?

* I prefer using Div. positioning because its flexibility and maintainability, the other reason why I choose DIV for positioning is the div tag groups certain elements on the page together and allows the user to format all of those elements with one same instruction , and the div positioning method separate layout from its presentation.