**different types of css?**  
inline  
internal  
external  
  
**what is opacity?**  
opacity property specifies the transparency of the element   
<div style="opacity:0.1"></div>  
values will be 0 to 1  
  
**float**  
specifies how an element should float  
used t positioning or formatting content  
float:left;  
left-the element floats to the left of its container  
right-the element floats to the right of its container  
none-the element does not float.this is default  
inherit-the element inherits the float value of it is parent  
  
**position**  
specifies the type of positioning method used for an element  
elements are then positioned using the top,left,right,bottom properties  
static- Html elements are positioned static by default  
Static positioned elements are not affected by top,left,bottom,right  
relative-element will get relative position   
it is a normal position  
fixed-it always stay in same position even if the page is scrolled  
absolute-we can move any where in the parent element space  
the parent element should contain relative position  
sticky-positioned based on the user's scroll position  
position:sticky  
position:-webkit-sticky  
IE/Edge 15 earlier version do not support sticky position  
  
**Overlapping elements**  
z-index--> when the element positioned,they can overlap other elements  
z-index:-1  
top element should have higher z-index value

**Table structure**  
<table>  
<tr>  
<th>name</th>  
<th>mobile number</th>  
</tr>  
<tr>  
<td>sakthi</td>  
<td>123456</td>  
</tr>  
<tr>  
<td>rajan</td>  
<td>1234567</td>  
</tr>  
</table>  
  
**List**   
**Ordered list**:list item marked with bullets  
<ol>  
<li>tea</li>  
<li>tea1</li>  
</ol>  
**un ordered list**:list item marked with numbers and letters  
<ul>  
<li>tea</li>  
<li>tea1</li>  
</ul>  
list-style-type: none; ==>we can remove list item style  
list-style-type:bullet,circle,square,upper-roman,lower-alpha  
  
**Display**  
Display property specifies how an element is displayed   
Every HTML element has a default display value depending on what type of element it is.  
The default value of most element is block or inline   
block level elements  
<div>  
<h1>-<h6>  
<p>  
<form>  
<header>  
<footer>  
<section >  
**Inline elements**   
<span>  
<a>  
<img>  
display:none ==> used with java script to hide and show elements without deleting and recreating them  
it does not take up space  
visibility:hidden==>  
It will take up space  
  
**padding**  
used to generate a space around an element's content  
padding-top  
padding-right  
padding-bottom  
padding-left  
negative values are not allowed  
specifies a padding in px,cm,pt  
padding:top right bottom left  
  
**margin**  
used to create space around the elements  
margin-top  
margin-right  
margin-bottom  
margin-left  
negative values are allowed  
specifies a padding in px,cm,pt  
margin:top right bottom left  
  
**Overflow**  
if the content exits   
overflow:visible==>content will not clipped.It will render the outside element box  
overflow:hidden==>the overflow content is clipped and rest of the content hidden  
overflow:scroll==>overflow content is clipped and added to scroll inside the box.this will add a scroll both horizontally and vertically  
overflow:auto==>it is similar to scroll.Only it add scrollbars when necessary  
overflow-x:scroll  
overflow-y:scroll

**Combinators  
space ==>**change all inner p background  
<div>  
<p></p>-->will change  
<p></p>-->will change  
<p></p>-->will change  
</div>  
div p{  
background-color:yellow;  
}  
**(>) ==>**change immediate children tag background means first 2 p tag background color will change  
<div>  
<p></p>-->will change  
<p></p>-->will change  
<span><p></p></span>  
</div>  
div > p {  
background-color: yellow;  
}  
**(+) ==>**immediate following element background means first p tag background color will change(outside div tag p element)  
<div>  
<p></p>  
<p></p>  
</div>  
<p></p>-->will change  
<p></p>  
div + p {  
background-color: yellow;  
}  
**(~) ==>**outside all following element background  
<div>  
<p></p>  
<p></p>  
</div>  
<p></p>-->will change  
<span></span>  
<p></p>-->will change  
div ~ p {  
background-color: yellow;  
}  
  
**pseudo classes==>**used to define a special state of an element  
a:link{  
color:red; // unvisited link  
}  
a:visited{  
color:red; //visited link  
}  
a:hover{  
color:red; //mouse hover link  
}  
a:active{  
color:red; //selected link  
}  
  
**pseudo elements==>**used to style a specific part of an element  
p::first-line {  
color: red; //first line color will change as red  
}  
<p>You can use the ::first-line pseudo-element to add a special effect to the first line of a text. Some more text. And even more, and more, and more, and more, and more, and more, and more, and more, and more, and more, and more, and more.</p>  
p::first-letter {  
color: red; //first letter color will change as red  
}  
<p>You can use the ::first-letter pseudo-element to add a special effect to the first character of a text!</p>  
h1::after {  
content: "tell";// used insert a content after the element content  
}  
<h1>sample</h1>  
h1::before {  
content: "tell";// used insert a content before the element content  
}  
<h1>sample</h1>  
::-moz-selection { /\* Code for Firefox \*/  
color: red;  
background: yellow;  
}  
::selection {  
color: red; //while selecting the h1 and p tag text it would change the text color and background color  
background: yellow;  
}  
<h1>Select some text on this page:</h1>  
<p>This is a paragraph.</p>  
  
**media queries**  
xs>600px  
sm-600px to 768px  
md-768px to 992px  
lg-992px to 1200px  
xl<1200px  
/\* Extra small devices (phones, 600px and down) \*/  
@media only screen and (max-width: 600px) {  
.example {background: red;}  
}  
  
/\* Small devices (portrait tablets and large phones, 600px and up) \*/  
@media only screen and (min-width: 600px) {  
.example {background: green;}  
}  
  
/\* Medium devices (landscape tablets, 768px and up) \*/  
@media only screen and (min-width: 768px) {  
.example {background: blue;}  
}   
  
/\* Large devices (laptops/desktops, 992px and up) \*/  
@media only screen and (min-width: 992px) {  
.example {background: orange;}  
}   
  
/\* Extra large devices (large laptops and desktops, 1200px and up) \*/  
@media only screen and (min-width: 1200px) {  
.example {background: pink;}  
}  
  
**view port**  
user's visible area of web page  
setting view port  
<meta name="viewport" content="width=device-width, initial-scale-1.0">  
width=device-width==>will take device width  
initial-scale-1.0==>initial zoom level when the page is first loaded by the browser  
  
**Flexbox**  
  
  
**What is specificity**  
If we apply same css to the HTML element it would take last one  
Ex  
<div>name<\div>  
div{  
background-color:green   
}  
div{  
background-color:red   
}  
It would take last one means red color will apply   
Ex  
<div class="name">name<\div>  
[div.name](http://div.name/){  
background-color:red   
}  
div{  
background-color:green   
}  
It would take last one means green color will apply   
Ex  
<div class="name" id="name">name<\div>  
div#name{  
background-color:green   
}  
[div.name](http://div.name/){  
background-color:red   
}  
It would take the id css because id will take first and then class css will take  
green color only will take  
Ex  
<div class="name" id="name">name<\div>  
div#name{  
background-color:green   
}  
[div.name](http://div.name/){  
background-color:red !important  
}  
it would take red color !important will take first priority  
  
  
data attribute  
  
**box model**  
boxes wrapped in all the elements   
content  
padding  
border  
margin  
<div></div>  
div{  
height:100px  
width:100px  
padding:10px  
border:20px solid red;  
margin:30px;  
background-color:blue;  
}  
default every box is a content box  
box-sizing:content-box  
  
box-sizing:border-box  
if i give this one the box will come quit smaller because the border will come part of the content box  
  
**triangle css**  
first step  
<div class="tri"></div>  
.tri{  
height:0  
width:0  
background-color:red  
border-top:100px solid blue;  
border-right:100px solid green;  
border-bottom:100px solid red;  
border-left:100px solid yellow;  
}  
second step  
<div class="tri"></div>  
.tri{  
height:0  
width:0  
//background-color:red  
border-top:100px solid transparent;  
border-right:100px solid green;  
border-bottom:100px solid transparent;  
//border-left:100px solid yellow;  
}  
  
**div center css**  
<div class="out">  
<div class="in">  
  
</div>  
</div>  
.out{  
height:300px  
width:300px  
position:relative  
background-color:red  
}  
.in{  
height:100px  
width:100px  
position:absolute  
background-color:yellow  
top:50%  
left:50%  
transform:translate(-50%,-50%)  
}